Genetic Algorithm-based Ensemble Methods for Large-Scale Biological Data Classification

By

Mohammad Nazmul Haque

MSc (DIU), BSc (DIU)

Thesis

submitted in fulfilment of the requirements for the Degree of

Doctor of Philosophy

School of Electrical Engineering and Computer Science
The University of Newcastle
Callaghan, New South Wales 2308, Australia
February 2017

Statement of Originality

The thesis contains no material which has been accepted for the award of any other degree or diploma in any university or other tertiary institution and, to the best of my knowledge and belief, contains no material previously published or written by another person, except where due reference has been made in the text. I give consent to the final version of my thesis being made available worldwide when deposited in the University's Digital Repository**, subject to the provisions of the Copyright Act 1968.

**Unless an Embargo has been approved for a determined period.

Mohammad Nazmul Haque, February 2017

Statement of Collaboration

I hereby certify that the work embodied in this thesis has been done in collaboration with Dr Nasimul Noman and Ms. Natalie De Vries from the Priority Research Centre for Bioinformatics, Biomarker Discovery and Information-Based Medicine (CIBM) and the School of Electrical Engineering and Computer Science, Faculty of Engineering and Built Environment, The University of Newcastle, Callaghan, New South Wales, Australia. I have included as part of the thesis a statement clearly outlining the extent of collaboration, with whom under which auspices.

Mohammad Nazmul Haque, February 2017

Statement of Authorship

By signing below I confirm that the work embodied in this thesis contains following published paper work of which **Mohammad Nazmul Haque** is the leading author. Chapter 3 contains parts of the paper "Heterogeneous Ensemble Combination Search Using Genetic Algorithm for Class Imbalanced Data Classification" published on the PLoS ONE, Volume 11, Issue 1 ([Haque et al., 2016a]). Chapter 4 contains the part of the paper "Optimising Weights for Heterogeneous Ensemble of Classifiers with Differential Evolution" has been published in the conference proceedings of IEEE CEC 2016 [Haque et al., 2016b]. Chapter 5 contains the part of the accepted chapter "A Multi-objective Meta-Analytic Method for Churn Prediction" authored by Mohammad Nazmul Haque, Natalie De Vries and Pablo Moscato in the 1st edition of "Business and Consumer Analytics: New Directions" book. These publications are the result of collaborative work with Prof. Pablo Moscato, the principal supervisor, Prof. Regina Berretta, the co-supervisor and, Dr Nasimul Noman and Ms. Natalie De Vries, collaborators from the Priority Research Centre for Bioinformatics, Biomarker Discovery and Information-Based Medicine (CIBM). It is worth mentioning that **Mohammad Nazmul Haque** had the active role in every stage of the study design, data collection and analysis, software development, conducting the experiment, result analysis and manuscript preparation.

Mohammad Nazmul Haque, February 2017

Dr Nasimul Noman, February 2017

Ms. Natalie De Vries, February 2017

Prof. Regina Berretta, February 2017

Prof. Pablo Moscato, February 2017

Acknowledgements

First and above all, Alhamdulillah, all praises to Allah for the strengths and HIS blessings to complete this thesis. This thesis reached in its current state due to the guidance and assistance of several people. I, therefore, would like to pay my sincere thanks to all of them.

I would like to give my heartfelt gratitude to my principle supervisor Prof. Pablo Moscato, for his constructive ideas, feedbacks and support throughout my PhD candidature. The achievements in this thesis would not be possible without his helpful and visionary supervision. I would like to give thanks to my co-supervisor Assoc. Prof. Regina Berretta for continuous correction works to my writings and publications. I would like to give special thanks to Dr. Nasimul Noman for his valuable directions and ideas to analyse the results, design the solutions and structuring contents for publications. This PhD would not be finished so smoothly without their help.

I would like to give a special thank to the former computational scientist at our centre, Dr. Carlos Riveros, for his valuable discussions and ideas for pre-processing of datasets. Also, I would like to extend thanks to my lab mates Dr. Ahmed Shamsul Arefin, Dr. Luke Mathieson, Natalie De Vries, Heloisa Milioli, Amir Salehipour, Leila Moslemi, Nisha Puthiyedth, Amer Abu Zaher, Shannon Fenn, Inna Tishchenko, Claudio Sanhueza, Francia Jimenez and Ademir Gabardo: for their numerous supports. Outside of the research group, I would like to thank my confirmation committee: Dr. Nasimul Noman, Dr. Yuqing Lin and Dr. Alexandre Mendes.

I would like to acknowledge my colleagues & friends for their tremendous support during the preparation and application period of the PhD admission. Especially, Syed Ashiqur Rahman, you are not only my mentor but also a role model. Special thanks to Kamanashis Biswas, Mia Md. Keyam Uddin, Safiqul Islam Mithun, Mehnaz Tabassum, Jashim Uddin and my other colleagues and friends from Daffodil International University (DIU). I would like to express my gratitude to Prof. Mohamad Shorif Uddin, Dr. Mohammad Abdullah-Al-Wadud, Prof. Dr. Abu Taher, Prof.Syed Akhter Hossain, Prof. Yousuf M. Islam and the faculty members of DIU for their inspirations, recommendations

and encouragements towards my academic career. I would also like to thank the authority of DIU for approving the study leave from my position at the Department of Computer Science to pursue the PhD research.

Thanks to my childhood friend Rokibul Islam and Taysia Kabir Inti, for settling my family down during the beginning of the life here in Newcastle and eventually continuously supporting. Thanks to all my childhood friends in Bangladesh, Bangladeshi Students' Association of Newcastle University (BSANU) and Bangladeshi Community in Newcastle for the joyful gatherings and all their supports.

A very special thank to The University of Newcastle, Australia for providing me the opportunity to pursue my PhD research with the financial support of 2012 UNIPRS and 2012 UNRSC scholarships.

Finally, a lot of thanks to my wife, Shohana Pervin, for her patience, sacrifices, hard work and supports in transforming the PhD journey a smooth one. Without her contributions and support, this work would not have been completed. Our little princess, Nabiha Shanum Nuhaa, you have shown maturity, generosity and serenity while passing a hard time with us with a smiley face, spared me space for study and provided me the power to face all odds. Both of you, in pairs, deserve the award of the best supportive persons in the world helping towards my PhD. I also wanted to express my gratitude to both of my family and in-laws family for supporting us by all their means for this journey of the PhD.

To
my little princess Nabiha Shanum Nuhaa,
my wife Shohana Pervin,
&
my parents.

Contents

List of Figures

List of Tables

List of Algorithms

Abstract

We study the search for the best ensemble combinations from the wide variety of heterogeneous base classifiers. The number of possible ways to create the ensemble with a large number of base classifiers is exponential to the base classifiers pool size. To search for the best combinations from that wide search space is not suitable for exhaustive search because of it's exponential growth with the ensemble size. Hence, we employed a genetic algorithm to find the best ensemble combinations from a pool of heterogeneous base classifiers. The classification decisions of base classifiers are combined using the popular majority vote approach. We used random sub-sampling for balancing the class distributions in the class-imbalanced datasets. The empirical result on benchmarking and real-world datasets apparently outperformed the performances of base classifiers and other state-of-the-art ensemble methods. Afterwards, we evaluated the performance of an ensemble of classifiers combination search in a weighted voting approach using the differential evolution (DE) algorithm to find if employing weights could increase the generalisation performances of ensembles. The weights optimised by DE also outperformed both of the base classifiers and other ensembles for benchmarking and real-world biological datasets. Finally, we extend the majority voting-based ensemble of classifiers combination search with multi-objective settings. The search space is spread over the all possible ensemble combinations created with 29 heterogeneous base classifiers and the selection of feature subset from six feature selection methods as wrapper approach. The optimisation of two objectives, the maximisation of training MCC scores and maximisation of the diversity among base classifiers, with NSGA-II, a popular multi-objective genetic algorithm, is used for simultaneously finding the best feature set and the ensemble combinations. We analyse the Pareto front of solutions obtained by NSGA-II for their generalisation performances. Datasets taken from UCI machine learning repository and NIPS2003 feature selection challenges have been used to investigate the performance of proposed method. The experimental outcomes suggest that the proposed multiobjective-based NSGA-II found the better feature set and the best ensemble combination that produces better generalisation performances in compared to other ensemble of classifiers methods.

1 Introduction

Machine learning facilitates a structured methodology to acquire, quantify and use knowledge that can be extracted from data. In contrast, learning in nature is not directed by any defined formula but is obtained from data. Machine learning is a widely used method in science and engineering, among other fields, that mimics the natural learning process of knowledge from data. Perhaps the most common method of data analysis and pattern mining is predicting data membership. The membership prediction by a model starts with the training of a classification algorithm with labelled data. Then, the model utilises this knowledge to assign a predefined categorical label to each object from an unknown dataset. Objects are characterised by a set of measurements named attributes or features. Selection of an appropriate feature set is crucial for the training of machine learning algorithms and referred to as 'feature selection' (FS). One of the major goals of training a classifier is that it shall perform (generalisation performance) well when it is applied to new data. The performance of a classifier for different types of datasets varies a great deal. No single classifier exists that performs extremely well for all datasets.

The ensemble of classifiers (EoC) is a methodology of combining multiple trained learning machines by treating them as a 'committee' of expert decision makers. The principle is to combine individual expert predictions in an appropriate manner to achieve better overall accuracy, on the average, than any individual committee member could accomplish on

its own. Several studies have demonstrated that ensemble models often outperform the accuracy of single classifier models. However, creating an EoC using selected base classifiers from a heterogeneous set might improve the generalisation performances more than the ensembles created with all base classifiers. Because not all classifiers perform well in all cases, the optimisation of ensemble combinations could possibly select the best set of classifiers for the problem in hand.

This inspirational research aimed to address the biological datasets' classification problem, using the best EoC combination selected from a heterogeneous pool to provide better predictive accuracy than that of conventional methods.

1.1 Biological Background

This PhD research project deals with biological data (microarray gene expression) as a data source for the classification method. First, we briefly describe the central dogma of molecular biology, followed by the microarray technology and gene expression data.

1.1.1 Central Dogma of Biology

First, we will recall that the 'central dogma' described in [Crick, 1970]: deoxyribonucleic acid (DNA) is a nucleic acid that stores genetic information needed for the development and functioning of all living organisms. The DNA information is replicated into DNA or transcribed into a ribonucleic acid (RNA) and then translated into a protein by ribosomes. DNA and RNA are composed of sequences of four nucleotides (noted by the letters G, A, T and C, for the nucleotides guanine, adenine, thymine and cytosine, respectively, in DNA; and messenger RNA (mRNA), which contains uracil, denoted by U, in place of T). The ribosomes read the mRNA triplet codons, which link amino acids in a specific order to build a peptide chain. Proteins carry out most of the cell functions, including regulation of replication, transcription and translation. The general process of the flow of information is expressed as:

$$DNA \underset{transcribed}{\longrightarrow} mRNA \underset{translated}{\longrightarrow} protein$$

The expression of an mRNA sequence, or a gene, provides an indirect estimation of protein abundance and can be monitored for determining biological functions. It is the fundamental level at which the genotype gives rise to the phenotype, that is, the observable condition or disease state.

1.1.2 Microarray Technology

Microarray technology captures the expression levels of many genes at once. This technology produces a large number of measurements, even for a small number of samples. A DNA microarray is an array of thousands of spots; each spot contains thousands of probes, DNA single strands, for a specific gene. These probes are attached to a solid surface (glass, plastic or silicon) and are complementary to the cDNA (cDNA is double-strand DNA synthesised from a single strand RNA) that is copied from the mRNA sequences, matching the different genes.

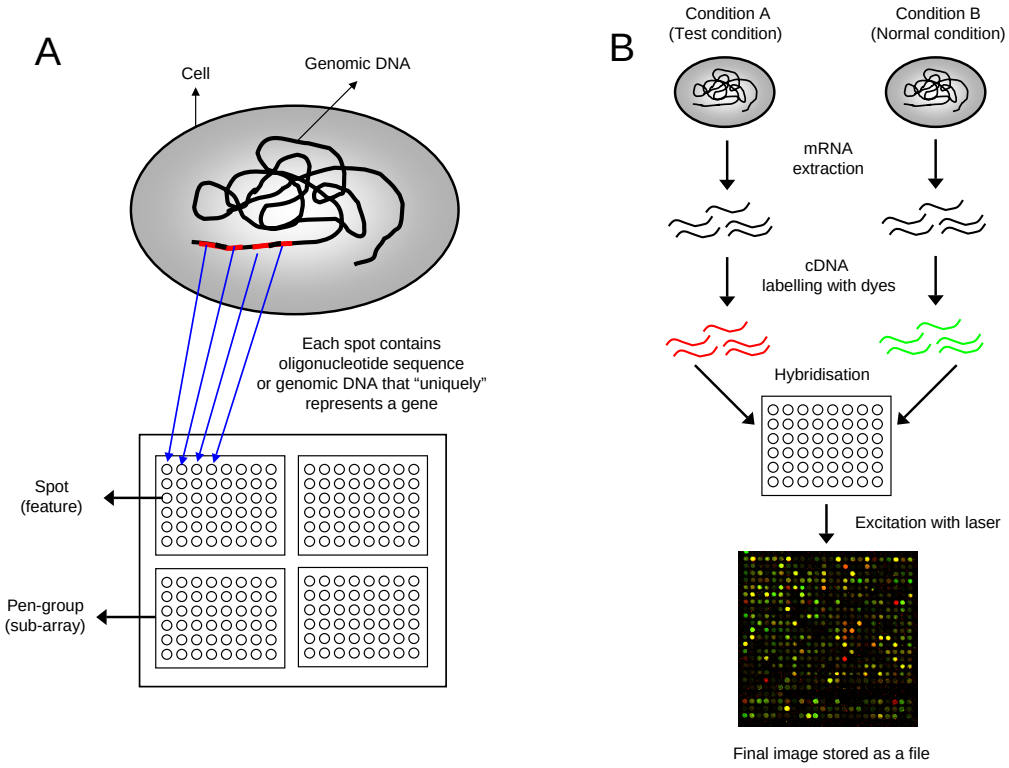


Figure 1.1: A schematic view of microarray experiments. (A) Microarray containing thousands of spots, where each spot contains DNA for a specific gene. (B) Schematic of the experimental protocol. (The figure is reproduced from [Babu, 2004] with written permission Appendix C.)

The first step is to extract the mRNA from two different samples (tissues or cell cultures) representing two different clinical conditions (e.g., normal and cancerous). The mRNA needs to be reversely transcribed into cDNA and marked using two different fluorescent dyes. Red is usually used for samples extracted from cancerous tissue and green is used for the normal tissue samples [Cohen, 2007]. Then, both samples are spread across

the microarray chip and left to hybridise to the complementary probes, forming a double-strand model. The hybridisation is identified based on fluorescence detection. A special scanner detects and records the intensity of the dyes for each spot. The levels of expression are then measured by comparing the colours of the two samples. An image is generated that contains four colours representing the abundance of genes. If genes are expressed in both samples, the colour will be yellow; while black represents genes that are not expressed in either of the samples. If genes vary in expression in one sample, the colour will be red (up-regulated) or green (down-regulated). The intensity of the colours indicates the level of expression. A schematic view of the microarray experiment is depicted in Figure 1.1.

The next step is to store the relative expression level of each gene as an image. The ratio of expression values (the most popular is log ratio: $\log_2(I_r/I_g)$) of two intensity levels of colour (I_g and I_r represent the intensity of green and red, respectively) is extracted from the image and forms the microarray data. These data are normalised and represented in the form of a matrix, called a gene expression matrix [Babu, 2004]. These data are used for further processing.

1.1.3 Gene Expression Dataset

This project deals with the gene expression microarray (GEM) data. A typical gene expression dataset can be represented by a matrix D of real numbers. Let us consider that the microarray dataset contains m rows (probes/genes or features) and n columns (samples). D is represented in Equation (1.1) as:

$$D_{m,n} = \begin{bmatrix} r_{1,1} & r_{1,2} & \cdots & r_{1,n} \\ r_{2,1} & r_{2,2} & \cdots & r_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ r_{m,1} & r_{m,2} & \cdots & r_{m,n} \end{bmatrix} \tag{1.1}$$

where r_{ij} represents the value of the expression ratio for the ith gene in the jth sample.

Typically, there are thousands of gene expression values m (usually defined by the ratios or log ratios instead of absolute gene expression values towards a more comparable range) and a smaller number of samples n available [Wang et al., 2008]. This makes the dataset difficult to examine, and many traditional methods in machine learning cannot handle these high-dimensional skewed data ($m \gg n$). To learn properly the data characteristics by the machine learning algorithm, a rule of thumb is to have at least 10 training samples per feature dimension [Kuncheva and Jain, 2000], whereas in microarrays, this ratio is often closer to 0.01 samples per feature [Allison et al., 2006]. This may also bring a class

imbalance to the dataset, where the number of samples belonging to each class is not the same. A class imbalance is common in the biomedical field, and it usually arises from high-dimensional microarray data [Blagus and Lusa, 2010]. Therefore, gene expression datasets require methods that can deal with class imbalance and high-dimensional data.

1.2 Pattern-Recognition Process

Current research on pattern recognition builds on foundations established in the 1960s and 1970s [Kuncheva, 2004]. In pattern recognition, machine learning and data mining, one important task is to create the appropriate ***data model***. A 'model' is commonly referred to as the learning or building structure of the data from the dataset [Zhou, 2012], whereas a 'dataset' usually consists of *feature vectors* (sometimes called an *instance* or an *example*). An object in the dataset (also referred to as a ***sample***) is described by a set of **features** (also denoted by *attributes*) from a feature vector. The ***dimensionality*** of the dataset is the number of features times the number of samples. The process of generating models from the dataset is called *learning* or training, which is accomplished by a *learning algorithm*. The learned model can be called a *learner*. Based on the learning process of the model (as there are many other classifications available depending considering other factors), a pattern-recognition problem can be divided into two major types–unsupervised and supervised.

Unsupervised: In the unsupervised category (also called *unsupervised learning*), we need to discover the structure of the dataset, if any. Many clustering algorithms have been developed to solve unsupervised pattern-recognition problems.

Supervised: In the supervised category (also called *supervised learning* or *classification*), each object in the dataset comes with an associated class label. We need to train a *classifier* to discover the class labels for unknown objects.

The pattern-recognition process in general can be seen in Figure 1.2. Suppose that a hypothetical user presents a question along with a dataset. Our task is to understand the question, and represent it using data classification terminologies. Then, we need to adopt either supervised or unsupervised methods, depending on the problem type. If the user wants to know whether any group exists in the datasets, and what the available characteristics are that can differentiate object groups, we apply the unsupervised-learning methods and use clustering algorithms. Because of the unavailability of any ground truth results, the usefulness of the clustering result is indicated by the subjective interpretation

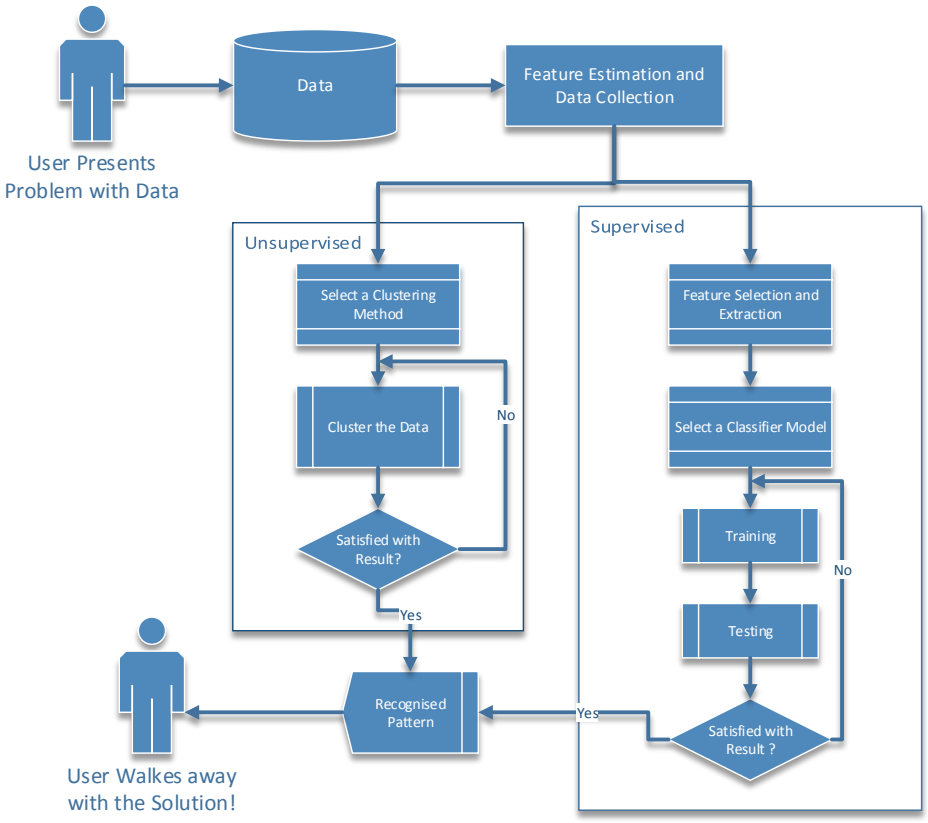


FIGURE 1.2: Typical pattern-recognition process. (The figure is reproduced from [Kuncheva, 2004] with Permission Number 3711660699232.)

given by the user. Conversely, if the user has some ground truth results, we can apply *supervised* techniques. Here, we need to train a classifier from the pre-assigned set of class labels. The classification knowledge learned by the machine is assessed using the prediction accuracy. When the learning process reaches a satisfactory level, the learner is then used to do the class labelling for the unlabelled dataset.

1.2.1 Data Classification

Data classification is a scientific discipline that studies the different ways of assigning class membership values (labels) to unknown observations, or to samples based on the learning from a set of known observations provided with labels. The task of data classification (supervised learning) uses the available training examples to construct a model that can be used to predict the targets of unseen data, which are assumed to follow the same

probability distribution as the available training data. Then, we need to *estimate the performance* of the model for a predictive task. The predictive capability of the trained model is evaluated by the generalisation ability from the training examples to unseen data. One possible definition of supervised learning for binary-class [Vapnik, 1999] is presented as follows:

Definition 1.1. The problem of supervised learning is to choose from the given set of functions $f \in F : X \mapsto Y$ based on a training set of random independent identically distributed (*i.i.d.*) observations drawn from an unknown probability distribution $P(\mathbf{x})$, such that the obtained function $f(\mathbf{x})$ best predicts the supervisor's response for unseen examples $(\mathbf{x}, y)$,

$$(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_n, y_n) \in X \times Y, \tag{1.2}$$

which are assumed to follow the same probability distribution $P(\mathbf{x}, y)$ as the training dataset in Equation (1.2). Here, $X = \{\mathbf{x}_1, \cdots, \mathbf{x}_n\}$ denotes the set of instances or samples and $Y = \{y_1, \cdots, y_n\}$ is the set of corresponding class labels. Each sample or object is characterised by an m-dimensional vector $\mathbf{x} = [x_1, \cdots, x_m]^T \in \Re^m$ from *feature space* $\Re^m$ and is associated with one, and only one, class label (ω_i) from the set of labels $\Omega = \{\omega_1, \cdots, \omega_c\}$.

A *classifier* $\mathbb{C}$ is a function which learns the mapping or decision function of feature set $(\Re^m)$ from the set $\mathbb{T} = \{x_1, \cdots, x_m\}, x_i \in \Re^m$ of training samples to the class label set $\Omega = \{\omega_1, \cdots, \omega_c\}$; that is:

$$\mathbb{C} : R^m \xrightarrow{\mathbb{T}} \Omega. \tag{1.3}$$

1.2.2 Classification Performance Measures

The importance of a classifier's prediction performance is critical. The prediction performance of a binary-classifier is usually reported using a two row by two column matrix, widely known as a *confusion matrix* [Stehman, 1997]. The four outcomes of a classification problem can be summarised in the confusion matrix as shown in Figure 1.3. Here, the outcomes are labelled as belonging to either the positive (Pos) or the negative (Neg) class. If the outcome from a prediction and the actual value is Pos, then it is called a true positive (TP); however, if the actual value is Neg then it is said to be a false positive (FP). Conversely, when both predicted and actual outcomes are Neg, then it is denoted as a true negative (TN). It is denoted as a false negative (FN) when the prediction outcome is Neg while the actual value is Pos.

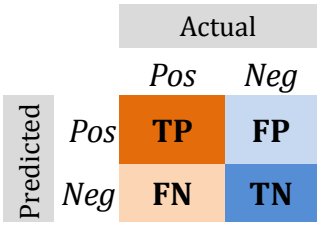


FIGURE 1.3: A confusion matrix summarises all possible outcomes in a binary-classification problem.

$$Accuracy = \frac{(TP + TN)}{(TP + TN + FP + FN)} \tag{1.4}$$

$$Recall/Sensitivity = \frac{TP}{(TP + FN)} \tag{1.5}$$

$$Specificity = \frac{TN}{(TN + FP)} \tag{1.6}$$

$$Precision = \frac{TP}{(TP + FP)} \tag{1.7}$$

$$F\text{-}Measure = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{1.8}$$

$$MCC = \frac{(TP \times TN) - (FP \times FN)}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{1.9}$$

Different measures of classifier performance can be calculated from the confusion matrix, as shown in Equations 1.4–1.9. The classification accuracy is one of the most significant metrics. Although classification accuracy is a measure used for comparing the prediction performance, it is not suitable for use with imbalanced data when the sample sizes of the two classes differ greatly. It only considers the proportion of correct classifications without considering the class distribution. This can result in the poor rating for the minority class being easily overwhelmed by the correct classification of the majority class. We note that identifying the minority classes can actually be an important outcome in many cases, such as fraud detection, cancer detection and intrusion detection. Thus, the more appropriate evaluation criteria are those that use all values from the confusion matrix for binary-classification problems. Other popular measures for comparing performance of different classifiers are *sensitivity* and *specificity*. Sensitivity (SEN) is measured for the

probability of predicting positive outcomes when the true states are positives (it is also referred as *true positive rate*), whereas specificity (SPEC) is measured by the probability of predicting negative outcomes when the true state of the cases are negatives (it is also referred as *true negative rate*). The *F-measure* is another measure of binary classification accuracy which considers both the *precision* and the *recall* values to compute the score. It does not use the TN values from the confusion matrix to calculate the score. While there is no perfect way of describing the confusion matrix of TP, FP, FN and TN by a single number, the Matthews correlation coefficient (MCC) quantifies the strength of the classifications using the confusion matrix. The MCC takes into account both the sensitivity and specificity, and can often provide a more balanced accuracy assessment of the model [Dutt and Madan, 2012]. An MCC with a higher value indicates better predictions. It is commonly used as a performance measure in the reference imbalanced datasets in different fields of machine learning, including bioinformatics. Notably, the MCC was chosen as an index of classifiers performance measure in the United States led initiative FDA MAQC-II based on microarray gene expression and genotyping data [Shi et al., 2010]. Further, for the analysis of the confusion matrix, the experimental results obtained by [Jurman et al., 2012b] indicate that the MCC is an ideal measure for the practical classification task. Thus, we consider the MCC our measure of classification performance.

1.2.3 Feature Selection

FS is one of the principal branches of machine learning. It is treated as a prerequisite for the classification problem, as shown in Figure 1.2. High-dimensional biological datasets contain a large number of features. It could be thought that having more features in a classification would bring more discriminating power. However, in reality, redundant and irrelevant features increase the complexity of the classification problem and degrade the classification accuracy.

Thus, some features have to be removed from the original feature set to mitigate these negative effects before being applied for classification. The task of removing redundant and irrelevant features from the original dataset is known as FS in the machine learning and data mining literature. It is the process for dimensionality reduction of data, where the original set of features $\mathcal{F}$ is reduced to another set $\mathcal{G} \subseteq \mathcal{F}$. This reduced feature subset usually contains important and non-redundant features that can best characterise the dataset.

For a microarray dataset, obtaining an irreducible set of features is a tough job because it contains tens of thousands of features (gene expression intensity values) in each dataset. However, one feature of the microarray dataset is that the number of samples collected

tends to be much smaller than the number of genes. At the same time, many genes do not contribute to disease diagnosis [Liu and Xu, 2009]. A widely used solution for this problem is to utilise the FS method to select a set of biologically significant genes. Thus, gene selection becomes similar to FS in data mining and machine learning.

1.3 Ensemble of Learning Machines

Recent advances in machine learning theory have extended the research areas to increase the capabilities of basic learning methods. These learning methods have been called 'multiple classifier systems' or 'committee machines' or 'ensembles'. The core principle of ensemble learning is to weigh several individual pattern classifiers, and then combine them to gain better accuracy than can be obtained by using each of them separately. In the machine learning paradigm, ensemble data mining methods strategically advance the power of committee methods, or combine models to achieve better prediction accuracy than any of the individual models could achieve [Oza, 2006]. The basic goal when designing an ensemble is to use independent models so that the combined model will produce a better performance. An empirical study on the classification method has shown that some classifiers perform best in some domains, but not in all application domains. This phenomenon was termed the *no free lunch theorem* by [Wolpert, 1996]:

> *'if one inducer is better than another in some domains, then there are necessarily other domains in which this relationship is reverse'*

Here, the inducer is the classifier, applicable for a specific domain. This *no free lunch* phenomenon presents a dilemma in selecting the best classifier for a specific domain. The ensemble learning methods (defined in Definition 1.2) can overcome this dilemma by combining the output of many independent classifiers that perform well in certain domains, but are suboptimal in others.

The reason behind nonexistence of perfect classifier could possibly be explained with the help from Figure 1.4. A classifier may be away from the best solution (point 2 in the figure) even with a perfect training algorithm (point 1 in the figure). The difference between these two points is called *approximation error*. This error appeared from the fact that we have a finite amount of training data and getting the optimal classifier is not always guaranteed. Point 3 denotes the best possible solution using given feature space and which may even not be achievable by the current classifier. Additionally, due to the *model error*, the optimal point (point 3 in the figure) could also not be achievable by the current classifier. Finally, the *bayes error* (which is irreducible) appeared for the

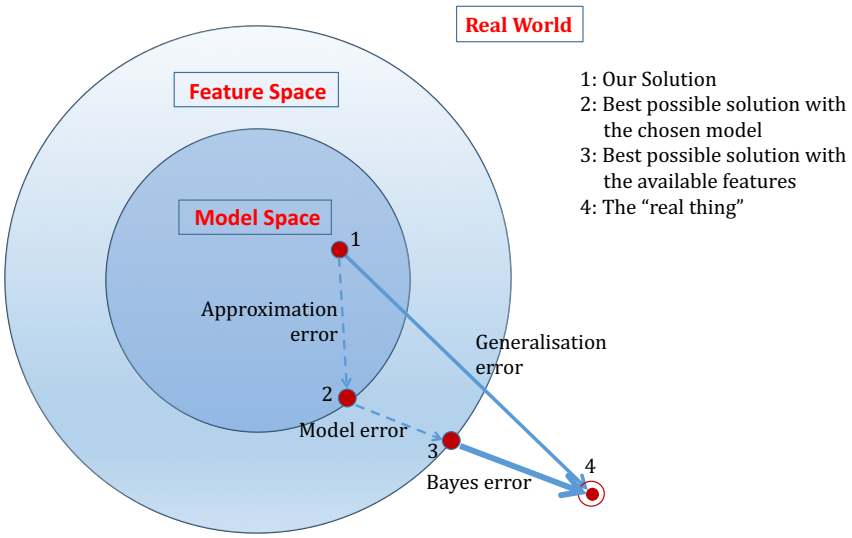


Figure 1.4: The composition of the generalisation error of classifier. (The figure is adapted from [Kuncheva, 2014] with Permission Number 3955870396941.)

insufficient representation of the feature. Thus, the *generalisation error* of a classifier trained on the given feature space can be expressed as:

$$Generalisation_{error} = Approximation_{error} + Model_{error} + Bayes_{error}. \tag{1.10}$$

The first term ($Approximation_{error}$) of the Equation 1.10 can be thought as *variance* due to using different training data. The $Model_{error}$ can be taken as the *bias* of the model from the possible solution [Kuncheva, 2014].

An ensemble of classifiers is used to collectively reduce the bias and variance error. In general, if we have n independent samples from a probability distribution with variance σ and mean μ, and we combine them (using average operation), then the new mean is the same (μ), but the new variance will be reduced ($\frac{\sigma}{n}$). If samples have some correlation ρ, the reduction in variance depends on the correlation (a low correlation means a big reduction in variance) [James et al., 2014]. Now, suppose we have a way to generate a model which has low bias and high variance. If we could generate multiple instances of such classifier models, then the final aggregated (averaged) model (ensemble of classifiers) should have the same bias (mean), but lower variance. Because of the bias-variance decomposition, this means that the ensemble of classifiers will have a lower error than the participating

models and will produce better generalisation performances.

Definition 1.2. (Ensemble Learning). An ensemble of learning machines is a set of learning machines that learn partial solutions to a given problem, and then integrate these solutions in some manner to construct a final or complete solution to the original problem.

Using a set of k individual base classifiers $\mathbb{C}_i$ denoted by $\mathbb{C}^* = \{\mathbb{C}_1, \cdots, \mathbb{C}_k\}$ for $i = (1, \cdots, k)$, a common example of an ensemble is

$$\mathbb{E}\left(\mathbb{C}^*\right) = \prod_{i=1}^{k} w_i \left(\mathbb{C}_i : \Re^m \underset{\mathbb{T}}{\longmapsto} \Omega\right), \tag{1.11}$$

where $w_i \in \mathbb{R}$ is the measure of goodness for the ith base classifier $\mathbb{C}_i \in \mathbb{C}^*$ used to form the EoC $\mathbb{E}$.

A typical ensemble method for classification tasks contains the following building blocks [Rokach, 2009]:

Training set: It is a labelled dataset used for training the ensemble. We use the notation $\mathbb{T} \in (\mathbf{x}, \Omega)$ as expressed in Equation (1.12). The training dataset contains n labelled objects and each object contains m attributes.

$$\mathbb{T} = \{t_1, \cdots, t_n\}; t_i \in \Re^m \tag{1.12}$$

Base Classifier: The base classifier $\mathbb{C}$ is an induction algorithm to form a generalised relationship among feature vector $\mathbf{x}$ and corresponding label ω from $\Omega = \{\omega_1, \cdots, \omega_c\}$ using the training set $\mathbb{T}$ as expressed in Equation (1.3).

Diversity Generator: This component is responsible for generating the diverse individual learners from labelled training data.

Combiner: The combiner or decision fusion is responsible for combining the set of base classifiers to achieve strong generalisation ability, rather than trying to find the best single learner.

Ensemble methods have been used by researchers from various disciplines such as pattern recognition, statistics and machine learning.

1.3.1 Design Process of the Ensemble of Classifiers

Although every ensemble method combines the outcome of multiple classifiers into a single decision, the building paradigms of the EoC usually differ from each other. They differ

with respect to the diversity generation mechanism among base classifiers and the strategy of combining them.

Let us briefly consider these building blocks of designing the EoC.

Diversity in Ensemble of Classifiers

Using diverse single classifiers to form an ensemble is the key issue to improving its generalisation performance because it is expected that individual classifiers make mistakes on different instances. Then, a strategic combination of these diverse classifiers can reduce the total error [Polikar, 2006]. The diversity can be achieved in many ways. Research shows that, based on the use of the training set, we can divide the diversity achievement in ensembles into three broad categories.

1. **Different Classifier:** To attain diversity among base classifiers, this method is used to train all single classifiers using the same training set. Here, they use different learning algorithms as base classifiers, or their variations of parameters of the base classifiers. This is known as the *heterogeneous* ensemble method [Tsoumakas et al., 2004].

2. **Different Training Set:** This diversity generation process uses different training sets obtained from the original training set by resampling techniques. Those training sets are used to train the base classifiers. The state-of-the-art ensembles, namely `Bagging` [Breiman, 1996] and `AdaBoost` [Freund and Schapire, 1996], manipulate training data using this approach. They use multiple instances of base classifiers trained on different training samples. This is called the *homogeneous* ensemble method.

3. **Ensemble Feature Selection:** Diversity can also be generated by training the individual classifiers with datasets that consist of different feature subsets, like the Random Subspace Method (RSM) [Ho, 1998]. In the ensemble FS, an ensemble method is used to select a feature subspace. Base classifiers are then trained using those subsamples of features.

Ensemble methods can use any one of the alternatives mentioned above to generate diverse individual base classifiers. To determine the final decision, the process of combining the outcomes comes next in the process.

Designing the Combination Rule

The combination rule is the second key component of the ensemble method. It helps to reach a final decision by integrating all participating single classifiers' outcomes. The combination rule can generally be designed in two ways.

1. **Classifier Fusion:** In this design, each classifier is trained over the entire feature space. Then, results of individual classifiers are combined in an appropriate manner to reach a final decision. The fusion method can use majority voting, weighted-majority voting, summation, product, maximum and minimum, fuzzy integral, the Dempster–Shafer-based combiner or decision templates to decide the final outcome [Woods et al., 1997, Kuncheva, 2004, Polikar, 2006] of a testing sample.

2. **Classifier Selection:** This design of combination method uses domain expert classifiers in the training phase. Here, each classifier is trained to become expert in a specific part of the total feature space. In the decision prediction stage, only one base classifier is used instead of a combination of all the decisions of all available classifiers. The selection of a final predictor classifier, among available expert single classifiers, can be done in two ways. These are the *static classifier selection (SCS)* method and the *dynamic classifier selection (DCS)* method.

   (a) *Static classifier selection (SCS):* In the SCS method [Ruta and Gabrys, 2005] the region of competence for a single classifier is defined during the training period. Therefore, one classifier is chosen for the prediction task.

   (b) *Dynamic classifier selection (DCS):* Conversely, the regions of competence of participating single classifiers are defined during the prediction phase, based on the characteristics of the testing dataset [Giacinto and Roli, 2000].

Over the past decade, ensemble-based systems have attracted increasing attention, gaining popularity and spreading the application areas into a broad spectrum. Some promising areas to apply ensemble methods are biomedical [Oh et al., 2011], financial [Liao et al., 2014], remote sensing [Han et al., 2012], genomic [Shen and Chou, 2006], incremental learning [Baraldi et al., 2011], business and consumer analysis [Huong et al., 2015] and other types of rapidly growing data analysis problems.

1.4 Genetic Algorithm

Genetic Algorithm (GA) is an optimisation algorithm inspired by natural selection or the survival of the fittest. In the field of artificial intelligence, GA is called a search heuristic or

meta-heuristic. It is a subtype of an evolutionary algorithm (EA) that generates solutions to optimisation problems inspired by the methods of natural evolution (such as mutation, selection and recombination). It is considered one of the most powerful algorithms in various engineering and scientific arenas for solving complex optimisation problems.

Algorithm 1: Genetic Algorithm

```
1 INITIALISE population;
2 EVALUATE each individual ∈ population;
3 repeat
4     SELECT parents ;
5     offspring ← CROSSOVER (parents);
6     MUTATE offspring;
7     EVALUATE new population;
8     SELECT individuals for next generation ;
9 until condition_terminate ≡ satisfied;
```

A basic introduction of GA is given in [Haupt and Haupt, 2004]. The essential features shared by all GAs are given in the following [Eiben and Smith, 2012]:

Population: The GA works by creating some candidate solutions encoded in the chromosomes. These candidate solutions are called *genotypes* or *individuals* and the pool of genotypes is called the *population* of GA.

Fitness: Each individual from the population is evaluated using a fitness function that mimics the survival of fitness or goodness measure. A new population is created by the survivors having better fitness values. One complete cycle that generates a new population by breeding from an old population is called a *generation*.

Biological Operators: To mimic natural evolution better, two operations similar to biology are used. The first one is *recombination* or *crossover*, which produces a new individual by mating two individuals (sometimes called *parents*). To better represent biological evolution, a second operator called *mutation* is applied, which modifies any permitted alteration of individuals. These two operators facilitate the diversity in the population from generation to generation.

The general scheme of a GA is given in Algorithm 1. GAs work with a string-coding of variables instead of individual variables. The advantage of working with a coding of variables is that the coding discretises the search space, even though the function may be continuous. Conversely, because GAs require only function values at various discrete points, a discrete or discontinuous function can be handled with no extra burden. This

allows GAs to be applied to a wide variety of problems. Another advantage is that the GA operators exploit the similarities in string structures to make an effective search. The most striking difference between GAs and many traditional optimisation methods is that GAs work with a population of solutions instead of a single solution because there is more than one string being processed simultaneously. In GAs, good information previously found is emphasised using a reproduction operator and adaptively propagated through recombination and mutation operators. Another advantage with a population-based search algorithm is that multiple optimal solutions can be captured easily in the population, thereby reducing the effort to use the same algorithm many times. These advantages of GA have led it to be appropriate for large search optimisation.

1.5 Motivation

A comprehensive range of classification algorithms has been developed and applied successfully for ubiquitous classes of real-world domains. The goal of every learning model or classifier is to minimise the error in prediction and become a perfect model for all possible cases. If we could build such a machine learning model, there would be no need for new classification methods. In real-world situations, no models are free from errors and have their own limitations. One way to balance the limitations and performances is by using ensembles of classifiers, where a diverse range of classifiers is pooled for making final classification decisions. Mathematically, classifier ensembles yield an extra degree of freedom in these trade-offs. Intuitively, ensembles can reach a better solution than can be obtained with only a single classifier.

The primary purpose of microarray data classification is to build a classifier learned from the labelled data and then use the classifier to classify future incoming data or predict the future trend of data. Because of the advent of DNA microarray technology, a vast amount of DNA microarray data has become widely available for classification. However, as a new technology, microarrays present new statistical problems to data classification:

Curse of Dimensionality: The term 'curse of dimensionality' was introduced by Bellman [Bellman, 1961]. It refers to the complexities in managing systematic searching through a high-dimensional space. Microarray data contain an enormous number of genes with a small number of samples and this problem has prevented many existing classification systems from direct dealing with these high-dimensional databases.

Imbalanced Class Distribution: Microarray datasets contain remarkably few genes representing disease classes and a high number of genes not associated with disease

progression. Therefore, balancing the class distribution in microarrays is another problem of high-dimensional dataset analysis.

Robustness: In addition, a DNA microarray database contains a high level of noise, irrelevant and redundant data, and those data will lead to unreliable and low accuracy of analysis. Most current systems are not robust enough to handle these types of data properly.

Scalability of Classifiers: The running time complexity of classification algorithms is exponential in terms of the number of dimensions in the dataset [Hegde et al., 2007]. Often, classification algorithms require unreasonably long training times for high-dimensional datasets. This reflects the inability of the classification algorithms to scale with the increase in the number of dimensions of the feature space.

As the number of dimensions increases, the sample size needs to increase exponentially to have an efficient estimate of densities. This is referred to as the Hughes phenomenon (or the peaking phenomenon) [Hughes, 1968]. If we increase the number of features by adding more, then the classification performance improves up to a limited point, but then deteriorates. Moreover, an increase in the dimensionality makes machine learning methods computationally intractable and the learning task exceptionally difficult. For these reasons, the classification analysis of microarray dataset problems becomes more challenging.

1.6 Research Objectives

The availability of more and more high-dimensional datasets poses a unique challenge to develop new approaches, and associated machine learning classifiers to address the inherent complications of the biological datasets. In a machine learning approach, FS is an optimisation problem that involves choosing an appropriate subset of features and hence reduces the dimensionality. The EoC is a new direction of machine learning techniques that constructs an ensemble using several base classifier systems rather than selecting the best feature set. In this regard, the combination of several classifiers could be a more effective approach to achieve better performance. However, the possible ways of combining multiple classifiers create a large search space and require efficient approach to find a better combination. Keeping all of these issues in mind, this PhD research aims to deals with both the combination of classifiers and feature selection with following objectives:

Obj 1: Propose an EoC architecture based on searching for a better combination of base

classifiers. The EoC will lead to levels of accuracy that no single conventional approach can achieve.

Obj 2: Perform substantial and detailed simulation studies to profile the performance of the proposed ensemble classifiers on extremely high-dimensional datasets.

Obj 3: Investigate the grounds for the moderate dimensionality reduction performance of existing FS methods for biological datasets and incorporate those inside the classification task.

Obj 4: Finally, this PhD research will lead to an efficient framework for both the choosing a combination of classifiers and feature selection.

Meanwhile, keeping these objectives in consideration, the PhD study seeks answers to:

Q1: *If we consider the MCC an evaluation criterion for high-dimensional biological datasets classification, can an EoC provide better generalisation results than base classifiers?*

Q2: *Which combination method to use between simple majority voting and weighted voting to increase the overall classification performance?*

Q3: *Which ways of imbalance class handling (algorithmic- or preprocessing-based) best suit the EoC in biological dataset classification?*

Q4: *Could optimising multiple objectives instead of only enhancing the MCC score be beneficial for the EoC?*

If we can find the answers to these questions, then we will be able to propose an efficient and effective solution for high-dimensional biological datasets.

1.7 Organisation of the Thesis

This thesis is organised into several chapters. The current chapter describes the appropriate biological and machine learning background to the problem domain, sets the context of the research and briefly states the problem. The rest of the report is organised as follows. Chapter 2 presents the current ideas in the problem area. Chapter 3 describes the proposed solution using the simplest majority-vote-based EoC's combination search using a GA. This includes design considerations, implementation details and associated interpretations. Chapter 4 describes the weighted-vote-based EoC's combination search using a differential evolution (DE) algorithm in single-objective optimisation approach. Chapter 5 first defines the multi-objective optimisation (MOO) of an ensemble combination

search using a GA (in Section 5.2). Then, the MOO-based ensemble combination search is extended to the wrapper-based FS for elimination of the curse of dimensionality from the biological dataset (in Section 5.3). Finally, conclusions drawn from the experiments are directed towards the future direction of this research in Chapter 6.

1.8 Summary

Although it is not possible to remove the curse of dimensionality, it is possible to begin the journey on this path through a number of innovative approaches. In general, the methods used to tackle the imbalanced data problem can be divided into two categories: the algorithm-specific approach or the preprocessing of the data (under-, over-, progressive, active). Between them, the data-preprocessing-based approach is more popular and dimensionality reduction or FS techniques are well known and fairly effective in this respect. The performance of any classification technique depends on the features of the training and testing datasets. However, the task of finding the aforementioned supportive and diverse set of classifiers is not trivial. The performance of the fusion function, which carries out the combination of the decisions provided by the base classifiers, may heavily depend on such a 'good' set of classifiers. Hence, the GA will serve as the driving force for finding a better set of classifiers from a large search space to improve overall prediction accuracy.

This page intentionally left blank.

2

Literature Review

In this chapter, we focus on the classification methods aimed at achieving better accuracy using ensemble methods. The canonical concept of ensemble approaches is to improve the prediction accuracy using a set of individual classifiers. Most preferably, distinct base classifiers in an ensemble carry out different patterns embedded in the whole range of data. These combined patterns predict the final class label. We consider the literature dealing with the enhancement of classification accuracy by the adoption of ensemble methods. We will also review the relevant literature where the GA has been used to create the ensemble. This review will be followed by explaining the effectiveness of the ensemble methods applied in the area of data mining for biological datasets.

2.1 Review of Ensemble Methods

Researchers have published several reviews and empirical analyses on ensemble methods. Most of them discussed and tried to establish the reason why the EoC outperforms the single classifier counterpart. First, [Jordan and Jacobs, 1994] proposed a statistical hierarchical tree-structured mixture model of classifiers utilising ideas from the mixture model estimations and generalised linear model theory. They presented an Expectation Maximisation (EM) algorithm for adjusting the parameters of the architecture using an

iterative approach to maximise the supervised learning performance. They did not emphasise important issues (such as convergence results and consistency of the result) in the paper. [Dietterich, 2000] evaluated several ensemble methods, named Bayesian averaging, bagging and boosting. The article addressed three principal reasons behind why the ensemble methods outperform any single classifier. The author identified that when the number of available training data is small compared with the number of testing data, the classifier suffers an under-fitting problem. Conversely, classifiers that use a local search may become stuck in local optima. The author provides experimental evidence for choosing the `AdaBoost` [Freund and Schapire, 1996] algorithm for ensemble creation. However, the study did not examine the interaction between `AdaBoost` and the properties of the underlying learning algorithm. In 2000, [Breiman, 2001] proposed `Random Forest` algorithm which created an ensemble using tree-based classifiers trained on randomly distributed feature vectors. The random selection of features for tree-classifiers helped the forest to reduce generalisation error and more robust to handle noisy data. In [Valentini and Masulli, 2002], the authors present a brief overview of ensemble methods. They proposed a taxonomy based on the combination rule for the base classifiers in ensembles. A few years later, [Polikar, 2006] examined the context in which the accuracy of an ensemble method is better than their individual classifier algorithms. They also analysed the various components of an ensemble system as well as various procedures by which individual predictions could be combined. Recently, [Krawczyk et al., 2016] proposed an ensemble of fuzzy classifiers using GAs for selecting samples for the under-sampling approach. They applied the approach to select samples for balancing medical image datasets and improved the classification performances.

[Roli et al., 2001] proposed two design methods based on the so-called *overproduce and choose* paradigm. In this scheme, *overproduce* is the generation of a large set of candidate ensemble classifiers. Then 'choose' is the extraction of the best performing sub-ensemble. They have used three search algorithms named forward search, backward search and tabu search as their choose methods. Even though these design techniques demonstrated some compelling features, they do not claim any best-choice method among the three, and they do not show how to design the optimal ensemble. Similarly, [Kotsiantis et al., 2006] described various classification algorithms and the recent effort for improving classification accuracy by using ensembles of classifiers. They have discussed some algorithms based on artificial intelligence (Logic-based techniques, Perceptron-based techniques) and Statistics (Bayesian Networks, Instance-based techniques). The book by [Kuncheva, 2004] was entirely devoted to the field of model combination. The book covers the topics of multiple binary-classifier systems and their base classifier combination methods. In [Tulyakov

et al., 2008], the authors proposed an EoC method to increase the performance of pattern-recognition applications. They introduced a retraining operation that adapts the optimal ensemble classifier with another set of training data and adjusts the associated weight. They also used a local neighbourhood search to identify similar subsamples of the test set to be used for the retraining of the classifier combinations. Such effects showed a significant impact on the performance of combinations. In the same year, [Oza and Tumer, 2008] reviewed ensemble methods for diverse classes of statistical classification algorithms and their various real-world application domains. They surveyed applications of ensemble methods to problems that have historically been most representative of the difficulties in classification. In particular, the survey covers the applications of ensemble methods to remote sensing, person recognition, one-vs-all recognition and medicine. Recently, [Galar et al., 2011] developed an empirical analysis of different aggregations to combine outputs of binarisation strategies for the dataset. They performed a dual study: first, they used different base classifiers to monitor the suitability and capability of each combination within each classifier. Then, they compared the performance of these ensemble techniques with that of the classifiers themselves. The research included several well-known algorithms, such as Support Vector Machines (SVMs), Decision Trees, Instance-based Learning and Rule-based Systems. Experimental evidence suggested that the binarisation techniques with the base classifiers were the most robust methods within this framework.

These research papers showed that ensemble methods are a promising approach for use on incremental learning, data fusion, FS, learning with missing features and multiclass classification domains. They also provided answers to the question: 'Why should we choose ensemble methods instead of improving individual classification performance?'. They have justified that the EoC is an efficient method for improving classification accuracy.

2.1.1 Feature Selection with Ensemble of Classifiers

A popular method for creating a perfect classifier from a set of training data is to build several classifiers, and then to combine their predictions to achieve better accuracy. For this purpose, researchers employed different types of EoC methods. We begin this section of the literature survey with embedded FS in classifiers.

[Tsymbal et al., 2003] created an ensemble consisting of multiple classifiers constructed by randomly selecting feature subsets. They conducted experiments on a set of 21 real-world and simulated datasets. In many cases, the ensembles have higher accuracy than the single classifier model. [Namsrai et al., 2013] introduced an ensemble classification method to classify cardiac arrhythmia disease. They applied feature subset selection methods to obtain a number of feature subsets from the original dataset. Then, they built classification

models using each feature subset and combined base classifiers decisions using a weighted-voting approach. They considered both classification error rate and FS rate (which means the frequency of a specific feature in feature subsets) to calculate a score for each classifier in the ensemble. The ensemble method improves the classification accuracy significantly. In contrast, [Srimani and Koti, 2013] conducted a classification analysis on five medical datasets, and their results showed that individual classifiers performed better in some cases than an ensemble. They concluded that only selected classifiers need to be used for each dataset and that to achieve optimal accuracy with a medical dataset, researchers need to select the classifier carefully. They selected default options for classifiers and did not use any optimisation techniques to find the optimal solution for ensemble classifiers. This was a missing component in their conducted research.

There exist some research results on FS with EoCs using tree-based structures. In this regard [Zhang et al., 2010] proposed an ensemble classification system based on diversified multiple trees that addressed the uncertainty of the microarray data quality problem. The proposed diversified multiple decision tree algorithms (DMDT) are able to determine most informative features from abundant features. Then, they used this unique diversity value as an ensemble combination function. The test results showed that the DMDT is more accurate on average than other well-known EoC methods on microarray datasets. Likewise, [Hu et al., 2008] proposed a maximally diversified multiple decision tree (MDMT) algorithm based on an ensemble method for robust microarray classification. They were concerned with the noise susceptibility of different decision tree algorithms and MDMT for microarray dataset classification. The experimental results showed that ensemble decision tree methods tolerate the noise values better than single tree methods do. They recommended decision-tree-based ensemble classifiers to deal with noisy microarray datasets. Similarly, [Osareh and Shadgar, 2013] proposed an ensemble method combining Rotation Forest and `AdaBoost` techniques that preserved the accuracy and diversity in microarray datasets. They applied five different FS algorithms to determine a concise subset of informative genes. Then, they applied a decision-tree-based ensemble classifier using selected features as a training set. The experiment showed that the proposed ensemble classifiers outperformed not only the conventional machine learning classifiers but also the classifiers generated by two widely used ensemble learning methods, that is, `Bagging` and `Boosting`. Previously, [Galar et al., 2012] tackled the classifier learning problem with datasets that suffer from imbalanced class distributions and provided empirical comparisons based on ensemble taxonomy. They reviewed the state of the art on ensemble techniques in the framework of imbalanced binary datasets. Their results show empirically that ensemble-based algorithms are advantageous because they outperform the use of mere preprocessing

techniques before learning the classifiers. [Koutanaei et al., 2015] proposed hybrid ensembles for FS and classification for credit scoring datasets. They considered three FS methods and evaluated their performance from the classification accuracy achieved by the SVM classifier. Then, they created an EoC using neural networks. The authors found the best classification accuracy for the ensembles trained with the feature subset selected using the principal components analysis approach.

These works advocate that EoCs are more effective methods than other combinations of classifiers like `Boosting` and `Bagging`. It is also possible to obtain the inherent power from embedded FS methods for imbalanced data classification, noise susceptibility and multiclass classifiers. In a few words, EoC methods are suitable enough to deal with high-dimensional biological datasets.

2.2 Ensemble of Classifiers using Genetic Algorithm

From the literature, we can see that in the EoC method, the GA has mainly been applied at three levels. The application level of GA in the EoC can be categorised as: decision combination level, FS level and classifier formation or creation level. First, we will discuss the literature in which GA has been employed in the base classifier creation. Next, we will discuss studies that use GAs for the FS and combination methods inside other major categories of EoC methods.

[Zhang and Bhattacharyya, 2004] demonstrated the potential of genetic programming (GP) as a base classifier algorithm in building ensembles in the context of large-scale data classification. An ensemble built upon base classifiers trained with GP significantly outperformed its counterparts built upon individual base classifiers. [Wang and Wang, 2006] proposed an EoC where each individual classifier is trained on a particular weighting over the training samples. They incorporated a GA to search a substantial weighting space. They tested the algorithm on the University of California, Irvine (UCI) benchmark datasets and discovered that the ensemble method was robust and consistent for face-detection applications. [Ranawana and Palade, 2006] presented a current overview of Multiclassifier Systems (MCSs) and provided an outline roadmap for MCS. They also presented a case study of the MCS theoretical issues, and simple guidelines for the selection of different paradigms. Moreover, they introduced a novel optimisation of the traditional majority-voting combination method that uses a GA. Subsequently, [Kim and Oh, 2008] proposed a hybrid GA for classifier ensemble selection. They used two local search operations to improve offspring prior to replacement. They parameterised the local

search operations to control the computation time and found it to be an effective approach. Later, [Espejo et al., 2010] surveyed existing literature on the importance of GP for classification. They showed the different ways in which an EA can help in the creation of accurate and reliable classifiers.

Taking into consideration the variations of base classifiers used, the EoC can be categorised into *homogeneous* and *heterogeneous* methods. The FS ensemble can be embedded into both of them as the diversity generation methods. Now, we will review literature falling into those two categories that use GA.

2.2.1 Homogeneous Ensembles

[Kim et al., 2006] proposed a meta-evolutionary (ME)-based homogeneous EoC method. It uses a two-level evolutionary search through the FS and ensemble creation. The authors applied a ME for FS using an ensemble method that works like the `Boosting` algorithm. The fitness of the ensemble is updated with the cost based on its predictive accuracy, as determined by majority voting with equal weight among base classifiers. They used a variable number of ANNs to form the ensemble and used majority voting as their combination function. Experimental results on 15 datasets suggested that the weighted ensemble is more effective than single classifiers and classic ensemble methods. They tested their method on smaller datasets (the maximum dimensions in the dataset were 3772 features, 27 samples); the suitability of their approach needs to be tested to handle large datasets.

Later, [Cleofas et al., 2009] proposed a GA-based ensemble for the Imbalance Class problem. They applied the GA on the FS level for balancing the class distribution by under-sampling the majority class samples. They used an m-dimensional chromosome, which represents all features in the training set of the m samples. Then, they applied features selected by GA to the EoC. This method is impractical to apply on high-dimensional datasets due to encoding all features into the chromosome.

Next, [Gaber and Bader-El-Den, 2012] proposed a GA-based random forest (GARF) ensemble method. They used a heuristic chromosome (HC) representation in which an individual represents an ensemble (`Random Forest`) and each gene represents a Random Tree. They stored precomputed classification results for all random trees. Then, they evolved a new ensemble using pregenerated trees (EV-Ensemble) to evaluate the fitness of the new random forest using the HC. They also applied an Indirect GA to create a random forest classifier. Their experiment showed that the performance of a random forest could be boosted when using GA.

More recently, [Oh and Gray, 2013] proposed a GA Robust Ensemble (GA-RE) that used the GA in the classifier formation level. It used variable sized individuals in the

range of 5 to 20 decision stump trees. But the stump tree they used contained only two terminal nodes, which possibly loses informative features. Instead of using decision stumps, they could try using decision trees. In 2015, [Zhang et al., 2015] proposed a homogeneous EoC selected using the GA approach for image data classification. They selected the homogeneous base classifiers created by varying the parameters using the GA. They experimented with SVM- and NN-based homogeneous EoC approaches.

Table 2.1 shows features of the GAs used in these homogeneous ensemble methods.

2.2.2 Heterogeneous Ensembles

Heterogeneous ensemble methods use different types of classifiers to create an ensemble combination. Some researchers use a single instance from each of the different classifiers and others use multiple instances of some classifier algorithms. GAs have been applied to FS, classifier selection and decision fusion level of the heterogeneous ensemble. In [Gabrys and Ruta, 2006], they proposed an interesting individual structure of GA applied to an ensemble method. They represented each individual using a three-dimensional incidence matrix. In the incidence matrix, one dimension represents features, another dimension represents classifiers and the third dimension represents combination methods. They also proposed associative genetic operators to work with the individual structure. They applied this method only on a very small dataset (having only six features). However, the structure is not suitable for high-dimensional datasets because it encodes all features in each individual. Next, [Xu and He, 2008] proposed a GA-based ensemble classifier. The GA was applied on both FS and decision combiner levels simultaneously. They applied it to a real-world multi-sensor dataset. They randomly selected features for each base classifier and used a very small number of features to build the classifier. There is scope for enhancing this system by adopting an appropriate FS method. Although the method has outperformed feature-level and decision-level fusion methods, it needs to be tested on larger datasets. Recently, [Thammasiri and Meesad, 2012] proposed a GA-based classifier ensemble method to select the appropriate classifiers. They used majority vote to increase the ensemble classification accuracy. Their evaluation showed that the proposed ensemble classification models selected by the GA yield highest performance and are efficient in building an ensemble. The maximum feature count for the applied datasets was 30. Later, [Lertampaiporn et al., 2013] applied a heterogeneous ensemble method using GA for the FS level. They used `Support Vector Machine` (SVM), `k-Nearest Neighbour` (k-NN) and `Random Forest` (RF) to create the ensemble and applied them to a large biological dataset. They applied `SMOTE`-bagging methods to balance the dataset and used a correlation-based feature selection (CFS) method with a GA search method. Their

Algorithm	GA Application Level	Representation	GA Operators	Ensemble Formation
MEE (Meta-Evolutionary Ensembles) [Kim et al., 2006]	1) Through the classifier space of ensembles, 2) Feature Subset selection	$len = D + log_2(G)$; D = features, $log_2(G)$ = binary representation of ensemble index and G = maximum size of the ensemble	Fitness: Weighted fitness. Selection: Offspring receives weight based on threshold. Mutation: randomly mutate $1 - bit$ at $p = 0.5$ rate.	Variable number of ANN (Artificial Neural Network) trained on different feature subspace. Performance Measure: average and maximum accuracy. Combination: Majority vote
GA-Ensemble for Imbalance Class [Cleofas et al., 2009]	FS level (GA applied to majority class undersample to balance the class distribution)	m-dimensional chromosome represents all of the training sets of the m samples	Fitness: Geometric Mean; Population Size: 15; Term. Cond: ($generation = 30$); Selection: n elite; Recombination: Uniform; Mutation: bit flip	ANN
GARF (GA-based Random Forest) [Gaber and Bader-El-Den, 2012]	EV-Ensemble (Evolving new Ensemble from Existing) Random Forest. IGA (Indirect GA): RF classifier creation Level	HAC (Heuristic Chromosome): Individual=Ensemble (Forest), Gene=A Random Tree	Fitness: Geometric Mean (GM), Pop Size: 15, Term Cond: ($generation = 30$), Selection: n elite, Recombination: Uniform, Mutation: Alter bits at $P = 0.10$ rate	Store all precomputed classification results for all trees in the buffer. Majority vote by Random Trees.
GA-RE (GA Robust Ensemble) [Oh and Gray, 2013]	Ensemble Classifier Formation Level	Variable Size (5–20): a set of weak learners (DS) and their weights Quad [weight, Split var, Split val, Prediction]	Fitness: error rate on test set, Pop Size: 20, Term Cond: ($gen = 2000$), Selection: Elitist saved, Recombination: Single Point Recombination	Initially 5 Decision Stump, Max allowed 20

TABLE 2.1: Homogeneous ensemble of classifier algorithms using genetic algorithm

Algorithm	GA Application Level	Representation	GA Operators	Base Classifiers
MCSS-3D (3D-Multiclassifier Selection system) [Gabrys and Ruta, 2006]	MGA (Multidimensional GA) applied on Ensemble Classifier Level	Three-dimensional binary incidence cube (Feature x Classifiers x Combination method)	Fitness: minimise the weighted misclassification rate; Population Size: 10; Selection: n elite. Mutation: bit flip on all dimensions; Combination: mean, minimum rule, maximum rule, product, Majority Vote	klclc (linear with KL expansion), Loglc (logistic linear classifier), Ldc (linear discriminant classifier), Qdc (Quadratic with normal density), Pfsvc (Pseudo-Fisher SVM classifier), Lmnc (Levenberg–Marquardt neural net)
GACEM (GA-Based Classifier Ensemble) [Xu and He, 2008]	Both FS Level and Decision Combiner Level	$(MN + N)$ bit binary, where $M = Dataset$ and $N = Classifier$	Selection: Roulette wheel; Population Size: 30; Performance Measure: Accuracy; Combination: Plurality Voting	Ldc, Quadratic Discriminant Classifier (QDC), k-Nearest Neighbour (k-NN), Classification And Regression Trees (CART)
DNSEGA (Decision tree, Neural net, SVM, Ensemble GA) [Thammasiri and Meesad, 2012]	Classifier Selection Level	binary string	Selection: Roulette wheel with elitism; Population Size: 30; Recombination: Multipoint; Mutation: bit flip; Performance Measure: Mean Absolute Percentage Error (MAPE); Combination: Majority voting	Total 30 classifiers (10 classifiers from each of Decision Tree, ANN and SVM)
HE (Heterogeneous Ensemble) [Lertampaiporn et al., 2013]	FS with CFS technique		Fitness: Merit score of CFS; Population Size: 200; Recombination: Single point; Mutation: bit flip; Combination: Probability Threshold Voting	Support Vector Machine (SVM), k-Nearest Neighbour (k-NN), Random Forest (RF), Naive Bayes (NB), MPL-NN, Decision Tree (J48), JRIP, RBF Networks

Table 2.2: Heterogeneous ensemble of classifier algorithms using genetic algorithm

Algorithm	**Dataset Dimensions ($n \times m$)**	**Dataset Separation**	**Training Data Split**
MCSS-3D (3D-Multiclassifier Selection system) by [Gabrys and Ruta, 2006]	Iris (150 × 4), Liver Dataset: (345 × 6)	Dataset split into Train and Test Sets	2-fold CV
GARF (GA-based Random Forest) by [Gaber and Bader-El-Den, 2012]	Letter Recognition (20000 × 16), vehicle (946 × 18)	Dataset split into Training (Build Random Trees), Validation (EV-training: Individual Model Build) and Testing (evaluate the GARF)	Use full training data to train a tree (gene)
GACEM (GA-based Classifier Ensemble) by [Xu and He, 2008]	Sound Data (665 × 25)	For each 19 Data point split into: four (04) for training, five (05) for validation in the fitness function and ten (10) for testing the Generalisation	Use full training data to train each classifier, use FS by GA and use them
DNSEGA (Decision tree, Neural net, SVM, Ensemble GA) by [Thammasiri and Meesad, 2012]	GermanCredit (1000 × 24), Bankruptcy (240 × 30)	Dataset split into Training (Build Classifiers Model), Validation (test the Ensemble) and Testing (evaluate the DNSEGA)	Bootstrap (randomly split the training data equal to the number of classifiers, N)
MEE (Meta-Evolutionary Ensembles) by [Kim et al., 2006]	Sick (3772×27), Hypo (3772×27), Sonar (208 × 60)	Dataset separated into training and testing sets using the holdout method	Use full training split
GA-RE (GA Robust Ensemble) by [Oh and Gray, 2013]	Kyphosis (81 × 3), WDBC (569 × 30), Glaucoma (196 × 62)	10-fold CV	10 runs of 10-fold CV
GA-Ensemble for Imbalance Class	Satimage (6435 × 36), Phoneme (7797 × 617)	Dataset separated using 80:20 holdout method (training (80% data) and testing (20% data)) sets	5-fold CV
HE (Heterogeneous Ensemble) by [Lertampaiporn et al., 2013]	microRNA precursor (pre-miRNA): Virus (23 × 240), Plant (52 × 4014), Animal (93 × 13972)	Train dataset separated using `SMOTE`bagging and used CV to evaluate the ensemble	5-fold CV

TABLE 2.3: The maximum feature dimension of the datasets used by different ensemble of classifier methods using genetic algorithm by ascending order.

experiment shows better performance than for single classifiers.

Table 2.2 shows features of the GAs used by these homogeneous ensemble methods. Different articles discussed here use different ways to handle datasets. Some of them used a cross-validation method on full datasets; some studies separated the dataset into training and testing datasets. Then, they applied different cross-validation or used full training datasets to evaluate the performances of the ensemble. We have tabulated all of those studies in ascending order of the datasets based on the maximum feature dimensions in Table 2.3.

Besides genetic algorithms, the 'hyper-heuristics' have emerged as a promising method for solving hard computational problems [Burke et al., 2003]. The hyper-heuristics algorithms solve a computational problem by separating it into several sub-problems and searching over the space by forming a set of low-level heuristics. A hyper-heuristic method either selects from a set of low-level heuristics or generates new heuristics from the existing ones to solve a problem. This approach has been used for automating the design and tuning of heuristic methods to solve hard computational search problems [Burke et al., 2007, Burke et al., 2009, Hart and Sim, 2016]. Besides those problems, hyper-heuristics have been applied in machine learning [Sim et al., 2012, Özcan et al., 2012, Pappa et al., 2014, Montazeri et al., 2015, Basgalupp et al., 2015]. Hyper-heuristics have been also applied as a search method for the homogeneous ensemble creation where divide and conquer approach is suitable to solve the problem [Basgalupp et al., 2015]. In tree-based ensemble and reinforcement learning, the hyper-heuristic has also found to be relevant [Özcan et al., 2012, Sim et al., 2012]. Because of the nature of hyper-heuristics, it is suitable for homogeneous ensemble creation where a classification problem is possibly divided and solve using homogeneous classifiers and the final solution could be found after the aggregation of those solutions. The hyper-heuristics will be applicable for classifier selection ensembles where different base classifiers are trained on different parts of the classification problem and the final result is decided by a selected trained classifier. In contrast, the heterogeneous ensemble of classifiers formed using base classifiers trained on full training data and their output is combined to find the final decision. Hence, the genetic algorithms tend to be more appropriate to explore the search space of a heterogeneous ensemble of classifiers.

2.3 Challenges in Ensemble of Classifiers

Knowledge discovery from imbalanced class datasets is a challenging topic in data mining. This problem occurs when the number of cases that represent one class is much lower than the number on other classes. This common situation, present in real-world applications,

has served to stimulate the growth of research.

[Kuncheva and Whitaker, 2003] studied 10 statistics to determine diversity among binary-classifier outputs. They examined the relationship between the ensemble accuracy and measures of diversity, and among the measures themselves. Their results disagreed with some accepted relationships between accuracy and diversity measures in building classifier ensembles in real-life pattern-recognition problems. [Brown et al., 2005] first reviewed the various attempts to provide a formal interpretation of error diversity, including several heuristic and qualitative explanations in the literature. They surveyed the various techniques used for creating various ensembles, and categorised them, forming an exploratory taxonomy of diversity creation methods. They introduced the concept of implicit and explicit diversity generation methods and proposed some new directions that may prove useful in understanding classification error diversity.

The EoC process can handle the issue of imbalanced data. We will examine existing works dealing with these concerns. [Li, 2007] proposed a bagging classification ensemble system for classifying imbalanced data. They utilised all of the minority class data without creating synthetic data or making changes to the existing classification systems. The experimental results using real-world imbalanced data advocated the efficacy of the system. Later, [Sun et al., 2009] reviewed existing methods for classification of data with imbalanced class distribution. They gave an overview of the classification of imbalanced data regarding the application domains and the nature of the problem. They also addressed the learning difficulties with conventional classifier learning algorithms and finally the class imbalance problem in the presence of multiple classes. The authors pointed out that the learning of the `AdaBoost` algorithm usually biased to the majority class because of the boosting weights being adjusted using accuracy. For this reason, diversity is particularly crucial for ensembles used on class-imbalanced data.

Another challenge for creating an EoC is the combination method. It plays a significant role in the process of ensemble classifiers' final decision and classification accuracy. In this regard, [Jain et al., 2000] explored the principles of several classifier combination methods. They also mentioned different types of combination processes, like feature sets, training sets, classification methods or different training sessions joined together. The final decision is combined by a number of base classifiers to improve overall classification accuracy. The nearest-mean method for classifier combination gave the best overall result. This was also the best result of the entire experiment. [Kleinberg, 2000] bridged the gap between the theoretical prediction expressed by stochastic discernment and practical explanation of algorithmic implementation. He also tested and found that stochastic discrimination outperformed both boosting and bagging in the majority of benchmark problems.

In machine learning, many researchers have used the EoC to improve the accuracy of individual classifiers by mixing several of them. Neither of these learning methods alone solves the class imbalance problem. The ensemble algorithms need to be specially designed to deal with these problems.

2.4 Application of Ensembles in Biological Datasets

Recently, the use of ensemble methods for biological datasets has increased dramatically. Numerous works dealing with ensemble methods have been published since 2012. [Krawczyk et al., 2013] evaluated seven state-of-the-art ensemble approaches in an extensive set of experiments on five benchmark datasets. They investigated the strengths and weaknesses of the Ensemble classification algorithms. [Lin et al., 2013] applied ensemble and hierarchical structure to examine the prediction of protein fold patterns. They combined a number of base classifiers to make the ensemble and distribute the dataset into several subdatasets. Their experimental results achieved 74.21% accuracy, which is much higher than results obtained with the conventional approach (varies from 45.6% to 70.5%). [Woźniak et al., 2013] presented an up-to-date survey on MCSs from the point of view of Hybrid Intelligent Systems. The article discusses significant issues, such as diversity and decision fusion methods, providing a vision of the range of recently developed applications. Let us examine further applications of the ensemble method in a biological dataset.

2.4.1 Microarrays

Several works utilise the information abundance of GEM focused on the correct classification or recognition of biological types in cancer research. In this regard, [Li et al., 2004] have developed an innovative ensemble decision system to implement multiple gene mining tasks efficiently. They have applied this approach to analyse two publicly available datasets (colon and Leukemia). The globally optimal gene subsets identified by their approach produced the highest accuracy for classification of colon cancer tissue types. [Hong and Cho, 2006] also utilised ensemble approaches to classify imbalanced microarray data. They proposed a new ensemble approach with GP that can increase the successful classification of these datasets. They applied the method to common gene expression datasets for ovarian, prostate and lung cancer and found that GP is able to improve the performance of the ensemble classifier in imbalanced datasets. Further, many studies design accurate and interpretable fuzzy systems using EAs under the name of genetic fuzzy systems. [Peng, 2006] proposed an ensemble method to eliminate the inadequacy of standard

machine learning techniques and achieved accurate and robust microarray data classifications. They made a pool of candidate base classifiers by gene subsampling and form the classification ensemble based on classifier clustering. Subsets of base classifiers are combined to form the ensemble. The experimental results of the proposed classifier method outperformed not only the typical machine learning method but also two widely used ensemble learning methods (`Bagging` and `Boosting`). Later, [Shah and Kusiak, 2007] proposed an integrated gene-search algorithm for microarray dataset analysis that involves a GA and correlation-based heuristics for data preprocessing and data mining. They used the same dataset used by [Hong and Cho, 2006]. The ensemble created with a decision tree and an SVM algorithm provides high classification accuracy with the ability to identify the most significant genes. They applied bagging and stacking algorithms to improve the classification accuracy. Further, [Kim and Cho, 2008] proposed an EA to construct advanced ensembles of features and classifiers to achieve high classification performance in the analysis of microarray data. Regardless of the exponential number of possible ensembles of individual feature–classifier pairs, the proposed method was able to find a perfect ensemble in a reasonable amount of time that was superior to individual classifiers. [Yang et al., 2010a] provide an overview of the most widely used ensemble learning methods and their application in various bioinformatics problems, including the main topics of gene expression, mass spectrometry-based proteomics, gene–gene interaction identification from genome-wide association studies and prediction of regulatory elements from DNA and protein sequences. Further, they identified and summarised the future trends of ensemble methods in bioinformatics. They also discussed some promising methods for bioinformatics, such as an ensemble of SVMs, meta-ensembles and ensemble-based FS.

2.4.2 Protein Folding

Protein folding is the process to assume the functional shape of a protein from the structure. The classification of protein structure represents an important process in molecular biology. It deals with the understanding the associations between sequence and structure as well as possible functional and evolutionary relationships. The computational study of 'protein folding' analyse the aspects related to the prediction of protein stability, kinetics, and structure. Recent improvements in structural genomics and high-throughput experiments have increased the dimensions in the biological databases at a rapid pace. The analysis and inspection of vast amount of structural data with traditional methods are impossible. Machine learning has been widely applied to bioinformatics and has gained a lot of success in this research area.

Recently, researchers have been devoting their efforts to data mining and knowledge

discovery from protein databanks. Protein data analysis employed fuzzy logic alongside the manipulation of GAs and ensemble methods. [Ishibuchi, 2007] proposed evolutionary multi-objective algorithms for interpretability–accuracy trade-off analysis of fuzzy systems. Afterwards, [Ding and Zhang, 2008] proposed a powerful method for mining protein databanks. They created an EoC where the base classifier was the fuzzy k-NN (FKNN). Moreover, they used an immune GA to search the optimal weight factors. The results obtained using the Jackknife test indicated the potential usefulness of the proposed method for protein function. [Shen and Chou, 2006] proposed an ensemble classifier with weighted voting to predict the protein folding patterns, which is much more complicated and difficult. Optimised k-NN rules were achieved using a number of basic classifiers, each trained in different parameter systems. Experimental results showed higher prediction accuracy than the relative rates obtained by different existing ANNs and SVM approaches. Similarly, [Shen and Chou, 2007] proposed an EoC-based prediction method for classifying membrane protein type. This is both a crucial and challenging topic in modern molecular and cellular biology. They formed a voting-based 'ensemble classifier' by fusing a set of k-nearest neighbour (k-NN) classifiers. [Krawczyk et al., 2016] proposed an ensemble of sample selection for class-imbalanced learning in microscopic images. They used an EA-based ensemble of FS to balance the class distribution of the dataset.

Experimental results demonstrated that the ideas of EoC can also be used to improve the prediction quality in classifying biological datasets.

2.5 Summary

In this chapter, we have reviewed significant advances in ensemble-based classification systems. We reviewed the application of GAs and other search-based approaches (including the hyper-heuristics) for finding the ensemble combinations. We found both approaches have their strengths and weakness in searching the large search space. However, recently proposed hyper-heuristics found to be more suitable for searching homogeneous ensemble of classifiers. Because, the low-level heuristics could possibly solve the part of the classification problem and aggregating them will give the final estimation of the problem. It is not suitable for applying in a heterogeneous ensemble of classifiers where each classifier learns from the full training dataset and aggregation will not benefit from the hyper-heuristics. In this case, the evolutionary search is more suitable. It can explore through the wide search space for finding the best combination of base classifiers to create the ensemble. This will facilitate the Obj 1 (in Section 1.6) of our research.

In literature, we found many applications of ensembles for high-dimensional and class-imbalanced data classifications [Li, 2007, Cleofas et al., 2009, Sun et al., 2005]. Among those methods, [Li, 2007] showed that using all minority samples, instead of SOMTE like synthetic oversampling technique, performed better for real-world data. Their experimental outcome inspired the development of our class imbalanced data handling approach for high-dimensional data. This will help us to achieve the goal in Obj 2 (in Section 1.6). We went over several works dealt with feature selection inside the ensemble of classifier [Breiman, 2001, Namsrai et al., 2013, Koutanaei et al., 2015]. Among those works, [Koutanaei et al., 2015] used multiple feature selection methods inside the ensemble of classifiers. [Namsrai et al., 2013] proposed an ensemble of classifier trained using feature subset selected by FS method. The outcome of those methods encouraged us to adopt this idea of selecting a feature subset inside the ensemble to satisfy the Obj 3 (in Section 1.6) of the research. However, unlike to these works, we will not use all of the base classifiers to form the ensemble. We will use genetic algorithm search for selecting best subset of features selected by participating FS methods to train selected base classifiers to formulate the ensemble. The literature study about the ensemble of classifier methods for biological data classification revealed it as one of the successful and efficient approaches for the real-world problems. It showed many applications in various types of biological data classification (microarrays, proteomics) [Lin et al., 2013, Kim and Cho, 2008]. Many other works have been discussed in the section 2.4. Those successful applications prove that the ensemble of classifiers is a suitable approach for dealing with real-world biological data classification. This will help towards the achievement of the Obj 4 (in Section 1.6) of the research work.

We studied the current techniques on ensemble learning and their advantages and disadvantages. We also examined the current literature on the analysis and application of diversity in classifier ensembles and GAs. This reviewed literature implies that the ensemble classifier is extremely promising and might become a convenient method in genomics, proteomics and bioinformatics. Moreover, we can conclude that the EoC could be a better solution to achieve more accurate classification. A diverse number of base classifiers can present a large search space for possible classifier combinations. GA is a potential solution for searching this sample space within the EoC to find better solutions.

This chapter contains parts of the paper by Haque MN, Noman N, Berretta R, Moscato P (2016) "Heterogeneous Ensemble Combination Search Using Genetic Algorithm for Class Imbalanced Data Classification". PLOS ONE 11(1): e0146116. doi: http://dx.doi.org/10.1371/journal.pone.0146116.

3

Genetic Algorithms in Majority Voting Ensemble of Classifiers

EAs deal with highly complex optimisation and search problems by a simulation of natural selection. They are able to solve high-dimensional and extremely complex problems that require adaptive behaviour, systematic exploration of alternatives and problems that are difficult to formulate with conventional analytical techniques. This nature of EAs has turned them into an inevitable component in efficient and intelligent systems design. There are many types of EAs dealing with single-objective optimisation problems. Genetic algorithm (GA), typically, a single-objective algorithm provides a strong performance for particular types of problems. To enhance the diversity of the problem domain when applying classification, an ensemble method is an efficient method that combines multiple classification algorithms together. The proposed method will utilise the strengths of an EoC and apply the GA-based search to obtain a better combination for creating the ensemble.

3.1 Ensemble of Classifiers

An ensemble method for classification aggregates the prediction of multiple classifiers to improve the classification accuracy. It is also referred to as a *classifier combination* method. Every base classifier in the ensemble learns from a training dataset. The base classifiers apply this training to predict the membership value of instances in a given testing dataset. The EoC method combines those votes of base classifiers to decide the membership of instances. A logical view of ensemble creation is shown in Figure 3.1. Here, we have k individual classifiers $\{\mathbb{C}_1, \cdots, \mathbb{C}_k\}$. We train each classifier with the same training dataset. Then, the final learning outcome is produced by the combination of classifiers ($\mathbb{C}^*$).

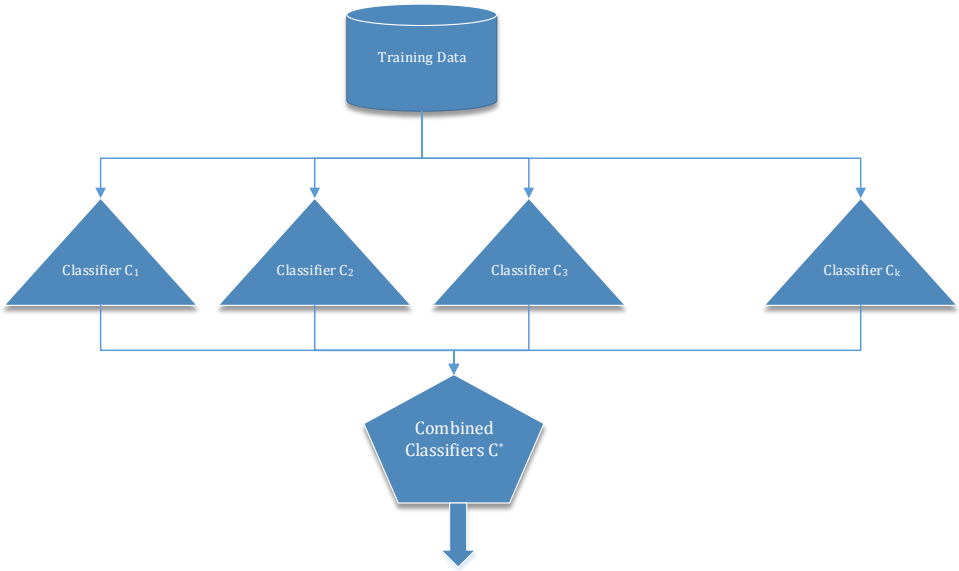


FIGURE 3.1: Logical view of the training process for base classifiers in the ensemble.

3.1.1 Rationale for the Ensemble Method

Improvements in classification tasks are often obtained by aggregating a group of classifiers (referred to as base classifiers). Next, we will explain in brief the reasons why ensemble methods tend to perform better than any single classifier. Then, the discussion will proceed to the proposed architecture of the GA for ensemble creation. We need to justify how an ensemble method using the **majority-voting** approach can improve the prediction accuracy. By using the Bernoulli trial formula [Kobayashi et al., 2011], we can obtain the

combined efficiency of the ensemble for the binary-classification problem. To obtain the probability (P) of observing r successes in n trials, we have

$$P(X = r) = \binom{n}{r} p^r (1-p)^{(n-r)}, \tag{3.1}$$

where, n = number of events, r = number of successful events $\left(r \geq \frac{n}{2}\right)$, p = probability of success on a single trial, $\binom{n}{r} = \frac{n!}{r! \times (n-r)!}$ and $1-p$ = probability of failure.

If the ensemble cardinality, $|\mathbb{E}| = k$, denotes the number of independent base classifiers used for creating the ensemble, then it would make a wrong prediction only when more than half of the base classifiers predict incorrectly. In this case, the probability of success $P(\mathbb{E})$, in the EoC can be derived from Equation (3.1) as

$$P(\mathbb{E}) = \sum_{j}^{k} \binom{k}{j} p^j (1-p)^{(k-j)}, \tag{3.2}$$

where $j = 1 - w$ is the number of base classifiers giving a correct prediction in majority voting ensemble approach.

Let us consider cardinality of ensemble $|\mathbb{E}|$ is twenty ($k = 20$), where each base classifiers having individual prediction accuracy $p = 0.6$ (many classifiers perform almost like random classifier or sometime even worse for real-world datasets) for creating the ensemble and $j = 11$ as the number of successful events (where $j \geq \frac{k}{2}$). Now we can calculate the probability to classify correctly for this ensemble method using the Equation (3.2) as

$$\begin{aligned} P(\mathbb{E}) &= \sum_{11}^{20} \binom{20}{11} * 0.6^{11} * (1-0.6)^{(20-11)} \\ &= \sum_{11}^{20} \binom{20}{11} * 0.6^{11} * 0.4^{(20-11)} \\ &= 0.9434736 \\ &\approx 94\%. \end{aligned}$$

From this calculation, we draw the conclusion that, if we create an EoC having 60% individual accuracy and combine them using the majority ($j \geq \frac{k}{2}$) voting approach, there is approximately 90% probability for the ensemble to predict accurately. We will use this **majority-voting** approach to combine the prediction of single classifiers and transform it into the final decision.

3.1.2 Majority-Voting Ensemble of Classifier

Now, we define the majority-voting (unweighted) ensemble ($\mathbb{E}_{mv}$) for a binary class ($\Omega = 0, 1$) problem. Let k be the number of individual base classifiers in the ensemble $\mathbb{E}_{mv} = \langle \mathbb{C}_1, \cdots, \mathbb{C}_k \rangle$ trained on the same training dataset $\mathbb{D}$. The unweighted majority-voting ensemble classifier outcome for each sample ($u_j \in \mathbb{U}$) is defined as:

$$\mathbb{E}_{mv}(u_j) = \begin{cases} 1 & \sum_{i=1}^{k} \mathbb{C}_i(u_j) > \frac{k}{2} \\ 0 & \sum_{i=1}^{k} \mathbb{C}_i(u_j) < \frac{k}{2} \\ \text{Random } \{1, 0\} & \text{Otherwise.} \end{cases} \tag{3.3}$$

The end result of the EoC is produced by the unweighted majority voting by all k trained base classifiers.

3.2 The Genetic Algorithm-Based Ensemble of Classifiers

A *GA* is an optimisation algorithm inspired by natural selection, or 'survival of the fittest'. The main difference between GAs and many traditional optimisation methods is that GAs work with a population of solutions rather than a single solution. The implementation of a GA can be parallelised easily because of this multipoint searching characteristic. Moreover, a GA is less susceptible to becoming stuck in local optima when compared with many other heuristics. The recombination and mutation operations help the heuristic to escape from local optimal solutions by producing significant randomness in the population. These advantages make GAs appropriate for large and complex optimisation problems.

In our design, we chose 20 base classifiers (listed in Table 3.1) from the Waikato Environment for Knowledge Analysis (WEKA) data mining software suite, version 3.6, [Hall et al., 2009], to create the ensemble combinations. Our chosen 20 classifiers can produce over a million unique ways to create ensembles. Because it is impractical to perform an exhaustive search through this huge set of combinations to find the best ensemble, we have implemented a GA to perform a heuristic search. The working process of the whole algorithm is divided into two phases. The first step creates cross-validation folds and models on training folds to be used by the next phase (Figure 3.2). Then, the GA is used to search for the best EoC from all possible ensemble combinations. This GA-based EoC searching method will be denoted as *GA-EoC* (Figure 3.3). The working phases of GA-EoC are:

Category	Count	Classifier	Brief Description
Bayes Network	3	`BayesNet`	Bayes Network learning algorithm
		`NaiveBayes`	Naive Bayes classifier using estimator [John and Langley, 1995]
		`NaiveBayesUpdateable`	This Navie Bayes can start building with zero training instances
Function	5	`LibSVM`	Implementation of Support Vector Machine [Chang and Lin, 2011]
		`Logistic`	Logistic regression with a ridge estimator. [Le Cessie and Van Houwelingen, 1992]
		`SGD`	Stochastic gradient descent for learning linear models
		`SimpleLogistic`	LogitBoost, the speed-up logistic regression [Sumner et al., 2005a]
		`VotedPerceptron`	Implementation of the voted perceptron algorithm [Freund and Schapire, 1999]
K-NN	1	`IBk`	K-nearest neighbours classifier [Aha et al., 1991]
Rule Learner	7	`DecisionTable`	Simple decision table majority classifier [Kohavi, 1995]
		`JRip`	Propositional rule learner [Cohen, 1995]
		`OneR`	1-R Classifier [Holte, 1993]
		`PART`	PART decision list [Frank and Witten, 1998]
		`RandomTree`	Tree classifier considers K randomly chosen attributes at each node
		`REPTree`	Fast decision tree learner
		`ZeroR`	0-R classifier (Predicts the mean for a numeric or the mode for a nominal class)
Tree	4	`DecisionStump`	Class for building and using a decision stump
		`J48`	Class for generating a pruned or unpruned C4.5 decision tree [Quinlan, 2014]
		`RandomForest`	Class for constructing a forest of random trees [Breiman, 2001]
		`LMT`	Classifier for building 'logistic model trees' [Landwehr et al., 2005a]

Table 3.1: List of 20 base classifiers used in genetic algorithm-based ensemble of classifiers (GA-EoC). Classifiers without references in brief description are available in WEKA [Hall et al., 2009]

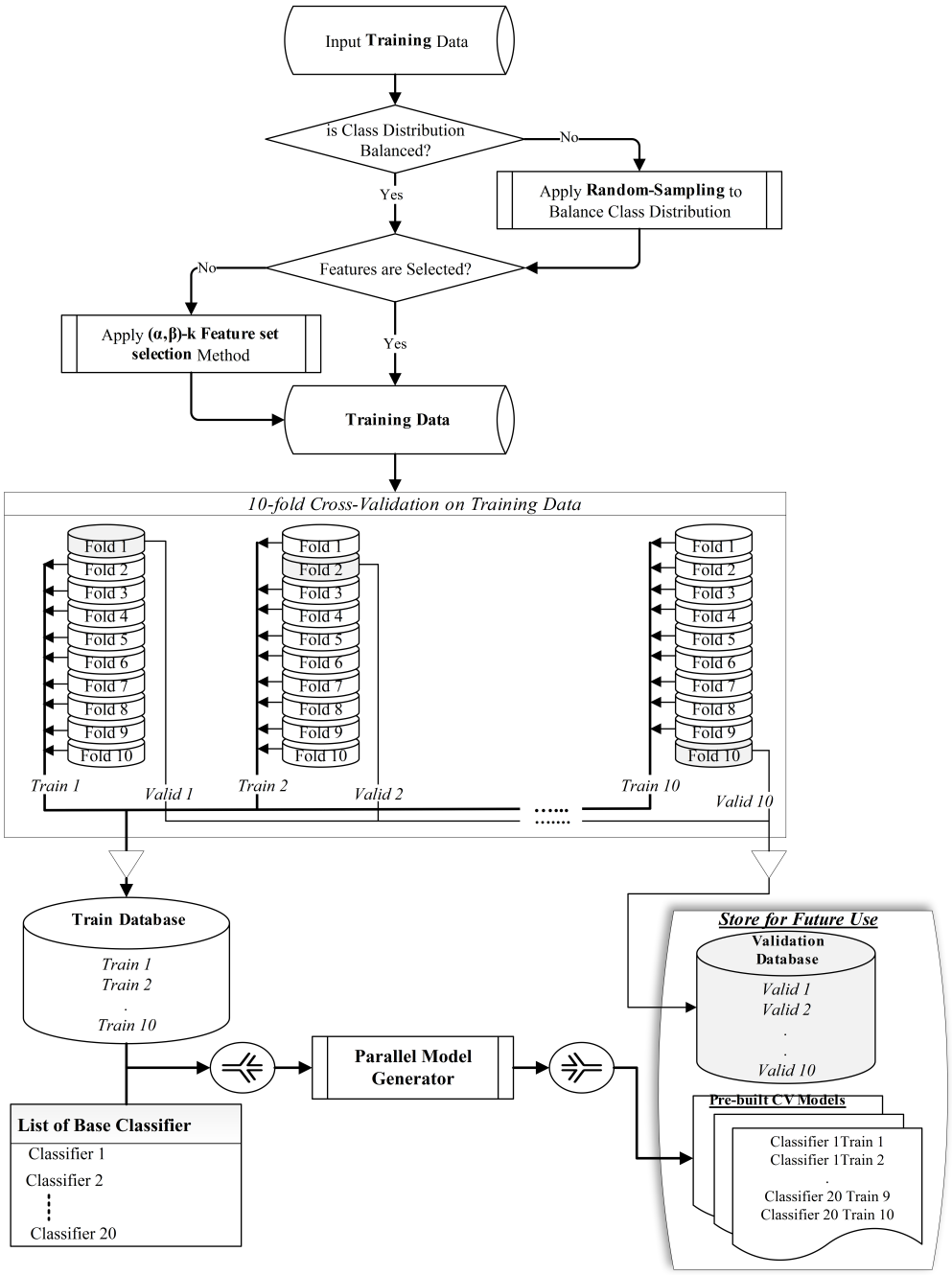


FIGURE 3.2: The steps in preprocessing the training dataset and generating the base classifier models.

3.2.1 Preprocessing

In the initial phase, the whole training dataset is taken as input. If the class distribution of the training dataset is imbalanced, multiple balanced training datasets are created. Then, we get training and testing folds for internal validation process of the GA-EoC using k-fold cross validation on the training dataset. A detail process description is as follows:

Class Distribution Balancing: For a class imbalanced datasets, we keep all samples belong to the minority class label (Ω_{min}) in separate. Then we randomize rest of the dataset containing only majority class labels (Ω_{maj}) and split it into equal proportions to the number of samples of the minority classes. Later we consider one portion from the Ω_{maj} class samples and combined with Ω_{min} class samples to formulate *balanced binary-class* datasets. Afterwards we apply the $(\alpha, \beta) - k$ feature set method (proposed by [Berretta et al., 2005]) on each of these balanced training datasets and consolidate their selected features. The $(\alpha, \beta) - k$ feature set problem finds a minimum set of features that conjointly maximise the inter-class discrimination and intra-class equity.

This method is based on the combinatorial optimisation and used mathematical programming to find the solution. Conventional statistical approaches for feature selection methods used univariate measures of class central tendency or variance to identify most informative features. This approach is different to those in some aspects: evaluates set of features as solutions instead of single feature, retains individual performances of feature within class, and maintains the interrelationship between different classes.

To identify features set for binary-class dataset using the $(\alpha, \beta) - k$ feature set approach, the data should be pre-processed by filtering and discretising of feature values. We have used an implementation of Fayyad and Iranis entropy-based heuristic method [Fayyad and Irani, 1993] for data discretising. It filters out features having insufficient information about class discrimination based on the minimal class information entropy (a measure of concentration threshold, based on the separability of the two classes). Then, it discretises features using the Minimum Description Length (MDL) principle and outputs the feature values into binary (below or above the concentration threshold value). This binary dataset is used as the instance of the $(\alpha, \beta) - k$ feature set problem. It can be represented by a bipartite graph, where pairs of samples from the same classes are denoted as α nodes, pairs of samples from different classes are denoted as β nodes and the features are placed in between α and β nodes. An edge from a feature to a α node denotes that the feature is able

to define the class-labels of a pair of samples (the intra-class similarity of the pair of samples). On the other hand, an edge from a feature to a β node denotes that the feature can define the class-labels of a pair of samples (the inter-class dissimilarity of the pair of samples). The optimization goal of the $(\alpha, \beta) - k$ feature set problem used here is: to cover maximum number of α nodes with maximal number of β nodes with least number of features (k). This optimization problem is solved using mathematical programming approach. If more than one solution is available for the problem, then the solution providing the greatest coverage of sample pairs belonging to different classes is selected. The $(\alpha, \beta) - k$ feature set problem provides a minimum set of features those in the group able to well characterise the dataset by maximising the intra-class relationship and inter-classes discrimination information.

This method has been applied successfully in several studies for feature selection and biomarker discovery for different diseases [Moscato et al., 2005, Berretta et al., 2007a, Moscato et al., 2007, Berretta et al., 2008, Hourani et al., 2008, Rosso et al., 2009]. Hence, features selected by $(\alpha, \beta) - k$ feature set method are kept for further processing of the training dataset. If the class distribution of the training dataset is balanced, then we use as it is for next step.

10-fold Cross Validation on Training Dataset: Afterwards, the training dataset is split using the 10-fold cross validation (CV) method, where the training dataset is divided into 10 equal sized subsamples. Out of these 10 subsamples, one subsample is preserved as the validation data, and the other 9 subsamples are used as training data. This cross-validation process is repeated 10 times (the folds), with each of the 10 subsamples used exactly once as the validation data. Each training and valiadtion split of 10-fold CV is preserved into separate databases named *train database* and *validation database*, respectively.

3.2.2 Model Generation

The second phase of the proposed algorithm involves generating the base classifier models. In this phase only training splits generated using the 10-fold CV on the training dataset are explored. The process trains each participating classifier with each of the training dataset from the train-fold database and save those models into the disk for future use. In this way, a total of 200 models are generated for 20 base classifiers. This model generation process uses the multi-thread pool technique, where each thread is assigned to the task of training one model of a classifier using one training fold from the train-fold database. The fixed thread pool is always assigned to a specified number of threads running. It is

automatically filled with a new thread whenever one is terminated. The size of the pool can be adjusted for the number of CPUs available in the machine.

3.2.3 The GA-EoC:

The genetic algorithm-based search for finding the best ensemble combination takes place in this phase and showed in the Figure 3.3. Let us explain the elements of genetic algorithm. We describe how the problem contexts are represented by the components of the genetic algorithms and how the evolution takes place.

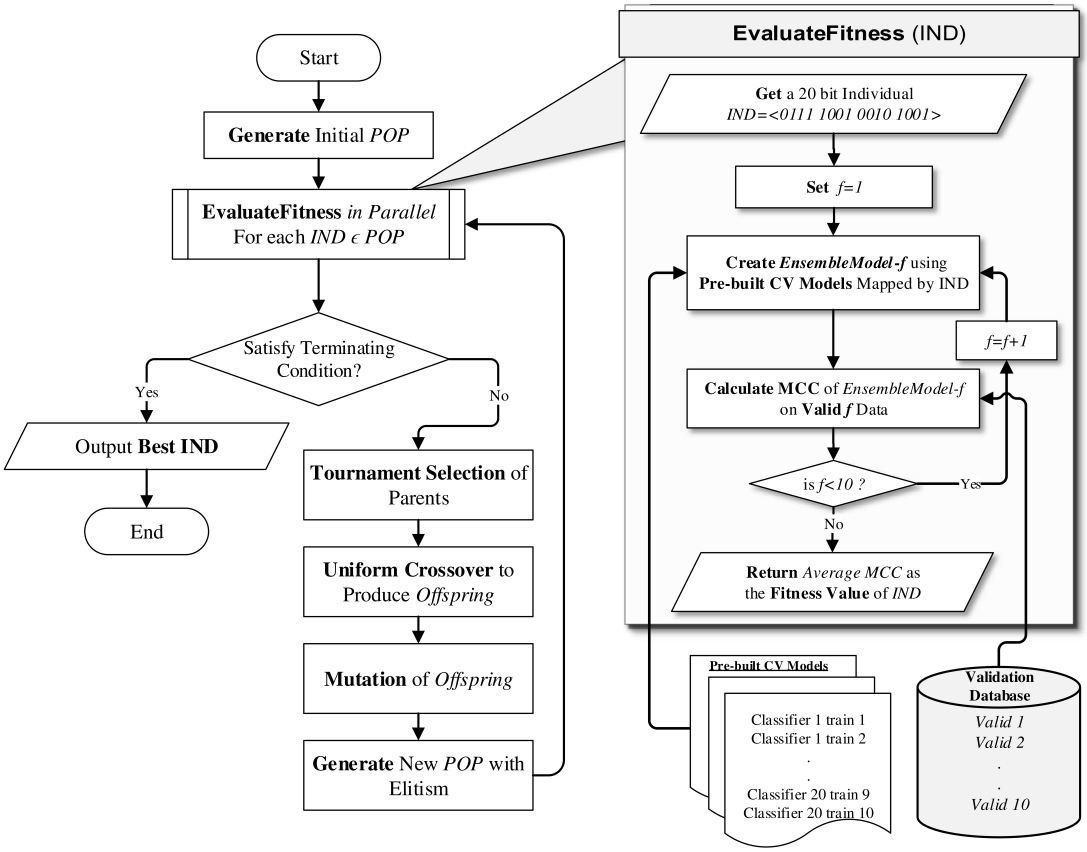


Figure 3.3: Overall process flow of the proposed genetic algorithm-based ensemble of classifiers (GA-EoC) algorithm.

Individual Representation

The efficiency and runtime of a GA depends on the **representation** of the individual and associated fitness function. This fitness function evaluates each individual inside the

population. We have used *20-bit binary encoding* for representation of the individual, where each bit position represents a particular classification algorithm. The selection of a specific classifier depends on the value of the corresponding bit in the individual.

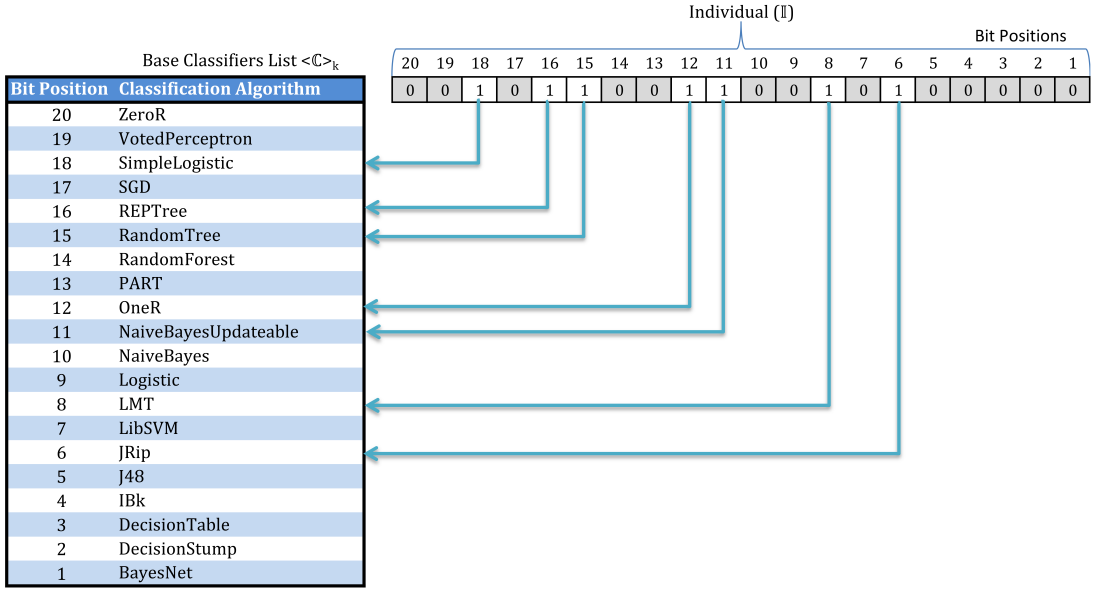


FIGURE 3.4: Representation of an individual in the genetic algorithm for creating the ensemble model

In Figure 3.4, we have a list ($\langle\mathbb{C}\rangle_k = \{\mathbb{C}_1, \cdots, \mathbb{C}_k\}$) of $k = 20$ base classifiers. An individual ($\mathbb{I}$) is also represented by a k-length array. Each element of the array $\mathbb{I}[i]$ selects the classifier $\mathbb{C}_i$ from the classifiers list $\langle\mathbb{C}\rangle_k$, if and only if the i-th bit position contains 1 ($\mathbb{I}[i] \equiv 1$ where $i = 1, \cdots, k$). These selected classifiers form the ensemble represented by an individual. The mapping function for the individual ($\mathbb{I}$) to an EoC ($\mathbb{E}_{mv}$) is denoted by

$$\mathbb{I} \mapsto \mathbb{E}_{mv} = \langle \mathbb{I}[i] \xrightarrow{\equiv 1} \mathbb{C}_i \in \langle\mathbb{C}\rangle_k \rangle. \tag{3.4}$$

Population

The GA-EoC begins with a population containing a set of random individuals. The initial population size can be set depending on the problem. A good rule of thumb for determining the size of the initial population $|\mathbb{P}|$ is given by [Cox, 2005] as

$$|\mathbb{P}| = min\left((5 \times k), \left(\frac{1}{2} \times e\right)\right), \tag{3.5}$$

where k is the size of an individual and e is the maximum number of possible ensemble combinations $\left(e = 2^k\right)$.

In our problem, we have 20 base classifiers that denote the size of an individual. The population size according to Equation (3.5) is given by:

$$\begin{aligned} |\mathbb{P}| &= min\left(\left(5 \times 20\right), \left(\frac{1}{2} \times 2^{20}\right)\right) \\ &= min\left(100, \frac{2.4329 \times 10^{18}}{2}\right) \\ &= 100. \end{aligned}$$

According to the formula, we have created a population containing 100 individuals.

Fitness Function

In the proposed GA, we used the MCC as our fitness function for the corresponding ensemble classifier. It provides a more balanced score among different scores by taking into account the correctness (the number of true predictions) and the incorrectness (the number of false predictions) of the comparisons. The process of generating the fitness value for an individual is illustrated in Figure 3.3. The process starts with taking an individual ($\mathbb{I}$) from the population ($\mathbb{P}$) for a fitness calculation. Then, it generates a set of single classifier models selected using Equation (3.4) and trains them using the training dataset. Afterwards, it creates the ensemble of individual classifiers using a specific combination rule. Finally, the testing dataset is applied to the ensemble and returns the MCC value, which expresses the fitness of the individual.

The fitness value of each $\mathbb{I}_i \in \mathbb{P}$ is calculated as follows. The individual $\mathbb{I}_i$ is mapped to the combination of base classifiers $\mathbb{E}_{mv}$ according to Equation (3.4). The fitness function (`fit`) for individual $\mathbb{I}$ is given by:

$$\texttt{fit}(\mathbb{I}) = \frac{1}{k}\sum_{f=1}^{k} \texttt{MCC}\langle \mathbb{E}_{mv} \text{ for fold} f \rangle. \tag{3.6}$$

The fitness evaluation process has been depicted in the `EvaluateFitness` method in Figure 3.3. We calculate the fitness value of an ensemble combination ($\mathbb{E}_{mv}$) using an unweighted majority-voting approach as per Equation (3.3). The ensemble combination is evaluated on each fold; we create the unweighted majority-vote ensemble using pregenerated base classifier models for each fold and test its classification performance (according

to the MCC metric) against the corresponding validation data taken from the validation database. We repeat this process for each of the k-folds (in our implementation we used $k = 10$) and the average MCC score is treated as the *fitness value* of the individual.

This task is performed in parallel, helping to reduce the running time for the fitness calculation process of individuals in the population. Our objective is to find the best individual from the population with the maximum fitness value attained by Equation (3.6). The objective function is given by:

Objective 3.1. ***MCC Score.*** *Given a population $\mathbb{P}$ with n Individuals. For each individual $\mathbb{I}_i$ in the population,*

$$(\textbf{Maximisation}) \quad Obj_{mcc}(\mathbb{I}) : \arg\max_{i=1,\cdots,n} \texttt{fit}(\mathbb{I}_i) \in \mathbb{P}, \tag{3.7}$$

where the function Obj_{mcc} returns the best individual from the population that *maximises* the fitness value using an unweighted majority vote. This denotes the goodness-of-fit measure for individuals in the population and the individual is denoted as the *fittest individual*.

Creating a New Population

After evaluation of each individual's fitness, the next task is to generate a new population. The new population generation process involves application of three operations named *selection*, *recombination* and *mutation*. We apply an *elitism* strategy, where the m best individuals are promoted to the new generation without any variation. Now, we will briefly describe the processes used for the parent selection and offspring creation.

Parent Selection: Selecting a fittest pairs of parents from the old population would produce fit offspring. First we move m best individuals (in the proposed method we used $m = 1$) directly into the next population. Then, remaining $(n - m)$ individuals are used for generating next population. We used the `TournamentSelection` method for selecting parents for offspring generation (shown in Algorithm 2). In the proposed method, we create a pool of 10 randomly selected individuals from the existing population. Then, the best individual based on the fitness score has been chosen as the first parent for breeding a new individual. We repeat the same process to select a second parent. The role of parent selection is to distinguish among individuals and prefer better individuals as parents of the next generation.

Recombination: The purpose of the recombination or recombination operator is to breed new offspring from a pair of parents. We applied *uniform recombination*, which

Algorithm 2: TournamentSelection for choosing a parent

Input: the population pool of $(n-m)$ individuals $\mathbb{P}$
Output: one parent for recombination $\mathbb{I}_{prnt}$

$size_{trnm} \leftarrow 10$; //Set size of the tournament pool
$\mathbb{P}_{trnm}[\ size_{trnm}\ ]$; //Declare a Tournament Population

//For each place in the tournament, select a random individual
for $i \leftarrow 1$ **to** $size_{trnm}$ **do**
  $rand_{idx} \leftarrow$ Random() ; //random index $rand_{idx} \in \{1, \cdots, (n-m)\}$
  $\mathbb{P}_{trnm}[i] \leftarrow \mathbb{P}[rand_{idx}]$;
end
//Get the fittest Individual as Parent
$\mathbb{I}_{prnt} \leftarrow$ GetFittest $(\mathbb{P}_{trnm})$
return $\mathbb{I}_{prnt}$

facilitates the mixture of two parents by generating a new offspring from them. The process of crossing over a pair of parents to breed a new individual is shown in the Algorithm 3. After selecting two parents according to the parent selection scheme, each gene (bit) of the offspring is inherited randomly from either of the two parents with a fixed recombination rate (R_χ). In this work we applied uniform recombination with a rate of 60%.

Algorithm 3: Recombination on a pair of parents to breed a new individual

Input: a pair of parents $\mathbb{I}_a$ and $\mathbb{I}_b$, uniform recombination rate R_χ
Output: new individual $\mathbb{I}_\chi$

$size \leftarrow |\mathbb{I}_a|$;
for $i \leftarrow 1$ **to** $size$ **do**
  $rn \leftarrow$ Random() ; //random number to be compared with R_χ
  if $rn \leq R_\chi$ **then**
    //Copy bit from the first parent
    $\mathbb{I}_\chi[i] \leftarrow \mathbb{I}_a[i]$;
  else
    //Copy bit from the second parent
    $\mathbb{I}_\chi[i] \leftarrow \mathbb{I}_b[i]$;
  end
end
return $\mathbb{I}_{child}$

Mutation: We have applied *random bit replacement* with a mutation rate (R_μ). To

sustain genetic diversity in the population, we have selected the following mutation rate according to [Cox, 2005]

$$R_\mu = max\left(0.01, \frac{1}{n}\right), \tag{3.8}$$

where n is the population size and R_μ is the mutation rate. So, in the proposed algorithm, we use 0.01 as mutation rate (R_μ). Here, bit values of an individual are flipped randomly. The process to mutate an individual is shown in Algorithm 4.

Algorithm 4: MUTATE an individual

Input: an individual $\mathbb{I}_a$ and the mutation rate R_μ
Output: mutated individual $\mathbb{I}_\mu$

```
size ← |I_a|;
for i ← 1 to size do
    rn ← Random() ;                    //random number to be compared with R_μ
    if rn ≤ R_μ then
        I_μ[i] ←Random() ;                                  //random bit ∈ {0, 1}
    else
        I_μ[i] ← I_a[i] ;
    end
end
return I_μ
```

Terminating Conditions

We have used three terminating conditions for the algorithm. The algorithm finishes the searching if any of the following conditions is satisfied:

1. The total number of generations has reached 10000.
2. The fitness of the best individual of the population has remained stationary for 50 consecutive generations.
3. The fitness value has reached the global optimal value (MCC is equals to 1.0).

3.2.4 Runtime Complexity Analysis of the GA-EoC

Runtime complexity analysis is the part of computational complexity which estimates the resource requirement (in this case the runtime) of an algorithm. The worst case running time estimation is expressed as a function of inputs by big-oh ($\mathcal{O}$) notation. The big-oh is

used to represent the upper-bound of the growth rate of the function and used to express the asymptotic behaviour.

The runtime complexity analysis is common for deterministic approach. It is not easy to find the asymptotic growth of heuristic or stochastic algorithms. However, we will estimate the worst case runtime for our proposed GA-EoC algorithm. First we begin with the pre-processing of GA-EoC. In the pre-processing, the worst case could having a class imbalanced dataset with $1:r$ ratio of c class distribution. The distribution balancing will require $\mathcal{O}(rc)$ time and create $r*c$ number of balanced datasets. Next, if features of the dataset are not already being selected then it will require additional $\mathcal{O}(\alpha)$ for the data pre-processing, where α is the function to represent the asymptotic behaviour of (α,β)-k feature selection method. So, the worst case scenario of having a class imbalanced dataset with features not selected will require $\mathcal{O}\left(rc\alpha\right)$ time. Later on, the runtime estimation for creating l base classifiers model for 10-folds of the training dataset $D_{m,n}$ with m features and n samples can be expressed by $\mathcal{O}(l*k*\mathbb{C}_b\left(D_{m,n}\right))$, where $\mathcal{O}(\mathbb{C}_b)$ express the runtime complexity of model building for base classifier $\mathbb{C}$. Hence, the asymptotic upper bound of *pre-processing* is expressed (including the removal of constant terms from the equation) by

$$\mathcal{O}(PrePro) = \mathcal{O}\left(r*c*\alpha\right) + \mathcal{O}(l*10*\mathbb{C}_b\left(D_{m,n}\right)) \tag{3.9}$$

$$= \mathcal{O}\left(\alpha\right) + \mathcal{O}\left(\mathbb{C}_b\left(D_{m,n}\right)\right). \tag{3.10}$$

Now, we will estimate the runtime behaviour of the proposed genetic algorithm for optimise the ensemble of classifier combination. To estimate the runtime complexity of the GA-EoC, first we will calculate the upper bound of Fitness evaluation method. In the fitness evaluation, the ensemble combination uses 10-fold cross validation by each of the base classifier model for evaluating the MCC of the ensemble combination $\mathbb{E}_{mv}$ created with l base classifiers. So, the runtime complexity (additionally, the removal of constant terms) of ensemble evaluation for calculating the fitness is given by

$$\mathcal{O}(Fit) = \mathcal{O}(\mathbb{E}_{mv}\left(D_{m,n}\right)). \tag{3.11}$$

The algorithm will repeat the evaluation for maximum of G generations. The worst-case estimation for runtime complexity of the GA-EoC for $PopSz$ population size could

be expressed using Equations 3.10–3.11 by

$$\mathcal{O}(GA\text{-}EoC) = \mathcal{O}(PrePro) + \mathcal{O}(PopSz * G * \mathcal{O}(Fit)) \tag{3.12}$$

$$= \mathcal{O}(\alpha) + \mathcal{O}(\mathbb{C}_b(D_{m,n})) + \mathcal{O}(PopSz * G * \mathbb{E}_{mv}(D_{m,n})) \tag{3.13}$$

The runtime require for the preprocessing is negligible compared to the runtime requirements of the genetic algorithm. So, the upper bound of the GA-EoC is estimated by

$$\mathcal{O}(GA\text{-}EoC) = \mathcal{O}(PopSz * G * \mathbb{E}_{mv}(D_{m,n})). \tag{3.14}$$

Hence, the Equation 3.14 estimates the runtime complexity of the proposed GA-EoC algorithm.

3.3 Computational Experiments

In this section, we will report the experimental outcomes of the proposed EoC using the GA described in Section 3.2. To validate the usefulness and competence of the proposed method in the classification problem, we will first report the classification performance of the proposed method using some benchmarking datasets from the machine learning repository maintained by UCI as a service to the machine learning community [Lichman, 2013]. Then, we will report the performance on biological datasets available in the public domain. To compare the performance of the proposed method, we will report the single classifier's best performances achieved by different research articles. We will also evaluate the performance of three popular ensemble methods, namely bagging, boosting and random forest in the same experimental settings as the proposed method.

3.3.1 Experimental Setup

We have two different types of datasets. The first one only contains training data, it does not have separate testing data. The other type has separate training and testing data. The experimental setup was differently assigned for each type.

For datasets of the first type, we adopted the 10-*fold cross-validation (CV)* technique for all of the experiments. In this procedure, each dataset was divided into 10 equal parts and each of our ensemble combination evaluations were repeated 10 times for a single dataset, each time with a different combination of test dataset and training dataset. For

each repetition, we used nine parts of the dataset for training and one part for testing. We repeated the whole process 100 times and report the average values of both the MCC and accuracy.

We also divided the dataset having a single file dataset into two parts, with a ratio of (70 : 30) by use of the *holdout* method. The bigger part (which contained 70% of the examples in the original training dataset) was used for building and evaluating ensemble models (using 10-fold CV on training split). The smaller part (with 30% examples) was used as a validation set for the ensemble combination returned by the proposed method. We repeated the whole process 100 times and report the average values of both the MCC and accuracy.

The datasets with separate training and testing files were treated in the same way as previous types in the training and evolution of the proposed method. The proposed method built models and evaluated their performances throughout the evolution process of the GA using 10-fold CV only on the training file. Finally, we applied the testing file to evaluate the performance (as measured by MCC and accuracy) of the fittest ensemble combination returned by the GA. This whole process was repeated 100 times for these types of datasets and we report average values of both the MCC and accuracy.

3.3.2 Description of Datasets

To evaluate the performance of the proposed method, we chose datasets from the UCI-ML repository [Lichman, 2013], one biological dataset on Alzheimer's disease [Ray et al., 2007] and a real world dataset for the face recognition problem [Pinto et al., 2011]. Two of the datasets have imbalanced class distribution characteristics. Next the datasets are described in detail.

UCI Machine Learning Repository Datasets

We used three datasets from the UCI-ML repository (Table 3.2). The first one is the 'Wisconsin Breast Cancer (Original)', referred to as *WBC* [Mangasarian et al., 1995]. This breast cancer dataset contains nine attributes/features and a total of 699 samples, divided into two classes, named *benign* and *malignant*. The class distribution for this dataset is imbalanced at a ratio of 1.90, where 458 samples are from the *benign* class, and the remaining 241 samples represent the *malignant* class. The second one is the 'Pima Indian diabetes' (*PIMA*) dataset, which contains two classes, eight features and 768 samples. Among these 768 samples, 500 of the tests (about 65.1%) expressed negative results and 268 samples (about 34.9%) confirmed positive results for diabetes. The class

distribution of this dataset is skewed at a ratio of nearly 1.87. We have taken another dataset called *BUPA*. This dataset has the information of some liver disorders that might arise from excessive alcohol consumption by individual persons. It contains a total of 345 samples described by seven features of individual alcohol consumption behaviour divided into two classes. The class imbalance ratio for this dataset is 1.38.

Dataset	**Full Name of the Dataset**	**#Samp (cls dist.)**	**#Feats**
WBC	Wisconsin Breast Cancer (Original)	699 (458,241)	9
PIMA	Pima Indians Diabetes	768 (500,268)	8
BUPA	BUPA Liver Disorders Data Set	345 (145,200)	7
Ray-AD-Trn-18	Ray et.al. - AD (18 Protein)	83 (43,40)	18
RMoscato-AD-Trn-5	Ravetti & Moscato - AD (5 Protein)	83 (43,40)	5
TestSetAD	Ray et.al. - AD (Testing)	92 (42,50)	120
TestSetMCI	Ray et.al. - MCI (Testing)	47 (22,25)	120

TABLE 3.2: Characteristics of the UCI Machine Learning datasets and Alzheimer's Disease datasets used for experiments.

Alzheimer's Disease Datasets

We also used the biological Alzheimer's disease (AD) datasets used in [Ray et al., 2007]. The dataset contained the signalling protein abundances from blood plasma that could classify AD. They measured the abundance of 120 known signalling proteins from 259 archived plasma samples collected from individuals with presymptomatic to late-stage AD. The Alzheimer's and nondemented control (NDC) samples were divided equally into a training set for supervised classification and a test set for class prediction of blinded samples (*Ray-AD-Trn* and *TestSetAD* in Table 3.2). The authors proposed 18 proteins as biomarkers for classification of AD. The classification was also performed on samples from two previously published cohorts of mild cognitive impairment (MCI) patients (named *TestSetMCI*) who converted to AD, developed other dementias (OD) or remained unchanged at a later stage. They combined the NDC and OD classification data into one group that represented the class of patients who did not convert to AD (non-AD). None of the samples from this test set were used in the training process of the classifier.

Ravetti and Moscato [Ravetti and Moscato, 2008] applied the $(\alpha, \beta) - k$ FS method for protein biomarker selection on the same dataset. They reported a set of five proteins as a better biomarker set (referred to as *RMoscato-AD-Trn-5*) for predicting the AD. Their discovered five protein biomarkers produced high accuracy in the prediction of AD from both testing datasets (*TestSetAD* and *TestSetMCI*).

We used training sets *Ray-AD-Trn-18* and *RMoscato-AD-Trn-5* from both studies

to train the proposed GA-EoC separately and tested the predictions on *TestSetAD* and *TestSetMCI*. The classification outcomes are compared for both studies.

3.3.3 Real-world Face-Recognition Dataset

The last dataset used to evaluate the GA-EoC was a subset of the *PubFig*83 dataset used in [Pinto et al., 2011]. In the year 2014, [Chiachia et al., 2014] selected 100 images of each for a group of celebrities and separated them into training and testing sets of 90 and 10 images, respectively. They tested an SVM classifier and identified the five most difficult celebrities to recognise (shown in Table 3.3). This subset of the *PubFig*83 dataset contains 450 image samples (90 per class) in the training set. The testing dataset contains a total of 50 image samples (10 samples per class). This subset of the *PubFig*83 dataset (denoted as *PubFig*05) has been used for further processing.

Person	**class_id**	**#Train Images**	**#Test Images**
Jenifer Lopez	0	90	10
Katherine Heigl	1	90	10
Scarlett Johansson	2	90	10
Mariah Carey	3	90	10
Jessica Alba	4	90	10
Total Samples		**450**	**50**

Table 3.3: Distribution of the training and testing data in *PubFig*05 dataset.

Data Preprocessing

To extract features from the images, the *HT-L3-model* (described in [Cox and Pinto, 2011]) has been used, yielding 25600 features. This dataset is a multiclass dataset. We have tried to build models for all base classifiers used in the proposed method. However, it took an unreasonable amount of times to build model on such a large dataset for some classifiers. Therefore, we have done some preprocessing on the dataset. This is listed step by step below:

Entropy Filtering: We applied an implementation of Fayyad and Irani's entropy-based filtering method to discretise the training dataset and discard features using the minimum description length (MDL) principle [Fayyad and Irani, 1993]. After applying this dimensionality reduction technique to filter out redundant features, only 4878 features passed the process.

Convert Multiclass to Binary-class: Because the proposed method is only able to handle binary-class problems at this stage, we need to convert the dataset to a

binary-class problem. We separated the $PubFig05$ dataset into five binary-class datasets by *one-vs-all* approach (one for each class). Each binary-class training dataset then contained $4,878$ features with a total of 450 samples separated into two (02) classes. Hence, these datasets became imbalanced at a ratio of $1:4$.

Balance the Class Distribution: Each of the binary-class datasets contain 90 samples from one class and 360 samples from all other classes. So, we need to balance the class distribution. For each binary-class dataset, we kept all samples belonging to the minority class label ω_m. Then, we randomly selected samples in equal proportion from the rest of the four (04) classes and labelled them as 'rest-of-ω_m' (or the majority class label ω_M). We repeated the random sampling method four times to obtain four (04) balanced datasets (denoted as $Baln\ Bin$). After completing this process for all binary-class datasets, we finished with 20 datasets (four balanced datasets for each of the five classes).

(α, β)-k **Feature Set Selection:** Then, we applied the (α, β)-k feature set selection method proposed by [Berretta et al., 2005]. For each class (using the respective four balanced binary-class datasets), we applied each procedure described below and the $PubFig05$ turned into 15 datasets (three for each class).

1. We apply the (α, β)-k Feature Set method to each of the balanced binary-class datasets and take the **union** of all selected features. Then, we apply the (α, β)-k Feature Set method to this consolidated binary-class dataset (denoted by UAB). The outcomes are tabulated in Table 3.4.
2. We apply the (α, β)-k Feature Set method similar to the UAB. However, instead of taking the union of selected features, we take the **intersection** of them to consolidate into binary-class datasets (denoted by IAB). The outcomes are tabulated in Table 3.5
3. In this last procedure, we first apply entropy filtering to each of the balanced binary-class datasets before applying the (α, β)-k Feature Set method. Then, we take the *union* of selected features to consolidate into a binary-class dataset (denoted by $(UEAB)$. The outcomes are tabulated in Table 3.6.

Total number of features selected by the (α, β)-k Feature Set Selection methods are summarised in Table 3.7 for three different setups.

Original	***Entropy***	***Multi-Cls 2 Bin***	***Baln Bin***	(α,β)-k	$\bigcup$ ***of Features***	(α,β)-k	***# Features***
25600	4878	Cls 0 vs all	Cls 0 0	4719	4828	Alfa: 4456 Beta: 4362 K: 4656	4656
			Cls 0 1	4642			
			Cls 0 2	4725			
			Cls 0 3	4750			
		Cls 1 vs all	Cls 1 0	4706	4855	Alfa: 4565 Beta: 4506 K: 4702	4702
			Cls 1 1	4800			
			Cls 1 2	4736			
			Cls 1 3	4802			
		Cls 2 vs all	Cls 2 0	4743	4835	Alfa: 4556 Beta: 4490 K: 4712	4712
			Cls 2 1	4687			
			Cls 2 2	4704			
			Cls 2 3	4762			
		Cls 3 vs all	Cls 3 0	4743	4855	Alfa: 4508 Beta: 4416 K: 4678	4678
			Cls 3 1	4629			
			Cls 3 2	4799			
			Cls 3 3	4799			
		Cls 4 vs all	Cls 4 0	4713	4834	Alfa: 4521 Beta: 4450 K: 4738	4738
			Cls 4 1	4735			
			Cls 4 2	4774			
			Cls 4 3	4719			

Table 3.4: Details of features selected by the (α,β)-k Feature Set method for the setup of *UAB* datasets. We applied entropy filtering on the whole multiclass dataset at the beginning and converted it into five (05) binary-class datasets (*Multi-Cls 2 Bin*). Then, we separated each of them into four balanced binary-class datasets (*Baln Bin*) using a random undersampling method. Afterwards, we applied the (α,β)-k Feature Set method on each of the balanced binary-class datasets and we took the **union** of selected features for each binary-class dataset. Finally, we applied the (α,β)-k Feature Set selection method to each of the binary-class datasets and obtained a set of features.

Original	***Entropy***	***Multi-Cls 2 Bin***	***Baln Bin***	(α, β)-k	$\cap$ ***of Features***	(α, β)-k	***# Features***
25600	4878	Cls 0 vs all	Cls 0 0	4719	4544	Alfa: 4363 Beta: 4327 K: 4495	4495
			Cls 0 1	4642			
			Cls 0 2	4725			
			Cls 0 3	4750			
		Cls 1 vs all	Cls 1 0	4706	4616	Alfa: 4506 Beta: 4470 K: 4598	4598
			Cls 1 1	4800			
			Cls 1 2	4736			
			Cls 1 3	4802			
		Cls 2 vs all	Cls 2 0	4743	4585	Alfa: 4476 Beta: 4427 K: 4563	4563
			Cls 2 1	4687			
			Cls 2 2	4704			
			Cls 2 3	4762			
		Cls 3 vs all	Cls 3 0	4743	4561	Alfa: 4410 Beta: 4361 K: 4501	4501
			Cls 3 1	4629			
			Cls 3 2	4799			
			Cls 3 3	4799			
		Cls 4 vs all	Cls 4 0	4713	4602	Alfa: 4419 Beta: 4387 K: 4553	4553
			Cls 4 1	4735			
			Cls 4 2	4774			
			Cls 4 3	4719			

TABLE 3.5: Details of features selected by the (α, β)-k Feature Set method for the setup of *IAB* datasets. In this data preprocessing, we applied entropy filtering to the whole multiclass dataset at the beginning and converted it into five (05) binary-class datasets (*Multi-Cls 2 Bin*). Then, we separated each of them into four balanced binary-class datasets (*Baln Bin*) using a random undersampling method. Afterwards, we applied the (α, β)-k Feature Set method to each of the balanced binary-class datasets and we took the **intersection** of selected features for each binary-class dataset. Finally, we applied the (α, β)-k Feature Set selection method to each of the binary-class datasets and obtained a set of features.

Original	***Entropy***	***Multi-Cls to Bin***	***Baln Bin***	***Entropy***	(α, β)-k	$\bigcup$ ***of Features***
25600	4878	Cls 0 vs all	Cls 0 0	493	Alfa:87, Beta:126, k:285	795
			Cls 0 1	634	Alfa:90, Beta:138, k:294	
			Cls 0 2	594	Alfa:115, Beta:167, k:384	
			Cls 0 3	609	Alfa:97, Beta:160, k:333	
		Cls 1 vs all	Cls 1 0	1313	Alfa:252, Beta:378, k:851	1554
			Cls 1 1	1192	Alfa:175, Beta:239, k:522	
			Cls 1 2	1368	Alfa:265, Beta:388, k:858	
			Cls 1 3	1345	Alfa:220, Beta:308, k:684	
		Cls 2 vs all	Cls 2 0	1726	Alfa:389, Beta:672, k:1278	2273
			Cls 2 1	1458	Alfa:353, Beta:510, k:1068	
			Cls 2 2	1789	Alfa:419, Beta:548, k:1263	
			Cls 2 3	1810	Alfa:444, Beta:574, k:1329	
		Cls 3 vs all	Cls 3 0	2334	Alfa:389, Beta:672, k:1278	2821
			Cls 3 1	2435	Alfa:599, Beta:787, k:1743	
			Cls 3 2	2297	Alfa:582, Beta:698, k:1763	
			Cls 3 3	2106	Alfa:550, Beta:822, k:1760	
		Cls 4 vs all	Cls 4 0	1018	Alfa:169, Beta:242, k:592	1081
			Cls 4 1	694	Alfa:128, Beta:177, k:443	
			Cls 4 2	748	Alfa:109, Beta:148, k:386	
			Cls 4 3	583	Alfa:109, Beta:148, k:386	

Table 3.6: Details of features selected by the (α, β)-k Feature Set method for the setup of *UEAB* datasets. In this data preprocessing, we applied entropy filtering to the whole multiclass dataset at the beginning and converted it into five (05) binary-class datasets (*Multi-Cls 2 Bin*). Then, we separated each of them into four balanced binary-class datasets (*Baln Bin*) using a random undersampling method and applied entropy filtering to each of them. Afterwards, we applied the (α, β)-k Feature Set method to each of the entropy filtered balanced binary-class datasets and we took the **union** of selected features for each binary-class dataset. Finally, we applied the (α, β)-k Feature Set selection method to each of the binary-class datasets and obtained a set of features.

Binary Class - Dataset	*UAB*	*IAB*	*UEAB*
Class 0 vs All	4656	4495	795
Class 1 vs All	4702	4598	1554
Class 2 vs All	4712	4563	2273
Class 3 vs All	4678	4501	2821
Class 4 vs All	4738	4553	1081
Average Features	**4697**	**4542**	**1705**

TABLE 3.7: Outcome of the (α, β)-k Feature Set Selection method for three different setups (UAB, IAB, UEAB) showing the number of selected features per binary-class datasets of $PubFig05$.

3.4 Performance of the Proposed Method

The GA-EoC algorithm was implemented in Java. We used the implementations of base classifiers from the WEKA framework version 3.6. All of the experiments except those on the *PubFig05* dataset were executed on a Dell PowerEdge III with Dual Xeon 5550 2.67 GHz (8 Cores) and 32 GB RAM. The machine was running on the Red Hat Enterprise Linux AS release 4 operating system. We executed the experiments for the *PubFig05* dataset on a Xenon Radon 6170 Supermicro server with Quad E5-4650 Sandy Bridge 2.7 GHz (32 cores) and 512 GB RAM because of the high memory requirement. The source code of GA-EoC is available at `https://sourceforge.net/projects/geneticensembleclassifier/` and other relevant information is available in the Appendix A.

Now, we will discuss the classification performances of the proposed method using the aforesaid datasets. We used the *MCC* as the performance measure. We also report the classification *accuracy* in addition to the MCC values, to compare more easily with the state-of-the-art methods. The features for all three datasets were available from the UCI-ML repository; therefore, we did not apply the (α, β)-k Feature Set selection phase for these datasets. However, the class-imbalanced characteristic of these datasets necessitates the application of a class distribution balancing phase. After class rebalancing, the cross-validation was performed using GA-EoC and the performance metrics were calculated. To compare the performance of BUPA dataset classification by the proposed method, we used the same 5-fold CV as used by [Fernández et al., 2010]. They provide benchmark results for different classifiers for the BUPA classification task and we used it as the baseline. We repeated the whole process 100 times and counted the performance of the proposed method for the best combinations on the testing datasets.

The classification performance of the base classifiers and GA-EoC in terms of MCC and accuracy are presented in Table 3.8 and Table 3.9, respectively.

Classifier	WBC	PIMA	BUPA	AD-18	MCI-18	AD-5	MCI-5	UAB	IAB	UEAB	#W	#B
BayesNet	0.94	0.43	0.04	0.79	0.31	0.91	0.28	0.37	0.36	0.46	0	0
DecisionStump	0.84	0.37	0.20	0.80	0.15	0.80	0.15	0.21	0.11	0.21	0	0
DecisionTable	0.87	0.38	0.14	0.80	0.10	0.80	0.10	0.24	0.17	0.22	0	0
IBk	0.90	0.33	0.24	**0.89**	0.37	0.79	0.16	0.59	0.60	0.57	0	1
J48	0.89	0.42	0.33	**0.89**	0.19	0.80	0.27	0.29	0.30	0.29	0	1
JRip	0.89	0.43	0.32	0.59	0.37	0.85	0.10	0.44	0.24	0.31	0	0
LibSVM	0.91	0.00	0.13	0.85	0.40	0.87	0.37	0.53	0.52	0.63	0	0
LMT	0.91	0.48	0.41	0.76	0.41	0.89	0.53	**0.65**	0.63	0.59	0	1
Logistic	0.92	0.48	0.35	0.72	0.41	0.89	0.51	0.49	0.49	0.38	0	0
NaiveBayes	0.91	0.47	0.15	0.87	0.33	0.91	0.35	0.39	0.36	0.41	0	0
NaiveBayesUpdateable	0.91	0.47	0.15	0.87	0.33	0.91	0.35	0.39	0.36	0.41	0	0
OneR	0.84	0.33	0.09	0.80	0.15	0.80	0.15	0.17	0.09	0.17	0	0
PART	0.87	0.43	0.26	0.81	0.32	0.82	0.18	0.22	0.24	0.34	0	0
RandomForest	0.91	0.43	0.36	0.78	0.21	0.89	0.19	0.35	0.30	0.46	0	0
RandomTree	0.86	0.32	0.24	0.63	0.08	0.69	0.07	0.23	0.09	0.21	0	0
REPTree	0.86	0.44	0.28	0.80	0.15	0.80	0.15	0.33	0.26	0.28	0	0
SGD	0.93	0.50	0.30	0.81	**0.44**	0.89	**0.48**	0.59	**0.64**	**0.61**	0	4
SimpleLogistic	0.91	0.48	0.36	0.76	0.41	0.89	0.53	**0.65**	0.63	0.59	0	1
VotedPerceptron	0.81	0.13	0.33	0.85	0.27	0.83	0.23	0.49	0.47	0.54	0	0
ZeroR	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	10	0
GA-EoC(avg)	**0.99**	**0.94**	**0.50**	**0.89**	0.36	**0.92**	0.27	0.62	0.57	0.56	0	5
GA-EoC (Stdev)	0.007	0.040	0.009	0.037	0.039	0.056	0.046	0.147	0.085	0.118		

Table 3.8: Classification performances (MCC scale) of the base classifiers and genetic algorithm-based ensemble of classifiers for all experiments. We used 10-fold CV for the experiments with the WBC, PIMA and BUPA datasets. For the AD-18 and MCI-18 experiments, we used the Ray-AD-Trn-18 dataset for training but TestSetAD and TestSetMCI as the testing dataset. We trained the classifier with RMoscato-AD-Trn-5 and tested on TestSetAD and TestSetMCI datasets for the AD-5 and MCI-5 experiments. For the UEAB, IAB and UAB experiments, the GA-EoC was trained on their own training datasets and performances have been measured on the respective test datasets. The same training and testing data manipulation approaches have been used to measure the classification performance of all experiments. The #B row express the number of times the algorithm exhibited best performance and #W denotes the frequency of the algorithm appeared as worst performing classifier.

Classifier	WBC	PIMA	BUPA	AD-18	MCI-18	AD-5	MCI-5	UAB	IAB	UEAB	#W	#B
BayesNet	97.28	74.35	56.81	89.13	63.83	95.65	63.83	78.00	78.80	81.20	1	0
DecisionStump	92.42	71.88	61.74	90.22	57.45	90.22	57.45	80.80	79.60	80.80	0	0
DecisionTable	94.13	72.40	59.71	90.22	55.32	90.22	55.32	76.40	74.40	77.60	0	0
IBk	95.28	70.18	63.19	94.57	65.96	89.13	57.45	86.80	87.60	86.40	0	0
J48	95.14	73.83	67.83	94.57	59.57	90.22	63.83	76.80	78.00	76.40	0	0
JRip	95.14	74.61	67.83	79.35	65.96	92.39	55.32	81.20	72.80	76.80	0	0
LibSVM	95.71	65.10	59.42	92.39	68.09	93.48	68.09	86.80	86.40	88.80	0	0
LMT	95.99	77.47	71.59	88.04	**70.21**	94.57	**74.47**	**89.20**	88.80	87.20	0	3
Logistic	96.57	77.21	68.99	85.87	**70.21**	94.57	**74.47**	83.60	85.60	80.00	0	2
NaiveBayes	95.99	76.30	53.91	93.48	63.83	95.65	65.96	76.80	76.40	76.40	0	0
NaiveBayesUpdateable	95.99	76.30	53.91	93.48	63.83	95.65	65.96	76.80	76.40	76.40	0	0
OneR	92.70	70.83	55.94	90.22	57.45	90.22	57.45	76.80	74.40	76.80	0	0
PART	94.13	74.48	64.06	90.22	65.96	91.30	59.57	76.40	77.60	78.00	0	0
RandomForest	95.99	74.22	68.12	89.13	59.57	94.57	59.57	82.80	81.20	84.40	0	0
RandomTree	93.71	69.14	63.48	81.52	53.19	83.70	53.19	75.60	70.00	75.20	3	0
REPTree	93.85	75.39	65.51	90.22	57.45	90.22	57.45	77.20	74.00	80.00	0	0
SGD	96.71	77.99	66.96	90.22	**70.21**	94.57	72.34	88.00	**89.20**	**87.60**	0	3
SimpleLogistic	95.99	77.47	69.28	88.04	**70.21**	94.57	**74.47**	**89.20**	88.80	87.20	0	3
VotedPerceptron	90.99	65.36	67.54	92.39	63.83	91.30	61.70	84.00	82.40	84.00	0	0
ZeroR	65.52	65.10	57.97	45.65	46.81	45.65	46.81	80.00	80.00	80.00	6	0
GA-EoC (avg)	**99.43**	**97.43**	**75.72**	**94.66**	67.14	**95.91**	62.98	88.40	86.80	86.80	0	5
GA-EoC (Stdev)	0.32	1.71	0.48	1.89	2.24	2.01	2.02	4.34	3.03	3.63		

Table 3.9: Classification accuracies achieved by the base classifiers and genetic algorithm-based ensemble of classifiers for all experiments. We used 10-fold CV for the experiments with WBC, PIMA and BUPA datasets. The classifiers were trained using the Ray-AD-Trn-18 dataset and tested on TestSetAD and TestSetMCI for the AD-18 and MCI-18 experiments. We trained the classifiers with RMoscato-AD-Trn-5 and tested on TestSetAD and TestSetMCI datasets for the AD-5 and MCI-5 experiments. For the UEAB, IAB and UAB experiments, classifiers were trained on their own training datasets and performances were measured on the respective testing datasets. The same training and testing data manipulation approaches were used to measure the classification performance in all experiments. The #B row express the number of times the algorithm exhibited best performance and #W denotes the frequency of the algorithm appeared as worst performing classifier.

3.4.1 Classification Performances on UCI Machine Learning Repository Datasets

We applied the (α, β)-k-Feature Selection method for datasets only where features were not selected. The features of UCI-ML datasets were already selected at source, hence the (α, β)-k-Feature Selection method was not performed. However, the class-imbalanced characteristic of these datasets necessitates the application of a class distribution balancing phase. After class rebalancing, the cross-validation was performed using GA-EoC and the performance metrics were calculated.

The classification performance of the base classifiers and GA-EoC in terms of MCC and accuracy are presented in Table 3.8 and Table 3.9, respectively. For UCI-ML datasets, we observe that the proposed method achieved an average MCC score of 0.99, 0.94 and 0.50 for WBC, PIMA and BUPA, respectively (Table 3.8). The closest performing base classifiers are the `BayesNet` classifier with 0.94 MCC score for the WBC dataset, the `SGD` classifier with 0.50 MCC score for the PIMA dataset and the `LMT` classifier with 0.41 MCC score for the BUPA dataset. The MCC gap between the best base classifier and the proposed GA-EoC is at least 0.05 for the datasets taken from the UCI-ML repository.

For the WBC dataset, the average accuracy of GA-EoC is 99.43% (Table 3.9). The best performing base classifier `BayesNet` achieved 97.28% classification accuracy on this dataset. The GA-EoC achieved 97.43% classification accuracy on the PIMA dataset whereas the closest performing base classifier `SimpleLogistic` achieved 77.99% accuracy. For the BUPA dataset classification, the proposed method achieved an average accuracy of 75.72% and `LMT`, the best performing base classifier, achieved an accuracy of 71.59%. The accuracy gaps between the GA-EoC and the best performing base classifier are 2%, 20% and 4% for WBC, PIMA and BUPA datasets, respectively.

From the results in Table 3.8 and Table 3.9, it is clear that the proposed GA-EoC method exhibited better performance than its constituent base classifiers. Moreover, the standard deviations of the GA-EoC scores (both in terms of MCC and accuracy) over 100 runs for all datasets are very low. For example, for the WBC dataset, the standard deviation in MCC is 0.007, which is less than 1%. A similarly low standard deviation value was observed for the BUPA dataset. However, the GA-EoC converges with five different MCCs among 100 repeated runs in the WBC dataset with a standard deviation of 0.040. This demonstrates that the proposed GA-EoC method performs consistently. We observed that for all of the UCI-ML datasets, the proposed method produced better classification performance than its base classifiers and there was no single base classifier that performed consistently well on all of these three datasets. These results support the effectiveness of the proposed GA-based ensemble construction technique.

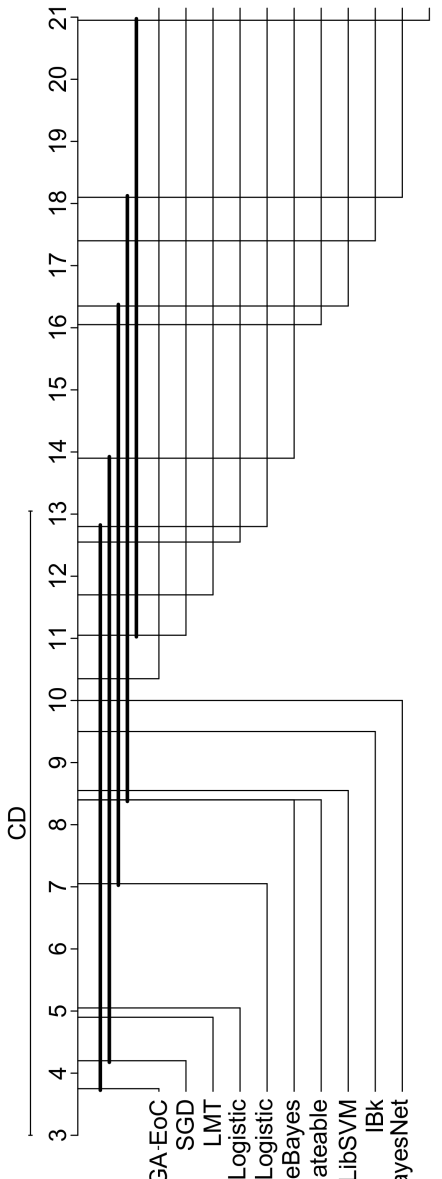


FIGURE 3.5: The critical difference (CD) plot shown the critically significance of classification algorithms over multiple datasets for the experimental outcomes in Table 3.8 using MCC score. The critical distance is showing the significance level of 0.05.

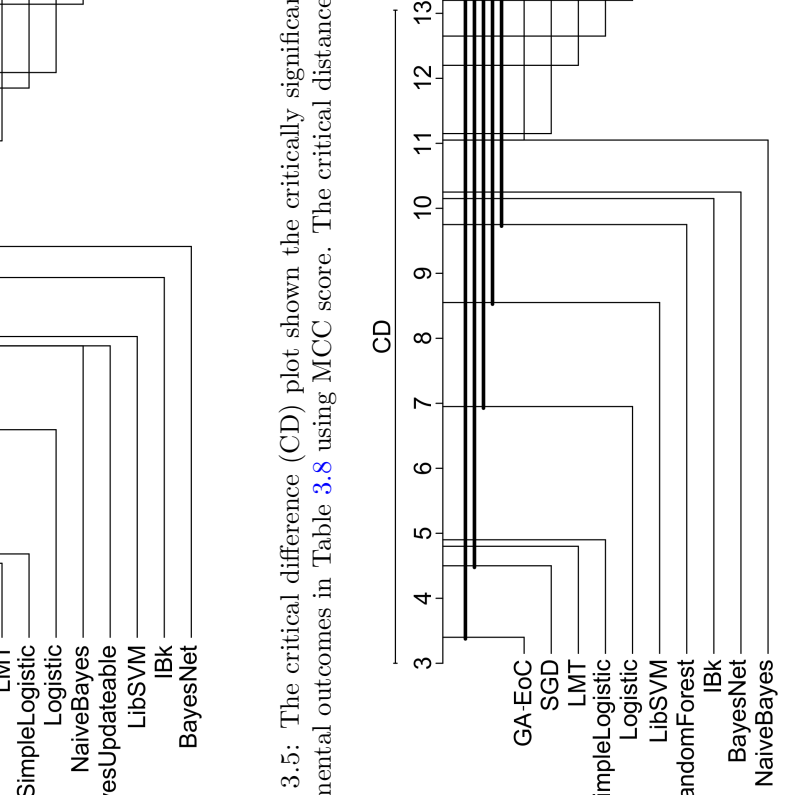


FIGURE 3.6: The critical difference (CD) plot shown the critically significance of classification algorithms over multiple datasets for the experimental outcomes in Table 3.9 using accuracy score. The critical distance is showing the significance level of 0.05.

Statistical Comparison of Results

Let usfigure outif there is one or more classification algorithms whose performance can be regarded as significantly different in Table 3.8 and Table 3.9. Wilcoxson signed-rank test is usually used to compare two algorithms over set of problems. But we have 20 base classifiers and Wilcoxon test is not suitable in this case. Hence, we apply non parametric methods for comparing multiple algorithms, the classical Friedman test and acorrection by Iman and Davenport as recommended in [Demšar, 2006] for comparing more than five algorithms.

The Iman Davenport's correction of Friedman's rank sum test provides chi-squared value $Q = 14.043$ for results for MCC score in Table 3.8. The *p-value* $<$ 2.2*e*-16 indicates that there is one or more classification algorithms exist whose performance is significantly different. The Corrected Friedman's chi-squared value $Q = 8.995$ is found for the accuracy scores in Table 3.9. It has the same *p-value* as Table 3.8. So, for both the MCC and Accuracy scores there exist at least one classifier whose performance is significantly different from others.

To visualise the critical difference (CD) among classifiers over a multiple problems (datasets), the plot of the CD is generated from Nemenyi test. Although the test is not a recommended choice in practice, however the plot is a good way to visualise the difference. Any two algorithms whose performance difference is greater than CD are regarded as significantly different. The algorithms with no significant differences of average ranks are grouped together using a horizontal line. The critical distance shown in above the plot is the required size of difference for considering two algorithms as significantly different. We used critical distance (CD) plot recommend by Demšar [Demšar, 2006] for visualising the critical difference between algorithms over multiple datasets. Here we used $alpha = 0.05$ which regarded as the significance level at 95%. In Figure 3.5, the CD plot shows the average rank difference of all base classifiers and GA-EoC for MCC scores. The CD plot for accuracy, is shown in Figure 3.6. The critical difference value is $CD = 10.048$ for both measures. Considering the GA-EoC for both of the measures, `REPTree`, `DecisionTable`, `DecisionStump`, `OneR`, `RandomTree` and `ZeroR` are in critical distance.

To better understand which base classifiers are statistically significant comparing the average performance of the GA-EoC in Table 3.10. We conducted a post-hoc calculation of Friedman's Aligned Rank test with Iman Davenport's correction. The *p-value* is shown for the base classifiers and GA-EoC for MCC and Accuracy scores. The *p-value* smaller than 0.05 in a row expressed that the performance of GA-EoC is statistically significant compared with the base classifier. Comparing the *p-value* for MCC scores, `IBk`, `LibSVM`, `LMT`, `Logistic`, `SGD` and `SimpleLogistic` are statistically similar to the GA-EoC. The

Classifier	p-value	
	MCC	*Accuracy*
BayesNet	0.00903	0.00342
DecisionStump	6.72E-06	0.000112
DecisionTable	7.21E-06	1.12E-05
IBk	**0.0867**	0.0409
J48	0.00193	0.00226
JRip	0.00167	0.000423
LibSVM	**0.151**	0.155
LMT	**0.77**	**0.831**
Logistic	**0.265**	**0.213**
NaiveBayes	0.0406	0.00422
NaiveBayesUpdateable	0.0406	0.00422
OneR	2.28E-06	6.27E-06
PART	0.000573	0.00105
RandomForest	0.00731	0.0167
RandomTree	4.99E-07	7.76E-08
REPTree	0.000176	0.000167
SGD	**0.756**	**0.797**
SimpleLogistic	**0.732**	**0.763**
VotedPerceptron	0.0248	0.0101
ZeroR	3.69E-10	2.96E-08

TABLE 3.10: The *p-values* from statistical test of classification performances of base classifiers and GA-EoC for benchmarking datasets using post-hoc calculation of Friedman's Aligned Rank test with Iman Davenport's correction. The statistically similar base classifiers of GA-EoC are shown in bold face and statistically significant classifiers are shown in normal font face.

GA-EoC is statistically significantly better than Remaining 14 base classifiers for MCC scores. The `LMT, Logistic, SGD` and `SimpleLogistic` exhibited the same performances of GA-EoC considering the accuracy score. Hence, the GA-EoC is statistically significant than remaining 16 base classifiers for accuracy score.

From the statistical test on the results, we found that GA-EoC is a significantly better approach than the majority of base classifiers (better than 14 base classifiers for MCC and 16 base classifiers for accuracy scores among 20 base classifiers).

Comparison with State-of-the-Art Classifiers

To compare the performance of the proposed method, we did a literature survey to determine the accuracy of the state-of-the-art methods for the classification tasks on WBC and PIMA datasets. The outcome of the survey is tabulated in Table 3.11 and Table 3.12 for WBC and PIMA, respectively.

Method	Accu	Reference
MSC (Multi-scale Classifier)	94.90%	[Lovell and Bradley, 1996]
RIAC (Rule Induction Algorithm Based on Approximation)	95.00%	[Hamilton et al., 1996]
IRSS (Influential rule search scheme)	95.89%	[Chatterjee and Rakshit, 2004]
C4.5 (decision tree)	96.00%	[Hamilton et al., 1996]
ANFIS (Adaptive-Network-based Fuzzy Inference System)	96.20%	[Jang, 1993]
ARFIS (Adaptive T-S-type Rough-Fuzzy Inference System)	96.35%	[Lee et al., 2007]
MAFIE with FCM (Modified Adaptive Fuzzy Inference Engine)	97.24%	[Hossen et al., 2013]
C-MLP2LN (Constructive MLP converted to Logical Network)	99.00%	[W. et al., 2011]

Table 3.11: Performance of state-of-the-art techniques on the WBC dataset classification task.

Method	Accu	Reference
EnSVM (Ensemble of Support Vector Machines)	75.03%	[Liu et al., 2006]
MLP-NN (Multilayer Perceptron Neural Network)	75.40%	[Afsari et al., 2013]
NeC4.5 (Neural Ensemble-based Decision Trees C4.5)	75.40%	[Zhou and Jiang, 2004]
FLEXNFIS (Flexible Neuro-Fuzzy Systems)	78.60%	[Rutkowski and Cpalka, 2003]
eClass (Evolving Fuzzy Rule-based classifier)	79.37%	[Lekkas and Mikhailov, 2010]
MOG3P (Multi-objective Genetic Programming)	81.87%	[Icke and Rosenberg, 2011]
RBF-SVM (Support Vector Machine)	82.20%	[Karatsiolis and Schizas, 2012]
L_2-norm S^3VMs (Semi-supervised support vector machines)	84.41%	[Yang and Wang, 2013]

Table 3.12: Performance of state-of-the-art techniques on the PIMA dataset classification task.

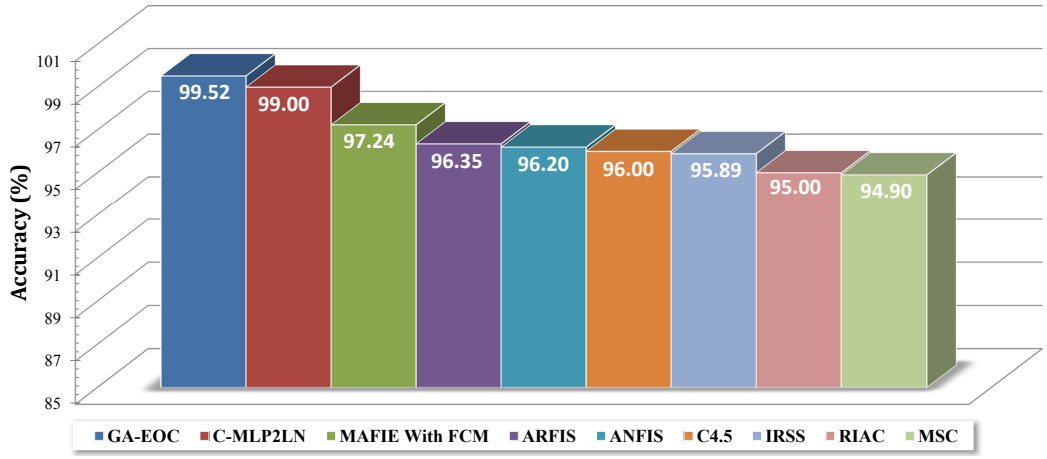


FIGURE 3.7: Classification accuracy of the proposed genetic algorithm-based ensemble of classifiers vs the top eight (08) studies for the WBC dataset using a 10-fold cross-validation method.

For the WBC dataset, we tabulated the classification accuracy of the top eight (8) state-of-the-art classifiers, in ascending order of their performances, in Table 3.11. This dataset was made publicly available in 1992. The next year, [Jang, 1993] proposed an Adaptive-Network-based fuzzy interface system (ANFIS), which achieved 96.20% classification accuracy. It is evident that most of the classifiers produced approximately 96% accuracy on the classification of this dataset. It can be observed that different types of hybrid classification algorithms, based on decision trees, fuzzy systems and neural networks have been applied to classify the WBC dataset. Among them, the `C-MLP2LN` method proposed by [W. et al., 2011] produced the highest accuracy of 99.00%. The classification performance of GA-EoC is plotted in Figure 3.7 using the accuracy measure. It is evident that the proposed method produces better accuracy for the WBC dataset.

The PIMA dataset classification has been reported by several researchers as a tough dataset for classification [Afsari et al., 2013]. We tabulated the top eight state-of-the-art methods, sorted in ascending order of their accuracy, in Table 3.12. Current benchmarks showed that most of the classifiers achieved less than 80% accuracy on the PIMA dataset classification task. We find that SVMs and their variations are more successful in this dataset classification task (such as EnSVM, RBF-SVM and semi-supervised SVM in Table 3.12). In the top eight classifiers, we find an ensemble of SVMs produced an accuracy of 75%. Recently, [Yang and Wang, 2013] were able to achieve nearly 85% accuracy for the PIMA dataset classification using a semi-supervised SVM. To the best of our knowledge, this is the highest accuracy reported in the literature so far. The column graph in Figure 3.8 shows the performances of the GA-EoC with the top eight methods. It can be

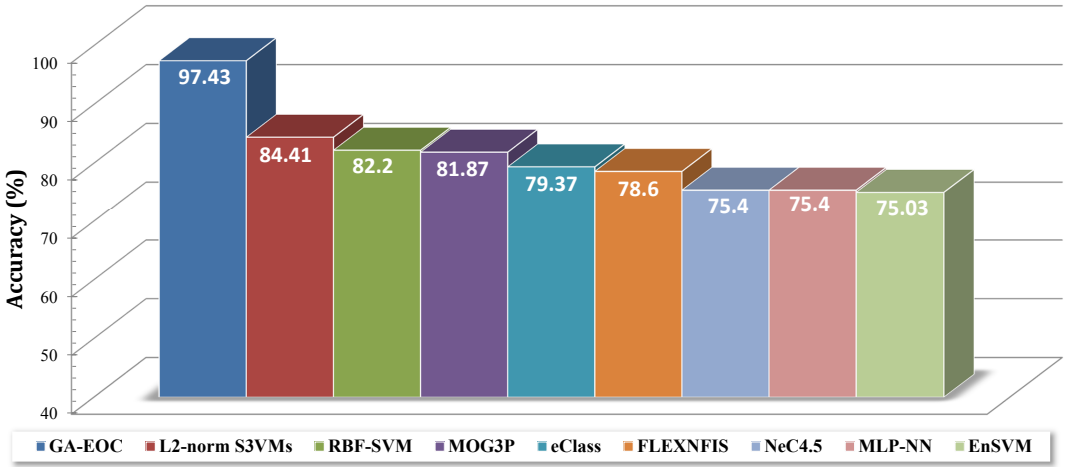


Figure 3.8: Classification accuracy of the proposed genetic algorithm-based ensemble of classifiers vs top eight (08) studies for the PIMA dataset using a 10-fold cross-validation method.

observed that our method clearly outperforms the best classifier from previous studies by producing an average accuracy of 97% on 100 runs.

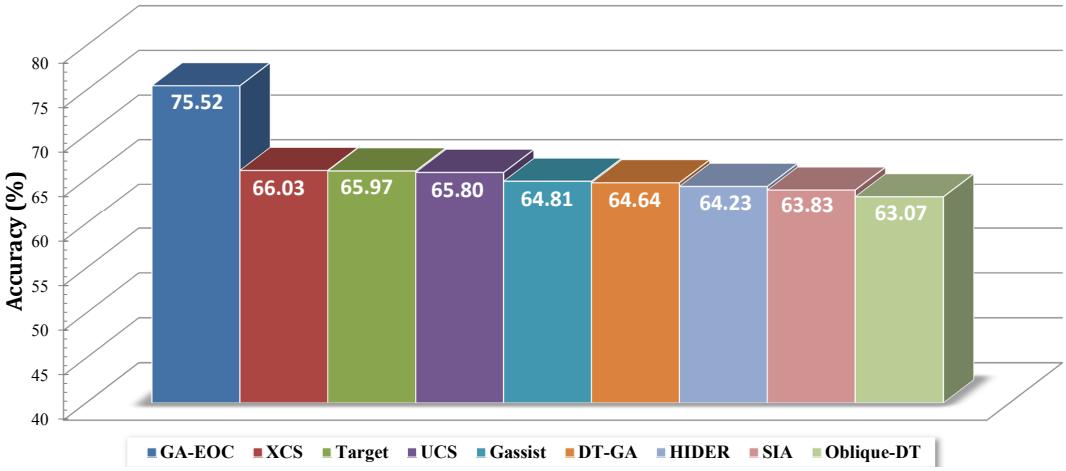


Figure 3.9: Classification accuracy of the proposed genetic algorithm-based ensemble of classifiers vs top eight (08) classifiers for the BUPA dataset using the 5-fold cross-validation method of [Fernández et al., 2010].

[Fernández et al., 2010] provided a benchmark for the GA-based machine learning algorithms for classification tasks. They include a comparative study of the genetics-based machine learning (GBML) methods in terms of classification accuracy for the BUPA dataset. We used their results as a baseline for comparison. We present a column graph in

Figure 3.9 with the average performance of the GA-EoC with the top eight state-of-the-art methods using 5-fold CV. It can be seen that the proposed method clearly outperformed the best classifier from the literature, having an accuracy of 66%, by producing 75% accuracy on the same 5-fold CV setup.

The classification performances achieved by the proposed method were promising for the datasets used from the UCI repository. We observed that for all of those dataset classification tasks, the proposed method produced better accuracy than the state-of-the-art techniques. We can expect that the proposed method would perform consistently in other cases also.

3.4.2 Performance on Alzheimer's Disease Datasets

Next, we applied the GA-EoC for classification of AD and non-Alzheimer's Disease (NAD) using the aforementioned training and testing datasets. We have compared the classification performance of the proposed GA-EoC with the performance achieved by [Ray et al., 2007] and [Ravetti and Moscato, 2008] using their respective biomarkers. The feature sets for these datasets were available and class distribution was balanced, therefore, neither the (α, β)-k Feature Set selection method nor the class rebalancing phase were necessary.

First, we compared the classification performances achieved by [Ray et al., 2007], [Ravetti and Moscato, 2008] and the proposed GA-EoC using an 18-protein biomarker. Figure 3.10 compares the best classification results generated by these three methods using confusion matrices. We also tabulated the average classification performance of these methods in Table 3.13.

Test Dataset	**[Ray et al., 07]**		**[Ravetti & Moscato, 08]**		**The GA-EoC**		**Gap**	
	Acu	*MCC*	*Acu*	*MCC*	*Acu*	*MCC*	*Acu*	*MCC*
TestSetAD	89.13%	0.78	90.82%	0.82	**94.66%**	**0.89**	3.84%	0.07
TestSetMCI	**80.85%**	**0.63**	66.19%	0.34	67.14%	0.36	-13.71%	-0.27

TABLE 3.13: Average classification performances (in terms of accuracy and MCC) using an 18-protein biomarker.

For the *TestSetAD* dataset, the `PAM` classifier used by Ray et al. achieved 89% accuracy with an MCC of 0.78 (Figure 3.10a). Conversely, in the experimental setup of Ravetti and Moscato, the best prediction performance for this testing dataset was reported as 95% accuracy with MCC of 0.89 using the `IBk` classifier (Figure 3.10c). The best EoC from the proposed GA-EoC outperformed both methods by producing 98% accuracy and an MCC score of 0.96 for the *TestSetAD* dataset (Figure 3.10e). In terms of average performance, GA-EoC also performed much better than the other two methods (Table 3.13) on the

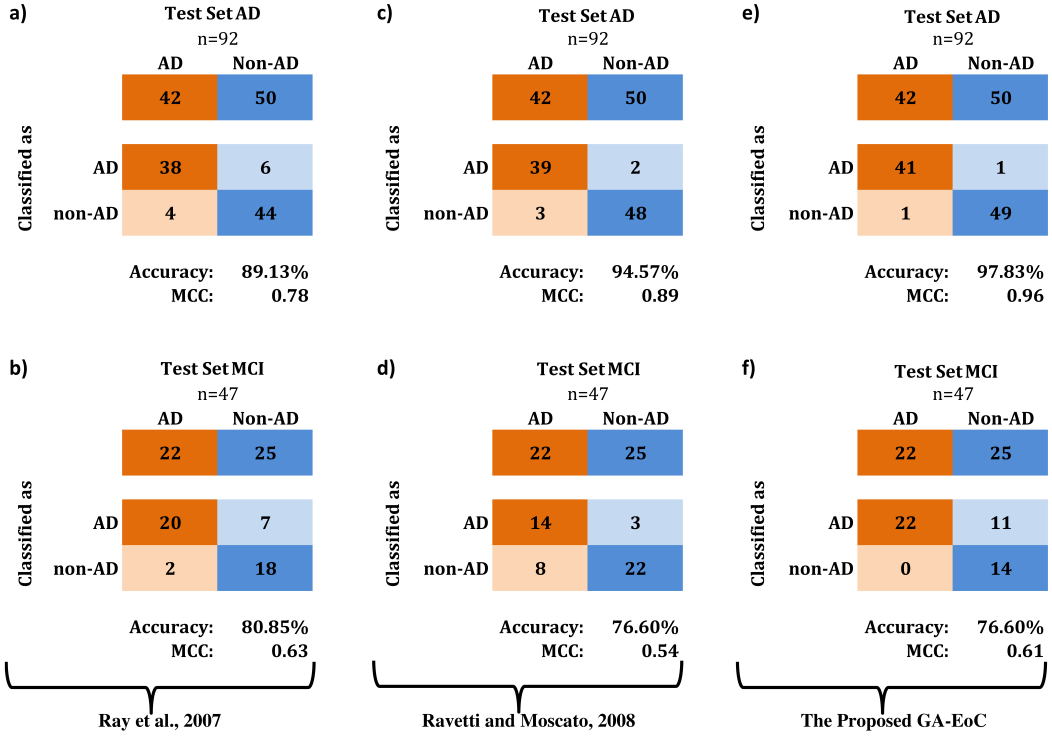


Figure 3.10: Confusion matrices for comparing the best classification performances using an 18-protein biomarker. (a, b) These classification performances were achieved by [Ray et al., 07]. (c, d) These classification performances were achieved by [R. Moscato, 08]. (e, f) These classification performances were achieved by the proposed GA-EoC for *TestSetAD* and *TestSetMCI*, respectively.

same dataset.

For the *TestSetMCI*, neither Ravetti–Moscato's method nor the proposed GA-EoC could exhibit even competitive performance in comparison with the approach of Ray et al. (Figures 3.10b, 3.10d and 3.10f.). The main reason behind the poor performance of GA-EoC and Ravetti–Moscato's method on *TestSetMCI* dataset is that the training dataset did not have any samples from the non-AD class (OD and MCI) [Ravetti and Moscato, 2008]. If the training dataset contained training samples from the non-AD class, the proposed method could have made more correct predictions for non-AD samples.

However, the proposed method classified Alzheimer's with 100% positive agreement (Figure 3.10f) with the follow-up clinical diagnosis, whereas the rate achieved by Ray et al. was 91% (Figure 3.10b). Moreover, the average performance of GA-EoC was better than that from Ravetti–Moscato's method in both scales (Table 3.13).

Next, the 5-protein biomarker discovered by [Ravetti and Moscato, 2008] was used as features for classifying the AD dataset. The average performance of GA-EoC on accuracy

and the MCC scale is compared with that of Ravetti–Moscato's method in Table 3.14. We can observe that the GA-EoC produced a better accuracy (3% more classification accuracy than that reported in [Ravetti and Moscato, 2008]) and MCC (0.06 more than that achieved in [Ravetti and Moscato, 2008]) for prediction in the *TestSetAD* dataset. On the prediction of *TestSetMCI*, the proposed method performed worse than Ravetti–Moscato's method (1.6% less in accuracy and 0.03 less in MCC scores). For better understand-

Test Dataset	**[Ravetti & Moscato, 08]**		**The GA-EoC**		**Gap**	
	Acu	*MCC*	*Acu*	*MCC*	*Acu*	*MCC*
TestSetAD	92.90%	0.86	**95.91%**	**0.92**	3.01%	0.06
TestSetMCI	**64.55%**	**0.30**	62.98%	0.27	-1.57%	-0.03

TABLE 3.14: Average classification performances (in terms of accuracy and MCC) using the 5-protein biomarker.

ing of the detailed performances, we show the confusion matrices of the best performance achieved using Ravetti–Moscato's method and GA-EoC in Fig 3.11. The best classification performance achieved by GA-EoC using the 5-protein biomarker produced 98% accuracy with 0.96 MCC for the *TestSetAD* (Fig 3.11c) and 79% accuracy with 0.62 MCC score for the *TestSetMCI* datasets (Fig 3.11d). The best ensemble combination of GA-EoC produced a better generalisation performance for all of the datasets tested by [Ravetti and Moscato, 2008]. Only for the MCI datasets, the average performance of GA-EoC was slightly poorer compared with the other method. The reason why the GA-EoC failed to perform well in the MCI datasets classification is explained in the previous section.

Finally, if we compare the performance of GA-EoC with that of the base classifiers (Table 3.9 and Table 3.8), then it is found that GA-EoC consistently performed at least as well as its base classifiers in classifying *TestSetAD* using both 18 and 5 biomarkers. But in the case of *TestSetMCI*, the performance of GA-EoC was found to be poor compared with some of its base classifiers (such as `SGD` in Table 3.8 and `LMT`, `Logistic`, `SimpleLogistic` in Table 3.9). As we have explained, this poor performance was due to the biased nature of the dataset.

3.4.3 Performances on the Face-Recognition Dataset

We generated three (03) sets of datasets from the original datasets, as described in Section 3.3.3. Each dataset from the setup contained five (05) sets of training and testing sets of 'one-vs-all' setup, but having different numbers of features. We applied the GA-EoC and the three ensemble methods named `Bagging`, `AdaBoostM1` and `Random Forest` on these datasets and report the average result for each of those datasets.

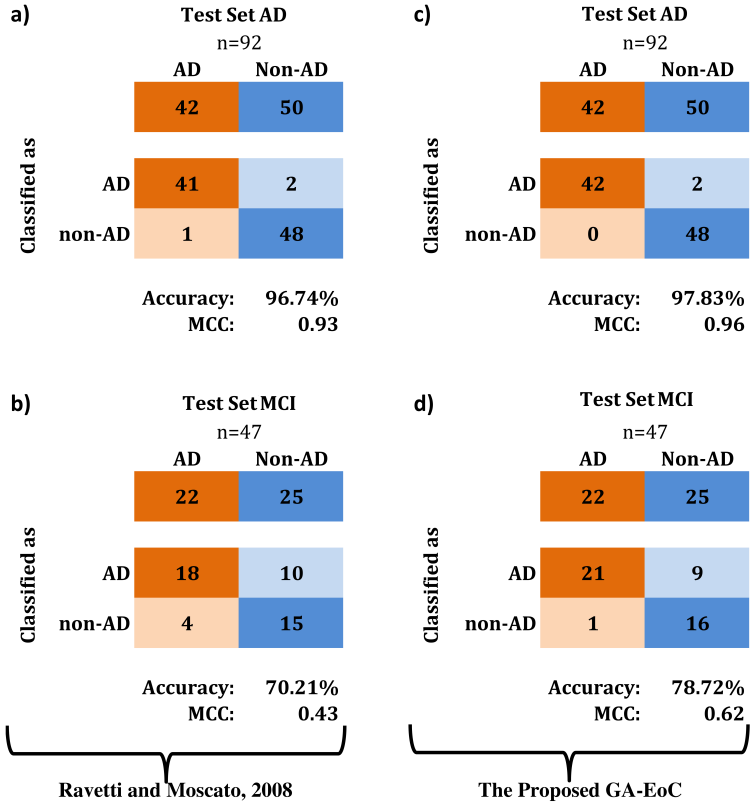


Figure 3.11: Best classification performances by the state-of-the-art method vs the proposed method with the 5-protein biomarker. The comparison of best classification performances using the 5-protein biomarker (RavettiMoscato-AD-Trn-5) as training dataset and *TestSetAD* and *TestSetMCI* as test datasets. (a, b) Classification performances achieved by [R. Moscato, 08], (c, d) Classification performances achieved by GA-EoC for the *TestSetAD* and *TestSetMCI*, respectively.

We have tabulated the classification performance of the proposed method vs three ensemble methods for the *UAB* datasets in the Table 3.15. We see that the GA-EoC performed (in a scale of both MCC and accuracy) much better than the other three EoC methods. In these training datasets, we have $4,700$ features on average. Then, we have applied the GA-EoC for all five binary-class datasets and found the average MCC on all five (5) binary-class datasets as 0.623 with average accuracy of 88.00%. The closest performing ensemble method, `AdaBoostM1`, has an average MCC of 0.387 with accuracy of 82.00%. We can observe the detailed performance of the proposed GA-EoC for all binary-class datasets from the confusion matrices in Figure 3.12.

Then, we used the *IAB* datasets to observe the classification performance of the proposed method with other ensemble methods. These datasets contained $4,500$ features on

Classifier	Precision	Accuracy	F-Measure	MCC
`Bagging`	0.825	83.20%	0.795	0.360
`AdaBoostM1`	0.807	82.00%	0.809	0.387
`Random Forest`	0.803	82.80%	0.799	0.348
`GA-EoC`	**0.886**	**88.40%**	**0.879**	**0.623**

TABLE 3.15: Average performance on the *UAB* datasets, where the average number of features was 4700.

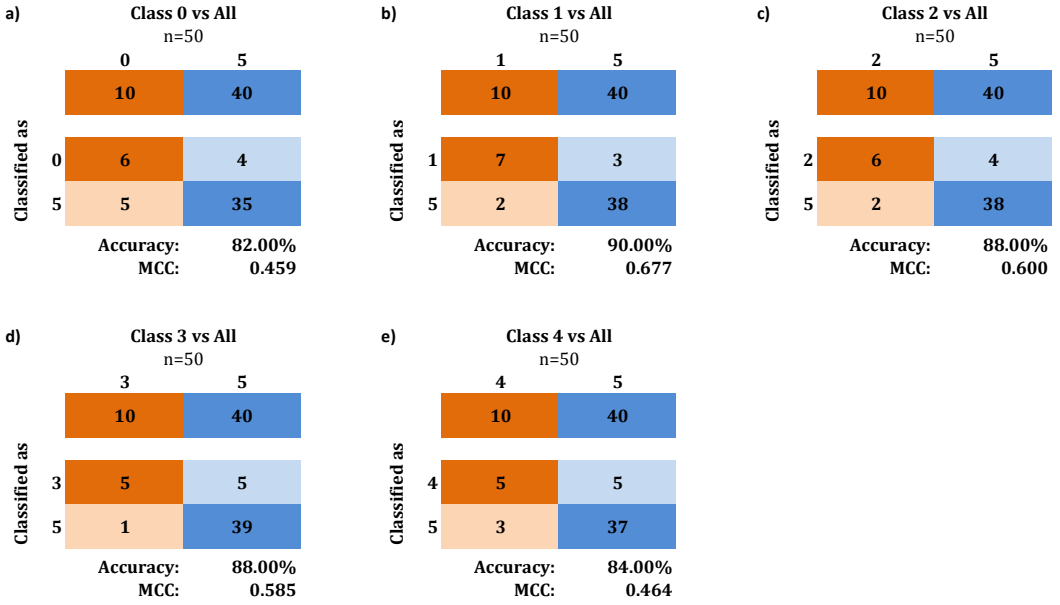


FIGURE 3.12: Confusion matrices showing the classification performances for the *UAB* datasets for 'one-vs-all' setup.

average. For these datasets, the GA-EoC outperformed all ensemble methods used in the experiment (shown in Table 3.16) by achieving 0.565 MCC with an accuracy of 86.80%. The closest ensemble method, the `Random Forest`, produced 0.414 MCC with 84.00% accuracy for these datasets on average. Confusion matrices for all binary-class datasets of the *IAB* are shown in Figure 3.13.

Finally, we executed the experiment for the *UEAB* datasets. These datasets have the least number of features (average number of features was 1700 per binary dataset) compared with the previous two datasets. The classification performance achieved by the GA-EoC using these features was better than for the previous two tests (shown in Table 3.17). The GA-EoC achieved 0.564 MCC with accuracy of 86.80%. The proposed method outperformed other ensemble methods as well. In this case, it used less than one-third of the total features of the *IAB*, but achieved the same MCC values. The

Classifier	Precision	Accuracy	F-Measure	MCC
`Bagging`	0.830	83.60%	0.801	0.379
`AdaBoostM1`	0.802	82.00%	0.805	0.370
`Random Forest`	0.837	84.00%	0.810	0.414
`GA-EoC`	**0.871**	**86.80%**	**0.858**	**0.565**

Table 3.16: Average performance on *IAB* datasets, where the average number of features was 4500.

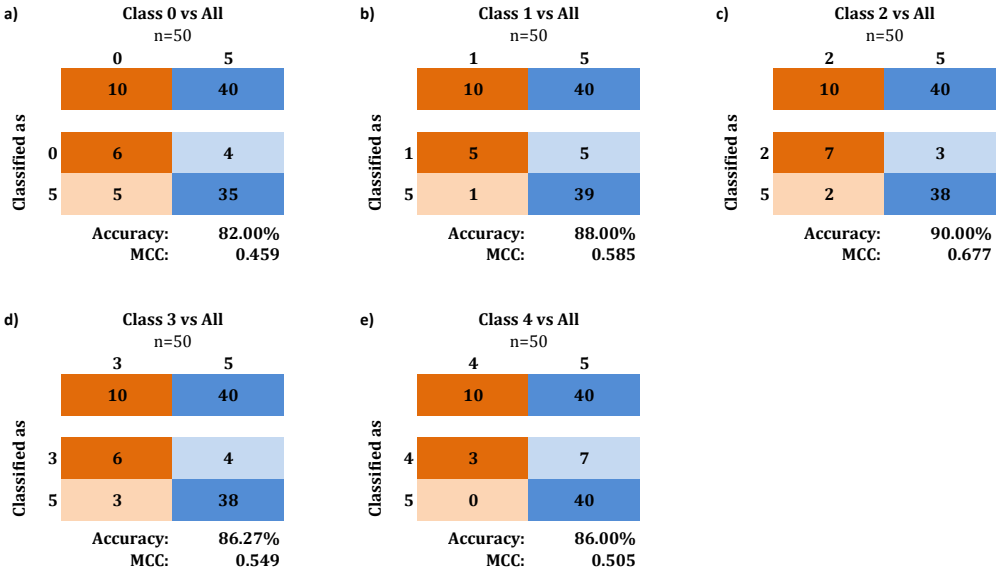


Figure 3.13: Confusion matrices to show the classification performances for the *IAB* datasets for 'one-vs-all' setup.

detailed performances of GA-EoC for the *UEAB* datasets are shown in confusion matrices in Figure 3.14.

Because MCC is the representative performance measure in this work, we analysed the MCC scores achieved by GA-EoC and other ensembles for the face-recognition problem datasets separately in Figure 3.15 (comparisons for other scores could be found in Figure S1 of Appendix E). Among the three dataset configurations, GA-EoC performed best in the UAB setup. The 25^{th} percentile of the MCC score achieved by the GA-EoC is nearly 0.60. In contrast, the `Bagging` classifier's 50^{th} percentile MCC score is below 0.40, which is the closest performing classifier for this configuration. In the IAB configuration, the 50^{th} percentile of MCC is nearly 0.60 for GA-EoC. The closest performing ensemble classifiers, `Bagging` and `Random Forest`'s 50^{th} percentile MCC scores are below 0.50 and 0.40 respectively, which shows that GA-EoC performed far better than other EoCs in this dataset also. The 50^{th} percentile MCC scores for the GA-EoC are above 0.60 for the

Classifier	Precision	Accuracy	F-Measure	MCC
`Bagging`	0.812	83.20%	0.795	0.347
`AdaBoostM1`	0.798	81.60%	0.802	0.357
`Random Forest`	0.836	84.40%	0.830	0.464
`GA-EoC`	**0.862**	**86.80%**	**0.863**	**0.564**

TABLE 3.17: Average performance on *UEAB* datasets, where the average number of features was 1700.

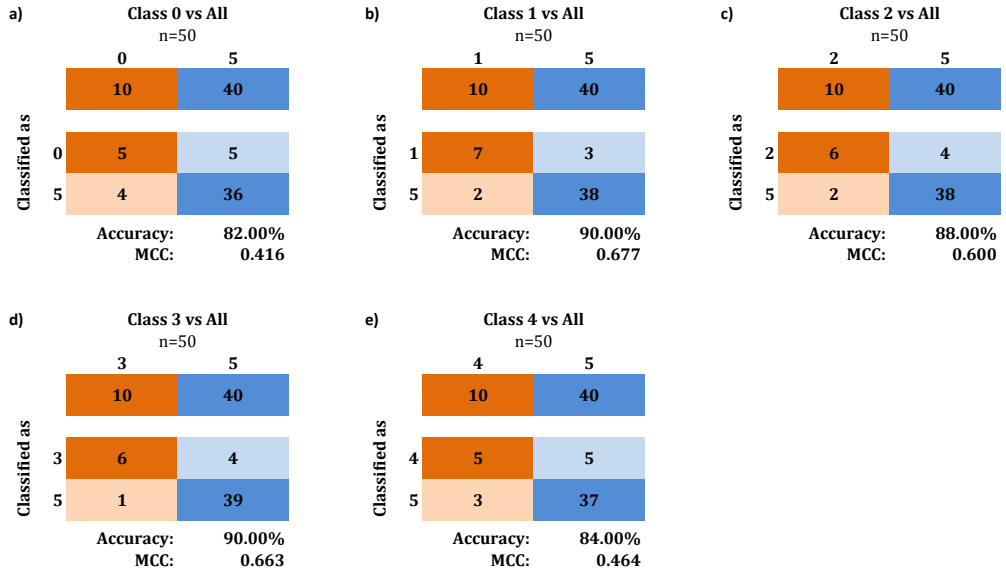


FIGURE 3.14: Confusion matrices to show the classification performances for the *UEAB* datasets for 'one-vs-all' setup.

UEAB configuration. In comparison with that, the rest of the classifier's 75^{th} percentile MCC scores are below 0.50 in the MCC scale. Classification performances (in terms of MCC) of the genetic algorithm-based ensemble of classifiers vs three ensemble methods for the PubFig05 datasets could be found in Figure S2 of Appendix E.

It is also notable that the UEAB-configured datasets contain the least number of features (average number of features was 1700 per binary dataset) among these three configurations. However, the classification performance achieved by GA-EoC using these features is comparable to the previous two tests. It has been observed from the result, the compact and good set of features selected by the (α, β)-k Feature Set selection method helped GA-EoC to achieve a good generalisation performance. Nevertheless, the other ensembles, using the same set of features, could not generate similar performances. These experimental outcomes support the effectiveness of the (α, β)-k Feature Set selection method and qualify it as a candidate for the dimensionality reduction techniques to be used with

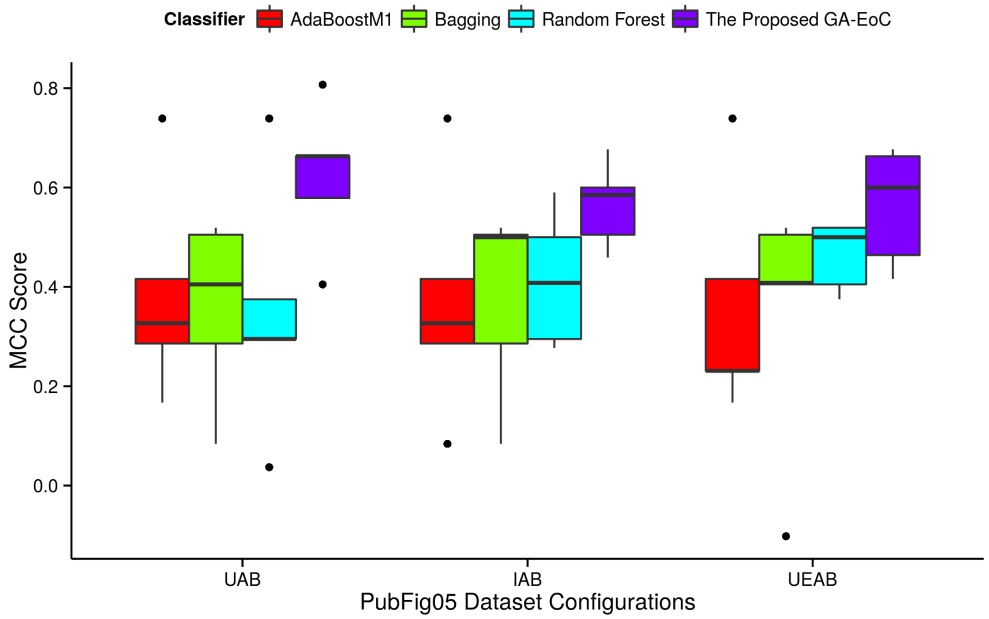


Figure 3.15: The MCC Scores of `AdaBoostM1`, `Bagging`, `Random Forest` and GA-EoC are shown for PubFig05 datasets.

GA-EoC.

3.5 Discussion

3.5.1 Classification Performances of GA-EoC

The performance comparison presented in Table 3.8 shows that there is no single classifier that achieves the best accuracy for all experiments performed with different types of datasets. Among the 20 base classifiers, `LMT` and `SimpleLogistic` were able to achieve the best accuracy for four experiments, which includes both MCI experiments. The `SGD` and `Logistic` were able to produce the best classification accuracy for three (one MCI experiment included) and two (both MCI experiments included) experiments, respectively. GA-EoC outperformed the best accuracies of all base classifiers for five experiments. Moreover, the average accuracies of GA-EoC are also close (3% less than the best accuracy for other experiments) to the best classification accuracies for other experiments.

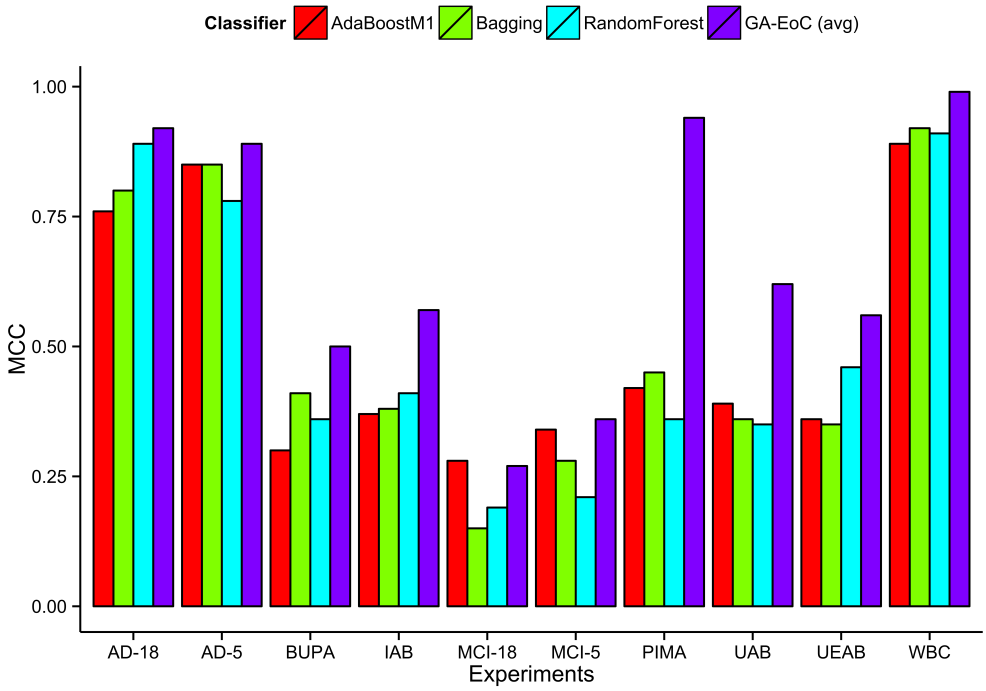


FIGURE 3.16: Comparison of MCC scores achieved by genetic algorithm-based ensemble of classifiers and other ensemble of classifiers (`AdaBoostM1`, `Bagging` and `Boosting`) for all experiments.

If we consider the MCC scores presented in Table 3.8, only the `SGD` classifier was able to achieve the best MCC for four experiments, which included two MCI experiments. For the other experiments, the best MCC is achieved by each of the four different base classifiers

in one experiment. The average MCC value of GA-EoC outperformed the best base classifier's MCC value for five experiments. Moreover, for the other three experiments, the average MCC scores of GA-EoC are close to the best performing base classifier. Therefore, summarising all the results from Table 3.8 and Table 3.9, it can be concluded that the average performance of GA-EoC, over a variety of classes of datasets, is better than that of any single base classifier. This is further illustrated in Figure 3.16.

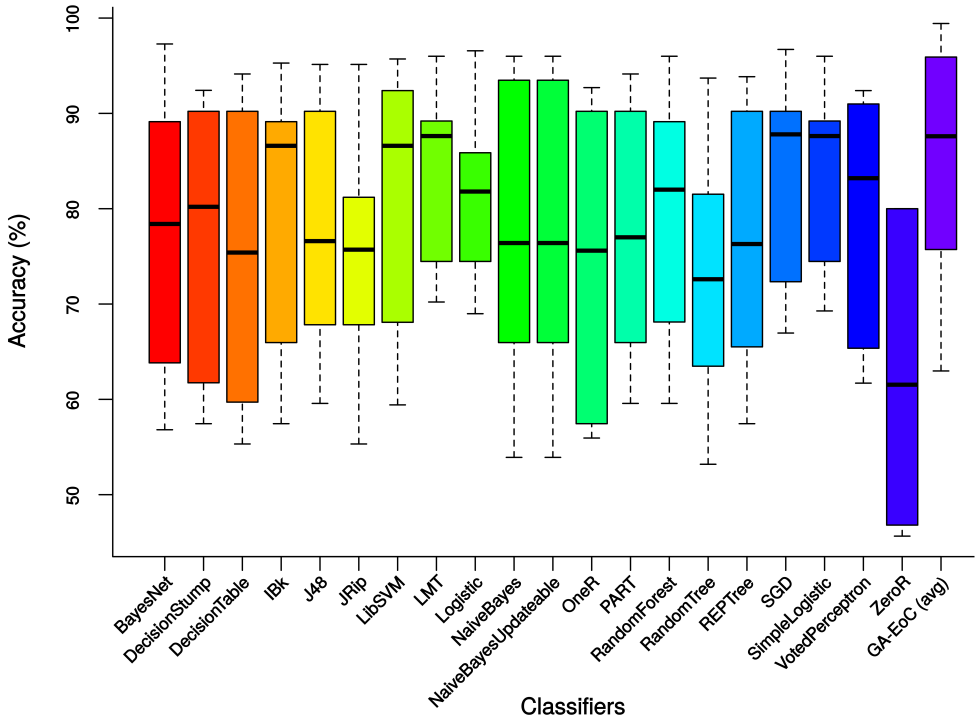


Figure 3.17: The accuracies of base classifiers and average accuracies of genetic algorithm-based ensemble of classifiers over all experiments.

The box plots in Figure 3.17 compare the average classification accuracy achieved by the proposed GA-EoC and the base classifiers over all experiments. From the box plot, it is clear that the median or 50^{th} percentile accuracy of GA-EoC is similar to accuracies achieved by `LMT`, `SGD` and `SimpleLogistic` classifiers. But the 75^{th} percentile accuracy of the GA-EoC is higher than for all of them. In terms of accuracy, GA-EoC performed better than the base classifiers considering all test cases. The box plot in Figure 3.18 shows the classification performances of base classifiers and the average performance of the GA-EoC for all datasets using the MCC scale. The median of the MCCs achieved by the GA-EoC is similar to MCCs of `LMT`, `SGD` and `SimpleLogistic` classifiers. However, the 75^{th} percentile MCC score of GA-EoC is once again found to be better than any of the

base classifiers, considering all experiments. These results show the robustness of GA-EoC as a single method compared with other base classifiers.

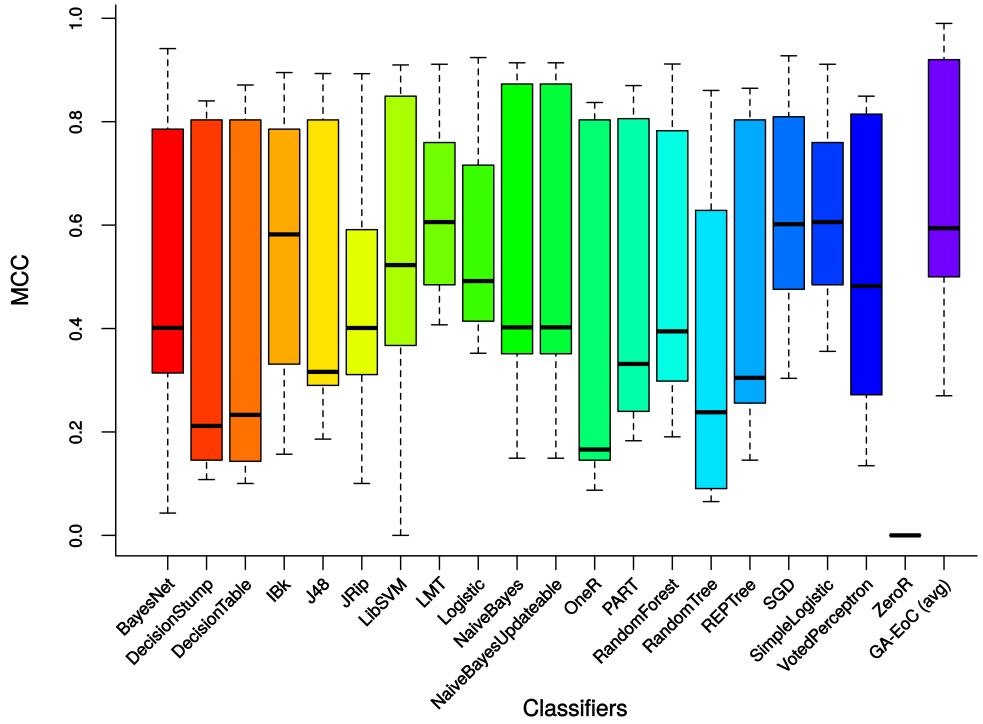


FIGURE 3.18: The MCC scores of base classifiers and the average MCC scores of genetic algorithm-based ensemble of classifiers over all experiments.

3.5.2 Base Classifiers Combination in GA-EoC

Next, we investigate the types of base classifiers selected in the GA-EoC ensemble pool in different experimental runs. In Table 3.18 we list the common base classifiers included in the ensembles constructed by GA-EoC for all experiments. The proposed method selected only seven different combinations of base classifiers over 100 runs in the PIMA dataset experiment. We found that the `DecisionStump`, `IBk`, `RandomForest`, and `RandomTree` classifiers are common among all ensembles. None of these base classifier's MCC score was over 0.45, but their ensemble produced an average MCC score of 0.94. Moreover, in terms of MCC, the `SGD` classifier was the best base classifier for the PIMA dataset, but it did not appear in any of the best ensembles selected by the GA-EoC. Analysis of the population revealed that although `SGD` appeared in individuals in early generations, it did not survive due to low fitness in the GA. For an example, `01010010000001101000` is one

of the individuals that contains the SGD classifier and produces an MCC of 0.9153, which is lower than the optimised fitness value. From this observation, it becomes clear that ensembles created with the best base classifiers do not always produce a better MCC, but a combination of diverse base classifiers could produce better outcomes. In other datasets, we also found that there were only a few different ensemble combinations in which GA-EoC converged over repeated runs. This observation supports our claim that GA-EoC is consistent in generating ensembles.

Training Dataset	Common Base Classifiers	#Ensm
BUPA	JRip, RandomTree, SGD	6
PIMA	DecisionStump, IBk, RandomForest, RandomTree	7
WBC	JRip, LibSVM, SGD	8
RMoscato-AD-Trn-5	JRip, LibSVM	6
Ray-AD-Trn-18	JRip	20
IAB	IBk, JRip, Logistic, PART, SGD, VotedPerceptron	5
UAB	IBk, Logistic, RandomTree	5
UEAB	IBk, Logistic, RandomTree	5

Table 3.18: The number of different ensembles (with common base classifiers in them) constructed by genetic algorithm-based ensemble of classifiers over repeated experimental runs.

Table 3.19 shows the percentage of times a base classifier appeared in ensemble combinations selected by the GA-EoC over repeated experimental runs. We observe that the IBk, RandomForest and RandomTree are the classifiers who appeared in over 90% of the ensemble combinations. In contrast, DecisionTable and LMT are the classifiers who never appeared in any of the ensemble combinations.

3.5.3 Comparison of Ensemble of Classifiers and Genetic Algorithm-Based Ensemble of Classifiers

Finally, we compared GA-EoC with other common EoCs (Bagging, AdaBoostM1, RandomForest, RandomCommittee and Stacking), over all experimental datasets. We used the default parameter settings for those ensembles of classifier algorithms available in the WEKA framework. The classification performances achieved by other ensemble methods and GA-EoC are shown in Table 3.20. The average accuracy of GA-EoC is better than those classifiers for all test cases. In terms of MCC score, AdaBoostM1 marginally outperformed GA-EoC only for the MCI-5 experiment and RandomCommittee outperformed GA-EoC for MCI-18 dataset. In other experiments, GA-EoC achieved better average MCC scores than the other ensembles. Based on these results, we claim that GA-EoC is a better choice than many other EoCs like Bagging, AdaBoostM1, Stacking and Random Forest for imbalanced class datasets.

Base Classifier	Frequency (in %)
BayesNet	8.52
DecisionStump	5.46
DecisionTable	0.00
IBk	98.47
J48	30.35
JRip	32.75
LibSVM	57.21
LMT	0.00
Logistic	4.80
NaiveBayes	21.18
NaiveBayesUpdateable	8.95
OneR	1.31
PART	15.28
RandomForest	94.98
RandomTree	99.34
REPTree	10.92
SGD	0.44
SimpleLogistic	4.15
VotedPerceptron	0.22
ZeroR	30.35

TABLE 3.19: The frequency (in percentage) of base classifiers appearance in the ensemble of classifiers selected by GA-EoC over repeated experimental runs.

3.5.4 Convergence Analysis of GA-EoC

While designing the proposed GA-based EoC method, we fixed three conditions in Section Terminating Conditions. We determined the maximum fitness achieved by each generation and plotted it in Figure 3.19 for the AD datasets. The algorithm converged within less than 100 generations for the training dataset named *Ray-AD-Trn-18*. It satisfied the terminating condition of 50 repeated fitness values and converged. We observed the satisfied terminating conditions for other datasets by the proposed algorithm. We noticed that, in most of the cases, the best fitness value in populations remained stationary for 50 consecutive generations, and therefore fulfils one terminating requirement for the GA.

3.5.5 Running Time of GA-EoC

To measure the running time of the proposed method, we needed to consider the base classifier models' building time, consider ensemble combination model evaluation, search for best combination time and the best prediction time. Now, we will discuss in detail the

Dataset	AdaBoostM1		Bagging		RandomForest		RandomCommittee		Stacking		GA-EoC (avg)	
	Acc	*MCC*	*Acc*	*MCC*	*Acc*	*MCC*	*Acc*	*MCC*	*Acc*	*MCC*	*Acc*	*MCC*
WBC	95.14	0.89	96.28	0.92	95.99	0.91	95.28	0.90	65.52	0.00	**99.43**	**0.99**
PIMA	74.35	0.42	75.78	0.45	72.14	0.36	75.26	0.44	65.10	0.00	**97.43**	**0.94**
BUPA	66.96	0.30	71.88	0.41	68.12	0.36	69.57	0.38	57.95	0.00	**75.72**	**0.50**
AD-18	92.39	0.85	92.39	0.85	89.13	0.78	84.78	0.70	45.64	0.00	**94.66**	**0.89**
MCI-18	65.96	0.34	63.83	0.28	59.57	0.21	**68.09**	**0.38**	46.80	0.00	67.14	0.36
AD-5	86.96	0.76	90.22	0.80	94.57	0.89	85.87	0.73	45.64	0.00	**95.91**	**0.92**
MCI-5	59.57	**0.28**	57.45	0.15	59.57	0.19	59.57	0.19	46.81	0.00	**62.98**	0.27
UAB	82.00	0.39	83.20	0.36	82.80	0.35	84.00	0.43	80.00	0.00	**88.40**	**0.62**
IAB	82.00	0.37	83.60	0.38	84.00	0.41	84.00	0.56	80.00	0.00	**86.80**	**0.57**
UEAB	81.60	0.36	83.20	0.35	84.40	0.46	83.60	0.44	80.00	0.00	**86.80**	**0.56**

Table 3.20: Classification performances of common ensemble of classifiers vs genetic algorithm-based ensemble of classifiers for all experiments.

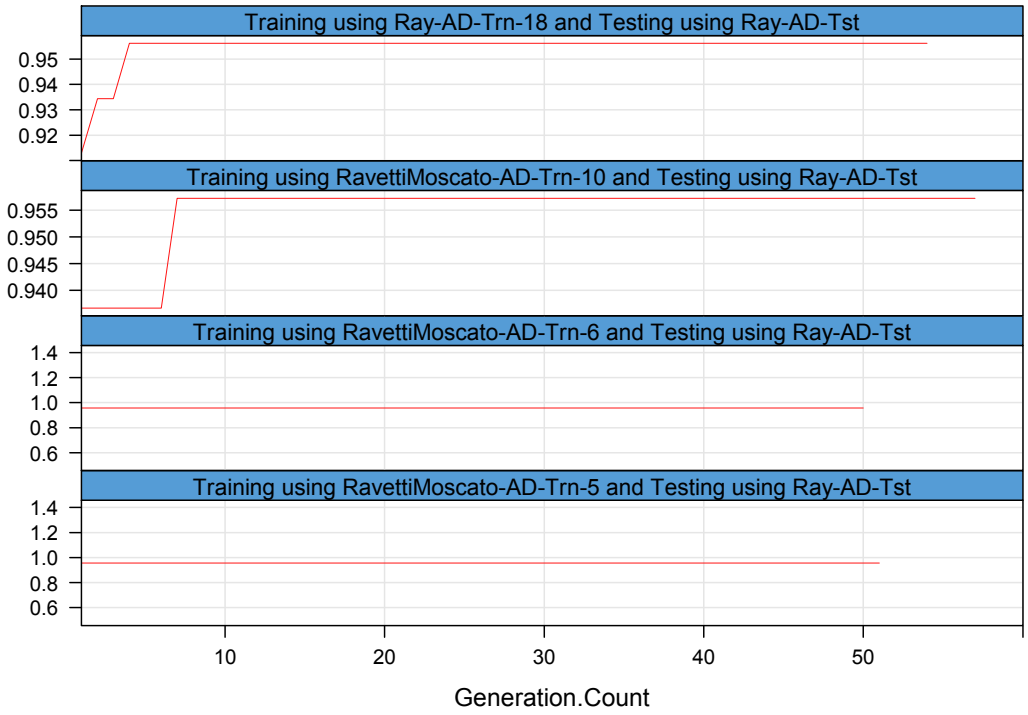


FIGURE 3.19: Convergence of the genetic algorithm for AD datasets.

central processing unit (CPU) time requirements for all of these processes.

To use a classifier for prediction, first, it needs to create a model using the testing dataset. Then, it uses that model to predict the class labels for unknown test datasets. Classifiers require more time to build the model than to use it for prediction. The training time or model building time depends on the training set and types of classifier. To evaluate the training performance, it is convenient to use the k-fold CV method. The k-fold CV method requires creating k different models on k different folds of the training dataset. For a conventional classifier, creating k models on a single training dataset increases the running time required for model building k times. We used the 10-fold CV method for evaluating the fitness of an ensemble combination. The proposed method uses more than one classifier, so it required more training time than conventional classifiers. To reduce the training time, we created the 10-fold CV models in parallel. This helped to reduce consolidated training time for the EoC.

We then considered the model creation and evaluation time for a specific combination. This involves creating a combined model using participating single classifiers for each fold of the combination. While creating the model, we also needed to evaluate the performances

of the ensemble combination under the 10-fold CV setting. We need to evaluate such combinations equal to the number of populations used in each generation of the GA. Next, we searched this population to find the best individual based on 10-fold CV evaluation. This population evaluation task was another time consuming process in the proposed method. It could be a suitable candidate for parallelism to reduce the running time.

Dataset	Worst	Average ± Std Dev	Best
WBC	13:10	8:55 ± 3:24	0:18
PIMA	17:51	12:52 ± 4:25	0:26
AD-5	3:30	2:18 ± 0:53	0:03
MCI-5	3:26	1:54 ± 1:11	0:03

Table 3.21: Running time statistics (in minutes:seconds) of the genetic algorithm-based ensemble of classifiers on different datasets.

After the GA converged, we received an ensemble combination as the fittest individual based on training performance. Then, we need to build that ensemble model to use it for future prediction on the testing dataset. The ensemble model building time was negligible compared with the time required for the whole process. However, it needed to build and combine models for participating base classifiers.

We tabulated the average running time (consolidated running times required for training folds evaluation for all generations, ensemble combination building and evaluating times for each run) required for different experiments in Table 3.21 for 100 runs. In the table, we observe that for the PIMA dataset, the GA-EoC required more running time when compared with others. Conversely, to process the AD dataset, the proposed method required the least CPU time. It is notable that for some generations, the process reached maximum fitness (training MCC=1.0) and converged before reaching the maximum allowed generation count.

3.6 Conclusion

This chapter presents a GA-based search method, named GA-EoC, for constructing heterogeneous EoCs. As the number of base classifiers increases, the number of possible ensembles that can be created rises exponentially. Because an exhaustive search for constructing the best ensemble is not feasible, we propose a GA for searching the best combination of base classifiers for constructing the ensemble. GA-EoC employs the majority-voting technique for combining the base classifier's decisions in a single final decision.

Imbalanced class distributions in many real-world datasets have become a significant challenge for a classifier's performance. In the case of imbalanced datasets, the GA-EoC,

as a preprocessing step, creates several balanced datasets depending on the imbalance ratio between two classes of the imbalanced dataset. We have applied the (α, β)-k Feature Set selection method for the datasets where no feature set was given *a priori* for classification. GA-EoC generates the models of base classifiers using 10-fold CV on these balanced training datasets and then reuses them while evaluating different combinations of heterogeneous ensembles.

The performance of GA-EoC has been evaluated using various datasets selected from the UCI-ML repository and other sources. These experimental results suggest that the ensembles constructed by GA-EoC are better than a single base classifier in general. Moreover, the ensembles constructed by GA-EoC were found to be better than those constructed by many established ensemble construction methods.

In this chapter, GA-EoC has been studied in a simple setting that can be improved and extended in many ways. For example, better classification accuracy can be achieved by fine-tuning the parameters (e.g., rule weights, membership functions) of base classifiers or utilising other fusion approaches [Kuncheva, 2004]. In the current implementation of the GA-EoC, a single-objective GA is used to optimise the MCC score of the ensemble. The work can be extended by using multi-objective GAs to handle more challenging and complex classification problems. Incorporation of such advanced and efficient components can improve the generalisation performance of GA-EoC and be more capable of handling the class-imbalanced and dimensionality challenges in modern datasets.

This chapter contains parts of the paper by M. N. Haque, M. N. Noman, R. Berretta and P. Moscato, "Optimising weights for heterogeneous ensemble of classifiers with differential evolution," 2016 IEEE Congress on Evolutionary Computation (CEC), Vancouver, BC, 2016, pp. 233-240, URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7743800&isnumber=7743769.

4

Differential Evolution in Weighted Voting Ensemble of Classifiers

The ensemble of classifiers (EoC) utilises decisions from multiple base classifiers to reduce classification error. Outputs from multiple classifiers need to be merged into a single decision for the EoC. Accumulation of the voting by base classifiers is one of the most widely used and simplest approaches to decision fusion. In the case of majority voting, a vote from each base classifier is treated as equal, despite the non-identical classification performances [Mao et al., 2015]. Conversely, in the case of a weighted-voting approach, the votes from base classifiers are weighted differently in the decision fusion process. Moreover, empirical studies on decision fusion approaches have revealed that weighted voting could significantly improve the classification performances [Bauer and Kohavi, 1999, Sun et al., 2005, Wozniak and Jackowski, 2009].

Storn and Price[Storn and Price, 1997, Price, 1999] introduced Differential Evolution (DE) in 1995 for numerical optimisation. The DE method works similarly to the genetic algorithms approach and is summarised in the Algorithm 5. DE 'evolve' like the same way of genetic algorithm evolves. It allows each successive generation of solutions to evolve from previous generation similar as GA does. DE also has the genetic operation of selection, crossover and mutation like any other evolutionary algorithm. The main advantage

of the DE is that it can be applied to real-valued problems over a continuous space with much more ease than a GA. The DE can be easily applied to a wide variety of real-valued problems despite noisy, multi-modal, multi-dimensional spaces, which usually make the problems very difficult for optimisation [Hegerty et al., 2009]. Another remarkable characteristic of DE is that the parameters do not require the same fine tuning which is necessary in many other evolutionary algorithms [Mezura-Montes, 2008]. DE has been effectively used in many application on various domains (such as neural network learning, digital signal processing, and image processing) where real-valued optimisation is necessary [Ilonen et al., 2003, Karaboga, 2005, Cuevas et al., 2016].

Algorithm 5: DIFFERENTIAL EVOLUTION ALGORITHM

```
Generate randomly an initial population of solutions.
Calculate the fitness of the initial population.
repeat
    foreach parent of the Population do
        select three solutions at random.
        Create one offspring using the DE operators.
    end
    foreach member of the next generation do
        if offspring(x) is more fit than parent(x) then
            Parent(x) is replaced.
        end
    end
until condition_terminate ≡ satisfied;
```

Different approaches have been utilised to adjust the weights of base classifiers in EoCs so far. Use of different types of discrimination measures [Bashir et al., 2015c, Kuncheva and Rodríguez, 2014, Wozniak, 2008], linear programming [Zhang and Zhou, 2011], dynamic adjustment [Valdovinos and Sánchez, 2009] and game theory [Georgiou et al., 2006] are some examples of weight adjustment approaches found in the literature. Several EAs have been proposed as well. Genetic Algorithm (GA) [Ekbal and Saha, 2011, Liu et al., 2014, Maghsoudi et al., 2006] and Differential Evolution (DE) [Bhadra et al., 2012, Zhang et al., 2014] algorithms were successfully employed in weight optimisation. Maghsoudia et al. [Maghsoudi et al., 2006] proposed a GA to adjust the weights of base classifiers in homogeneous ensembles. The weight was adjusted for the overall accuracy of each class in random subspaces of the dataset. Bhadra, Bandyopadhyay and Maulik [Bhadra et al., 2012] proposed a DE to optimise weights of a homogeneous EoC. They used a combined fitness function with different classifier performance measures and tested it on three

datasets. EAs have also been employed for weight optimisation in the small-scale Heterogeneous EoC (HEoC). For instance, Ekbal and Saha [Ekbal and Saha, 2011] proposed a GA for optimising the weights of HEoCs for the named-entity recognition problem. Each base classifier received separate weight per class labels based on the F-measure score. They created several instances of base classifiers varying the training features. Liu et al. [Liu et al., 2014] used a GA for weight optimisation of a vote-based extreme learning machine. Optimised weights were used to form the ensemble of neural network classifiers. Zhang et al. [Zhang et al., 2014] proposed a DE algorithm for optimising weights of five base classifiers for voting in an EoC. They used accuracy as the fitness score to optimise the weight of base classifiers. These applications of EAs demonstrate the potential in optimising base classifiers' weights in the vote-based EoC. Most of the experiments performed for weighted voting in the literature were for homogeneous EoCs. A very few HEoCs have been proposed, but they were formed with a small number of base classifiers. Hence, weighted-vote-based ensemble creation using a large number of heterogeneous base classifiers needs to be explored for its suitability and robustness.

In this chapter, we have adopted an approach to optimise the weight and to select the best combination of base classifiers for creating the HEoC using a DE algorithm [Storn and Price, 1997]. DE is a simple but efficient and robust EA for optimisation of real-valued parameters. The inherent advantages of DE will facilitate to optimise the associated weights of base classifiers in HEoC. This powerful optimisation algorithm is used with the MCC score [Matthews, 1975] as the fitness value to determine the optimal weights for base classifiers used in HEoC. The algorithm finds the best combination of base classifiers alongside optimising their weights. We use maximisation of MCC as the objective of the DE because it gives a more balanced measure of the generalisation performances of a classifier than other popular measures (such as accuracy, precision and recall) [Dutt and Madan, 2012]. We propose the DE-HEoC algorithm, which optimises the weights of base classifiers in a HEoC using a DE to create the vote-based EoC.

4.1 The Differential Evolution

DE is much simpler and easier to implement in any programming language compared with many other EAs. Despite its simplicity, DE exhibited remarkable performance in solving a wide variety of real-world problems in a reasonable amount of computation time [Das and Suganthan, 2011].

DE is a parallel direct search method for optimising D-dimensional real-valued parameters. It starts with a randomly generated initial population of NP number of individuals

and evaluates their fitness. The i-th individual (parameter vector) in the population for the current generation G is given by Equation (4.1):

$$\overrightarrow{X}_{i,G} = [x_{1,i,G}, x_{2,i,G}, \cdots, x_{D,i,G}], \tag{4.1}$$

where $i = 1, 2, \cdots, NP$ and $G = 0, 1, 2, \cdots, G_{max}$ is the generation number. Each of the parameter values are usually bounded by lower and upper limits. The DE utilises the same computational framework that is used by other standard EAs, and therefore three operators, namely mutation, recombination and selection, are used to create a new generation of individuals. This process is repeated until the stopping criterion is satisfied. Brief descriptions of them are presented below.

Mutation: After initialisation of the population, DE creates a donor vector $\overrightarrow{V}_{i,G}$ for each population member or target vector $\overrightarrow{X}_{i,G}$ in the current population using the mutation operation. Storn and Price proposed a couple of alternative mutation and recombination strategies for DE [Price, 1999, Storn and Price, 1997]. In this work, we used the $DE/rand/1$ mutation strategy, which is given by Equation (4.2).

$$DE/rand/1 \ : \overrightarrow{V}_{i,G} = \overrightarrow{X}_{r_1^i,G} + F.(\overrightarrow{X}_{r_2^i,G} - \overrightarrow{X}_{r_3^i,G}) \tag{4.2}$$

Here, r_1^i, r_2^i and r_3^i are mutually exclusive integer indices randomly chosen from the population for individual $\overrightarrow{X}_{i,G}$. The amplification factor F controls the scaling of the difference vectors.

Recombination: DE uses the recombination operation to generate the trial vector $\overrightarrow{U}_{i,G}$ from the donor vector and the target vector. In general, two types of recombination operators are used in classical DE, namely exponential and binomial recombination. The DE variant used in this work uses binomial recombination (bin). For each j-th variable from D dimensions, the binomial recombination operates as Equation (4.3):

$$u_{j,i,G} = \begin{cases} v_{j,i,G}, & if\ (rand_{j,i}(0,1) \leq\ C_r)\ \ or\ j = I_{rand} \\ x_{j,i,G}, & otherwise \end{cases} \tag{4.3}$$

The recombination probability $0 \leq C_r \leq 1$ is a user-defined parameter to control the fraction of parameter values that are copied from the donor vector. I_{rand} is a random integer from $\{1, 2, \cdots, D\}$ to ensure that at least one variable is copied from the donor vector $\overrightarrow{V}_{i,G}$. This is called binomial recombination, because the number of inherited parameters from the donor has almost the binomial distribution.

Selection: The selection operator determines whether a target or the corresponding trial vector survives to the next generation. The selection is made as Equation (4.4).

$$\overrightarrow{X}_{i,G+1} = \begin{cases} \overrightarrow{U}_{i,G}, & if\left(Obj\left(\overrightarrow{U}_{i,G}\right) \geq Obj\left(\overrightarrow{X}_{i,G}\right)\right) \\ \overrightarrow{X}_{i,G}, & otherwise \end{cases} \tag{4.4}$$

where, $Obj\left(\overrightarrow{X}_{i,G}\right)$ is the objective function to be maximised. Therefore, the new trial vector $\overrightarrow{U}_{i,G}$ promoted to the next generation only if it produces better or equal objective score compared to the objective score of the target vector.

4.2 Weighted Voting Ensemble of Classifiers

Here, we define the weighted-voting ensemble $\mathbb{E}_{wv}(S)$ for a binary-classification problem. Assume that class labels $\Omega = [\omega_1, \omega_2, \cdots, \omega_k]^T$ for an unknown sample S are given by k base classifiers $L = [\mathbb{C}_1, \mathbb{C}_2, \cdots, \mathbb{C}_k]^T$, where $\omega_i \in \{0, 1\}$ and $i = 1, 2, \cdots, k$. Each base classifier $\mathbb{C}_i$ is associated with one weight $0 \leq x_i \leq 1$ encoded as a parameter of the DE individual.

From the weighted-voting ensemble, total weights for class label 0 (w^0) is calculated as

$$w^0 \leftarrow \sum_{i=1}^{k} x_i, \text{when } \omega_i = 0 \tag{4.5}$$

and total weight for class label 1 (w^1) is calculated as

$$w^1 \leftarrow \sum_{i=1}^{k} x_i, \text{when } \omega_i = 1. \tag{4.6}$$

The weighted-voting ensemble decides the class label of S using w^0 and w^1 as Equation (4.7).

$$\mathbb{E}_{wv}(S) = \begin{cases} 0, & if\left(w^0 > w^1\right) \\ 1, & if\left(w^0 < w^1\right) \\ Rand\,(0,1), & otherwise. \end{cases} \tag{4.7}$$

Here, the class decision goes for the maximum weight gaining class label. The class label is randomly selected in the case of a tie.

4.3 The Proposed DE-HEoC

We employ DE to optimise the weights of base classifiers in the vote-based HEoC. First, we discuss how the EoC is formed, and then we explain the weighted-voting approach. Finally, we discuss the DE algorithm used for weight optimisation.

We created an EoC using the 20 heterogeneous classifiers listed in Figure 4.1. We have taken diverse types of commonly used base classifiers from the WEKA data mining software suite [Hall et al., 2009]. Each base classifier is associated with a weight. Each classifier in the ensemble casts a weighted vote for deciding the class label (cl) of a data sample S. The class label with higher total weighted score becomes the class label for the sample.

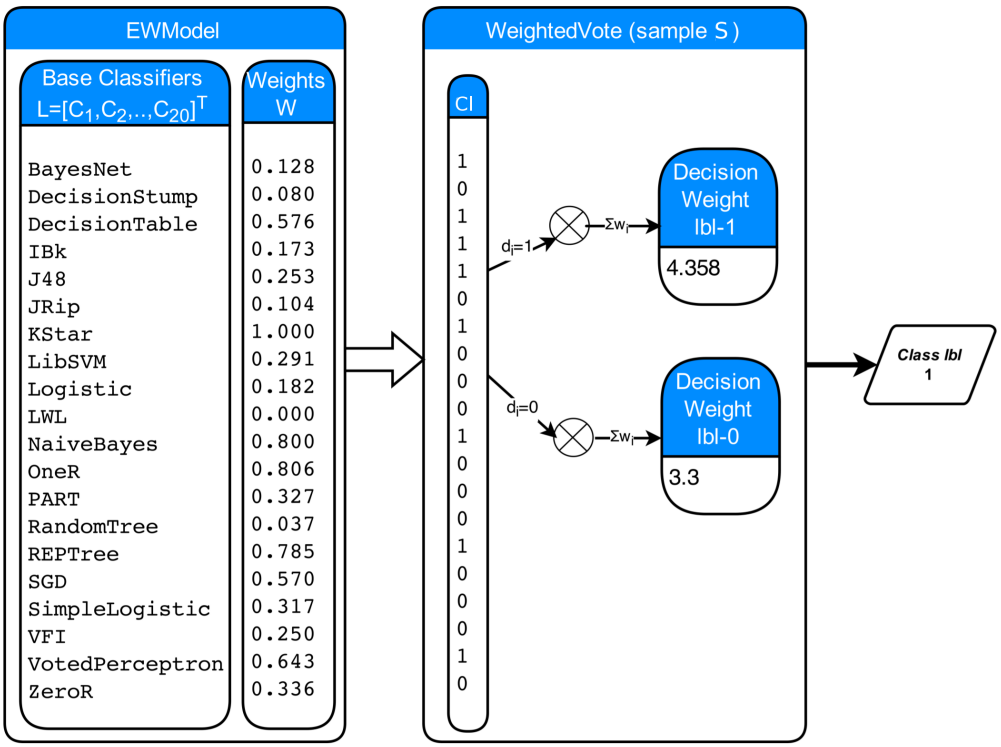


FIGURE 4.1: An example showing the process of deciding the class label of a sample (individual evaluation process) from the weighted voting of an ensemble of heterogeneous classifiers.

4.3.1 Differential Evolution for Weight Optimisation

We used $DE/rand/1/bin$ for optimising the value of the 20 parameters. Each parameter in the individuals of DE corresponds to the weight of one heterogeneous base classifier.

The objective in the DE-HEoC is to maximise the MCC score (see Equation (1.9)) of HEoC. The MCC score quantifies the strength of classification considering both the true positive rate and the true negative rate. A higher MCC score indicates better predictions. The experimental results in [Jurman et al., 2012a] indicate it is an ideal measure for analysis of the confusion matrix. Thus, we consider the MCC our measure of classification performance.

The working process of `DE-HEoC` is shown in Algorithm 6, which returns the best weighted-voting ensemble ($\mathbb{E}_{wv^*}$) for the training data. For internal validation purpose, we created 10 train-fold data (TD) and 10 validation-fold data (VD) from the full training data. Each base classifier in the pool L is trained on each fold of TD and saved as trained model (TM) for future use. Individuals in the population are evaluated for fitness (lines 3–5) by calculating the average classification performances in the MCC score on validation-fold data VD. The fitness value (fit), calculated by Algorithm 7, for each individual is stored within the individual and accessible by the objective function `Obj()`. The population generation and evaluation with DE is repeated (lines 6–22) until the stopping condition is satisfied. The `DE-HEoC` returns the optimised weights of base classifiers in the ensemble that produced the maximum MCC score for 10-fold CV on the training data.

The process `FitnessEvaluation()` for an individual is shown in Algorithm 7. Here, we created one weighted-vote ensemble ($EWModel$) for each fold (f). We used base classifiers weights (W) and pretrained base classifiers (TM) on respective train-fold data to formulate the ensemble. This weighted-voting ensemble is evaluated for MCC on respective validation-fold data (f-th fold of VD). Each sample in the validation-fold data is classified by the weighted-majority-voting ensemble according to the process in Figure 4.1. The average MCC on 10 validation-folds (using the evaluate() function in line 11 of Algorithm 7) is used as the fitness score (fit) of an individual and returned by the `FitnessEvaluation()` process.

The process `GetBestSolution()` is shown in Algorithm 8. It compares the fitness score (line 3) of each individual in population (Pop) and saves the best individual ($\overrightarrow{Best}_{i,G}$). Finally, it returns the best individual in the population having the maximum fitness score.

All parameter values used for the DE-HEoC are shown in Table 4.1. We use the maximum evaluation threshold as the stopping criterion in DE-HEoC. The algorithm returns the optimised weights of base classifiers. Now, we need to use that weighted-voting ensemble to validate the generalisation performance on unknown testing data.

Algorithm 6: The DE-HEoC Algorithm

Input: NP, D, F, R_χ, TM, VD
Output: $\mathbb{E}_{wv^*}$

$G \leftarrow 0$
$Pop \leftarrow$ `InitialisePopulation(`NP, D`)`
`//Calculate Fitness Values of All Individuals in the Initial Population`
for $\overrightarrow{X}_{i,G} \in Pop$ **do**
    $Pop.fit[i] \leftarrow$ `FitnessEvaluation(`$\overrightarrow{X}_{i,G}, TM, VD$, D`)`
while $\neg$ `StopCondition()` **do**
    $NewPop \leftarrow \phi$
    for $\overrightarrow{X}_{i,G} \in Pop$ **do**
        `//Randomly chose three mutually exclusive parents`
        $\overrightarrow{X}_{r_1^i,G} \leftarrow$ `RandomMember(`Pop`)`
        $\overrightarrow{X}_{r_2^i,G} \leftarrow$ `RandomMember(`Pop`)`
        $\overrightarrow{X}_{r_3^i,G} \leftarrow$ `RandomMember(`Pop`)`
        `//Mutation Operation`
        $\overrightarrow{V}_{i,G} \leftarrow$ `DE/rand/1(`$\overrightarrow{X}_{r_1^i,G}$, $\overrightarrow{X}_{r_2^i,G}$, $\overrightarrow{X}_{r_3^i,G}$,$F$`)`
        `//Recombination Operation`
        $\overrightarrow{U}_{i,G} \leftarrow$ `binRecombination(`$\overrightarrow{X}_{i,G}$, $\overrightarrow{V}_{i,G}$,Cr`)`
        `FitnessEvaluation(`$\overrightarrow{U}_{i,G}, TM, VD, D$`)`
        `//Selection Operation`
        if $\left(Obj(\overrightarrow{U}_{i,G}) \geq Obj(\overrightarrow{X}_{i,G})\right)$ **then**
            $NewPop$.add$(\overrightarrow{U}_{i,G})$
        else
            $NewPop$.add$(\overrightarrow{X}_{i,G})$
    $Pop \leftarrow NewPop$
    $G \leftarrow G + 1$
$\mathbb{E}_{wv^*} \leftarrow$ `GetBestSolution(`Pop, NP`)`
return $\mathbb{E}_{wv^*}$

Algorithm 7: Pseudo-code of FitnessEvaluation

Input: $\overrightarrow{X}_{i,G}, TM, VD, D$
Output: fit
//Get each base classifiers weights
for $c \leftarrow 1$ **to** D **do**
  $W[c] \leftarrow x_{c,i,G}$
//Create and evaluate weighted-vote ensemble for each fold
$MCC \leftarrow 0.0$
for $f \leftarrow 1$ **to** 10 **do**
  $EWModel \leftarrow \phi$
  //Build weighted vote EoC for f
  for $c \leftarrow 1$ **to** D **do**
    $Cls \leftarrow TM[c][f]$
    $EWModel$.add(Cls)
  //Evaluate and get MCC score on validation data VD for fold f
  $fMcc \leftarrow EWModel$.evaluate$(W, VD[f])$
  $Mcc \leftarrow Mcc + fMcc$
$fit \leftarrow Mcc/10$
$\overrightarrow{X}_{i,G}$.fitness $\leftarrow fit$
return fit

Algorithm 8: Pseudo-code of GetBestSolution

Input: Pop, NP
Output: $\overrightarrow{Best}_{i,G}$
$\overrightarrow{Best}_{i,G} \leftarrow Pop[1]$
//Compare fitness values for each individual in the population
for $i \leftarrow 2$ **to** NP **do**
  if $\left(Obj(Pop[i]) \geq Obj(\overrightarrow{Best}_{i,G})\right)$ **then**
    $\overrightarrow{Best}_{i,G} \leftarrow Pop[i]$
return $\overrightarrow{Best}_{i,G}$

Parameter	Value
Individual length (D)	20
Population Size (NP)	100
Mutation Strategy	$DE/rand/1/bin$
Scaling Factor (F)	0.9
Recombination Rate (R_χ)	0.6
Objective Function	$max(MCC)$
Maximum Evaluation	10000

Table 4.1: Parameter settings of the proposed DE-HEoC.

4.3.2 Runtime Complexity Analysis of the DE-HEoC

To estimate the runtime complexity of the DE-HEoC algorithm we begin with the most time demanding part of the algorithm, the runtime estimation of `FitnessEvaluation()`. The code for getting the base classifier weight has runtime of $\mathcal{O}(D)$, here D denotes the dimension of real-valued parameter of the individual. It is also the number of base classifiers in the algorithm. For evaluate the weighted-vote ensemble the runtime estimation is given by:

$$\mathcal{O}(FitnessIndiv) = \mathcal{O}\left(f * D * \mathbb{C}\left(TD_{m,n}\right)\right), \tag{4.8}$$

where, f denotes number of fold and $\mathbb{C}\left(TD_{m,n}\right)$ denotes the runtime estimator for classifier building for the ensemble using the training dataset $TD_{m,n}$ with m features and n samples.

Now, we calculate the runtime of DE-HEoC algorithm. Inside the main loop (line 5-29), each individual in the population will be used to generate a new population. The runtime complexity estimation of a DE algorithm is given in [Zielinski et al., 2005] as:

$$\mathcal{O}(DE) = \mathcal{O}\left(NP * D * G_{max}\right) \tag{4.9}$$

The runtime complexity estimation of DE-HEoC will be given by the runtime estimation of DE times the runtime for $\mathcal{O}(FitnessIndiv)$. So, the equation is given as:

$$\begin{aligned} \mathcal{O}(DE\text{-}HEoC) &= \mathcal{O}\left(DE * FitnessIndiv\right) && (4.10)\\ &= \mathcal{O}\left(NP * D * G_{max} * f * D * \mathbb{C}\left(TD_{m,n}\right)\right) && (4.11)\\ &= \mathcal{O}\left(NP * G_{max} * f * D^2 * \mathbb{C}\left(TD_{m,n}\right)\right) && (4.12) \end{aligned}$$

After removing the constant terms we found the upper limit for runtime complexity of DE-HEoC as

$$\mathcal{O}(DE\text{-}HEoC) = \mathcal{O}\left(NP * G_{max} * D^2 * \mathbb{C}\left(TD_{m,n}\right)\right) \tag{4.13}$$

4.4 Computational Experiments

In this section, we describe the datasets, technical details of the experiments and their results.

4.4.1 Description of Datasets

We have taken a total of 10 binary-class benchmark datasets from the UCI Machine Learning (UCI-ML) repository [Lichman, 2013]. Key characteristics of those datasets are shown in Table 4.2. It shows the number of features, sample counts and the class distribution of samples for each dataset. Features in the datasets have already been selected at the source. Therefore, they contain a small number of features (the maximum feature count is 60) for the datasets. The table also shows the count for each type of feature: real number (R), Integer (I) and Nominal (N) within parentheses. The appendicitis, sonar, titanic and wdbc datasets contain only real numbers as feature values. Conversely, the haberman and monk-2 datasets have only integer feature values. The rest of the datasets contain a mixture of real, integer and nominal feature values. Thus, we have diverse types of datasets for the experiment.

Dataset	#Feat (R/I/N)	#Samp	Class Distribution
appendicitis	7 (7/0/0)	106	85, 21
australian	14 (3/5/6)	690	383, 307
bupa	6 (1/5/0)	345	145, 200
haberman	3 (0/3/0)	306	81, 225
monk-2	6 (0/6/0)	432	204, 228
pima	8 (8/0/0)	768	500, 268
saheart	9 (5/3/1)	462	302, 160
sonar	60 (60/0/0)	208	97, 111
titanic	3 (3/0/0)	2201	1490, 711
wdbc	30 (30/0/0)	569	212, 357

Table 4.2: Characteristics of 10 binary-class datasets taken from the UCI Machine Learning Repository.

4.4.2 Experimental Setup

The DE-HEoC algorithm was implemented in the Java language and compiled with JDK version 7. We used the WEKA 3.7 data mining framework [Hall et al., 2009] and jMetal framework 4.3 [Durillo and Nebro, 2011] for implementing the DE-HEoC. All of the experiments were executed on a Dell PowerEdge III equipped with Dual Intel Xeon 5405 CPU of 2.00 GHz (8 Cores) and 32 GB RAM. The operating system of the machine was Red Hat Enterprise Linux Server 6.6. The experiment of DE-HEoC was performed 30 times in each of the datasets with different random seeds, and the average score is reported as its performance score. The DE-HEoC program and source code are available for non-commercial use at: http://sourceforge.net/projects/de-heoc/.

4.4.3 Performances of the Proposed Method

The performance of DE-HEoC on the 60–40 split of the chosen benchmark is presented in Table 4.3. The best, average (avg.) and standard deviation (std.) of the classification performance measured both in terms of MCC and accuracy are reported in the table. The standard deviations of MCCs achieved by the DE-HEoC are within 0.04 for all 10 datasets. It is less than 4% in terms of accuracy scores. These low deviant measures (both in terms of MCC and accuracy) in all experiments highlight the consistency in the DE-HEoC's performance.

Datasets	**MCC**			**Accuracy (%)**		
	Best	Avg.	Std.	Best	Avg.	Std.
appendicitis	0.62	0.55	0.04	88.68	86.16	1.41
australian	0.81	0.76	0.02	90.72	88.19	0.84
bupa	0.51	0.34	0.07	75.00	66.90	3.11
haberman	0.50	0.44	0.03	81.70	78.82	1.08
monk-2	1.00	1.00	0.00	100.00	100.00	0.00
pima	0.53	0.49	0.02	79.17	77.48	0.82
saheart	0.40	0.33	0.04	73.59	71.20	1.37
sonar	0.78	0.71	0.04	88.46	84.26	2.05
titanic	0.54	0.53	0.01	80.00	79.56	0.36
wdbc	0.96	0.94	0.02	98.24	97.14	0.74

TABLE 4.3: Summary of classification performances (in MCC and Accuracy scores) achieved by the proposed DE-HEoC for 30 runs on a 60–40 split of participating datasets.

4.5 Discussion

We have conducted further analyses on the experimental results to gain a deeper insight into DE-HEoC's performance. We compared the performances of the base classifiers, other state-of-the-art EoCs and DE-HEoC on the same train–test data splits. We have used `AdaBoostM1` (AB) [Freund and Schapire, 1996], `Bagging` (BG) [Breiman, 1996], `Random Forest` (RF) [Breiman, 2001] and `RandomCommittee` (RC) [Hall et al., 2009] as other state-of-the-art EoCs.

4.5.1 Comparison with Base Classifiers

We have measured the classification performances obtained by the base classifiers and the DE-HEoC. The classification performances on the scale of MCC and Accuracy achieved by each base classifier and DE-HEoC for the chosen 10 datasets are summarised in Figure 4.2 and Figure 4.3, respectively. In these box-and-whisker plots, the upper and lower hinges

represent the first (Q_1) and third (Q_3) quartiles (the 25^{th} and 75^{th} percentiles) of data. The whiskers are expanded as far as the highest and lowest values that are not outliers. Data points outside the 1.5 interquartile range ($IQR = Q_3 - Q_1$) from hinges are marked as outliers. The second quartile (Q_2) or median is shown by a horizontal line in the box.

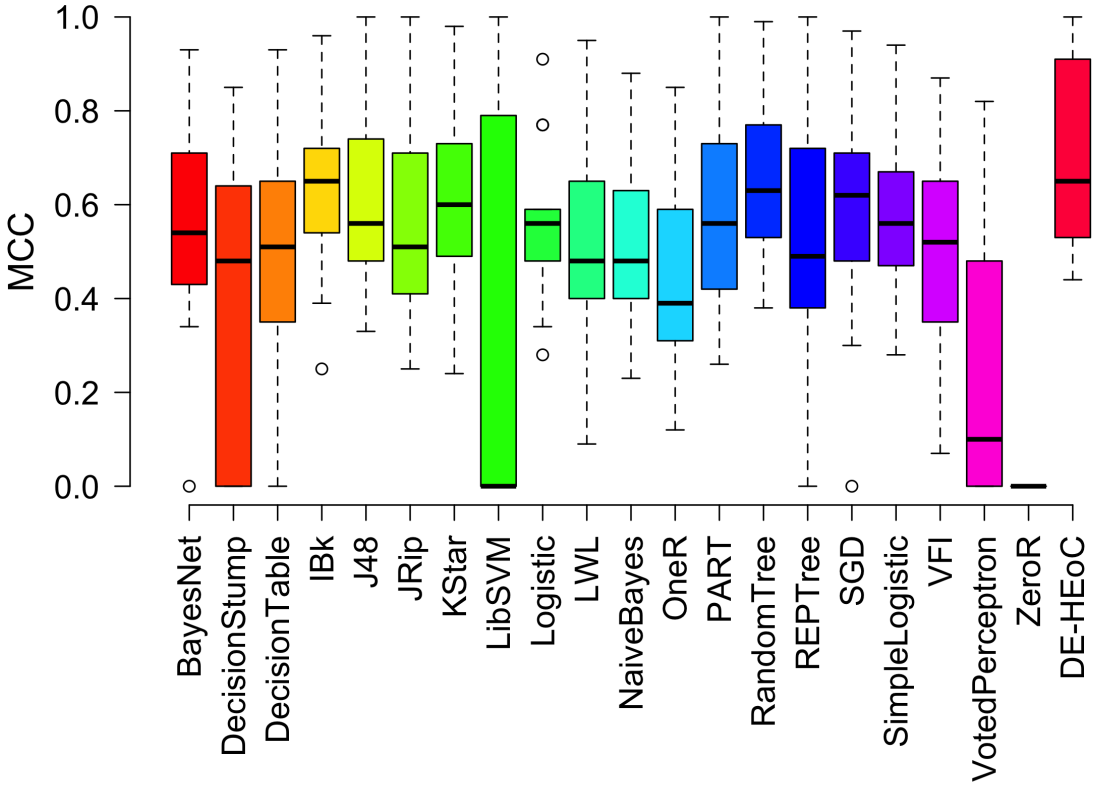


Figure 4.2: Comparisons of MCC scores achieved by base classifiers and DE-HEoC (average of 30 runs) on 10 datasets from the UCI Machine Learning Repository.

We created box-and-whisker plots of MCCs achieved for 10 datasets by each base classifier and DE-HEoC in Figure 4.2. The dotted line expresses the trend of mean MCCs of these classifiers. Between the 20 base classifiers, the `IBk` classifier performs better than all of the others with higher median and mean MCC scores. The `ZeroR` classifier is the worst performing base classifier considering the MCC score. It is clear from the figure that the mean and median of MCC scores achieved by DE-HEoC for the 10 datasets are higher than those of all the base classifiers.

In the box-and-whisker plots of Figure 4.3 we summarise the accuracies achieved by DE-HEoC and the 20 base classifiers for the 10 datasets. Base classifiers named `IBk`, `SGD`

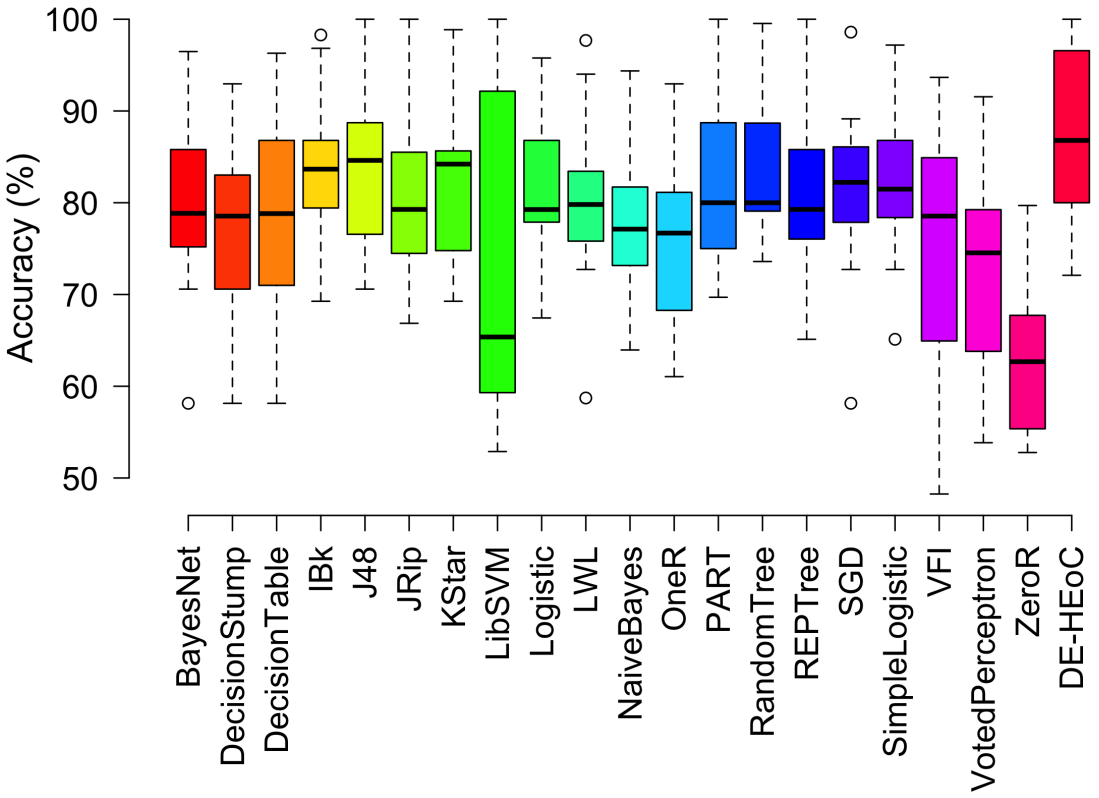


FIGURE 4.3: Comparisons of accuracies achieved by base classifiers and DE-HEoC (average of 30 runs) on 10 datasets from the UCI Machine Learning Repository.

and `VFI` achieved similar mean and median accuracy in all experiments. Their performances are also better than other base classifiers' achievements. The median and mean accuracy achieved by DE-HEoC for the 10 datasets have higher values than all of the base classifiers.

From these experimental results, we conclude that DE-HEoC clearly outperformed all base classifiers on experiments in the 10 datasets, considering both the MCC and the accuracy measures.

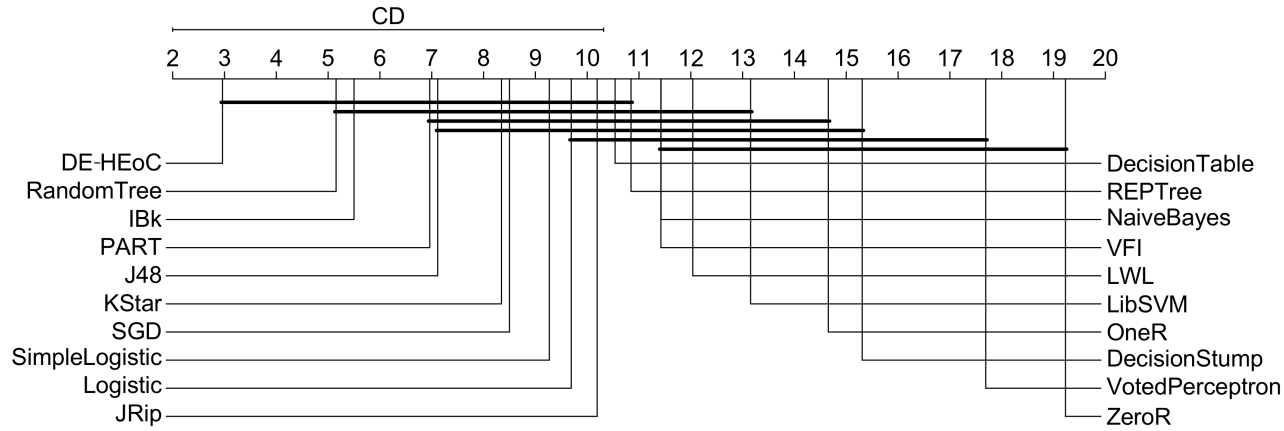


Figure 4.4: The critical difference (CD) plot show the critically significance of classification algorithms over multiple datasets for the experimental outcomes in of DE-HEoC for MCC score. The critical distance is showing the significance level of 0.05.

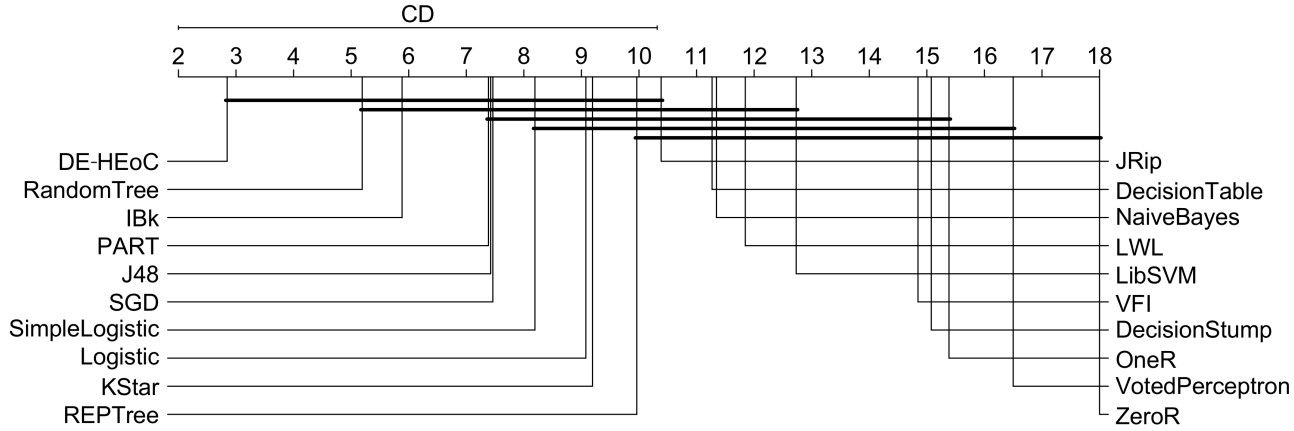


Figure 4.5: The critical difference (CD) plot shown the critically significance of classification algorithms over multiple datasets for the experimental outcomes in of DE-HEoC for accuracy score. The critical distance is showing the significance level of 0.05.

Statistical Comparison of Results

We apply the classical Friedman test and a modification by Iman and Davenport for statistically comparing the performances of DE-HEoC and base classifiers. The Iman Davenport's correction of Friedman's rank sum test provide chi-squared value $Q = 10.793$ for results for MCC score. The $p\text{-}value < 2.2e\text{-}16$ indicates that there is one or more classification algorithm exist whose MCC is significantly different. The Corrected Friedman's chi-squared value $Q = 9.7935$ is found for the accuracy scores. It has the same *p-value* as MCC score. So, for both the MCC and Accuracy scores there exist at least one classifier whose performance is significantly different than others.

Classifier	**p-value**	
	MCC	*Accuracy*
BayesNet	7.14E-03	0.00728
DecisionStump	1.06E-06	3.00E-06
DecisionTable	0.00214	0.000929
IBk	**0.361**	**0.205**
J48	**0.204**	**0.178**
JRip	0.00592	0.00553
KStar	8.34E-02	0.0269
LibSVM	2.31E-05	7.79E-05
Logistic	0.0103	0.0125
LWL	0.00106	0.00206
NaiveBayes	6.11E-04	0.000223
OneR	1.25E-06	1.01E-06
PART	**0.104**	**0.0996**
RandomTree	**0.595**	**0.402**
REPTree	0.000532	0.00646
SGD	0.0176	0.0285
SimpleLogistic	0.016	0.0329
VFI	2.57E-04	1.75E-05
VotedPerceptron	1.35E-09	1.39E-08
ZeroR	4.39E-12	1.53E-10

Table 4.4: The *p-values* from statistical test of classification performances of base classifiers and DE-HEoC for benchmarking datasets using post-hoc calculation of Friedman's Aligned Rank test with Iman Davenport's correction. The statistically similar base classifiers of DE-HEoC are shown in bold face and statistically significant classifiers are shown in normal font face.

To visualise the critical difference (CD) among classifiers over a multiple problems (datasets), the plot of the CD is generated from Nemenyi test. Here we used *alpha* =

0.05. In Figure 4.4, the CD plot shows the average rank difference of all base classifiers and GA-EoC for MCC scores. The CD plot for accuracy is shown in Figure 4.5. The critical difference value is $CD = 8.7813$ for MCC measures. Considering the DE-HEoC for MCC scores, `VFI, LWL, LibSVM, OneR, DecisionStump, VotedPerceptron` and `ZeroR` are in critical distance. The value of critical difference, CD, is 8.7813 for accuracy score also. The base classifiers `DecisionTable, NaiveBayes, LWL, LibSVM, VFI, DecisionStump, OneR, VotedPerceptron` and `ZeroR` are in critical distance from the DE-HEoC considering accuracy score.

To better understand which base classifiers are statistically significant comparing the performance of DE-HEoC, we conducted post-hoc calculation of *Friedman's Aligned Rank test with Iman Davenport's correction*. The *p-value* is shown for each of the base classifier and DE-HEoC in the Table 4.4, for MCC and Accuracy scores. The *p-value* smaller than 0.05 in a row expressed that the performance of DE-HEoC is statistically significant compared with the base classifier. Comparing the *p-value*, `IBk, J48, PART` and `RandomTree` are statistically similar to DE-HEoC for both of the MCC and Accuracy scores. Hence, the DE-HEoC is statistically significantly better than remaining 16 base classifiers.

Form the statistical test on the results, we found that DE-HEoC is significantly better approach than majority of the base classifiers (better than 16 base classifiers among 20 base classifiers for both MCC and Accuracy scores).

4.5.2 Comparison with Ensemble of Classifiers

We tested the performances of four state-of-the-art EoCs, namely `AdaBoostM1` (AB), `Bagging` (BG), `RandomForest` (RF), `RandomCommittee` (RC) and `Stacking` (Stack) available in the WEKA data mining suite [Hall et al., 2009]. We used the default parameter values for those algorithms in our experiments. The classification performances obtained for them are shown in Table 4.5 and Table 4.6 in terms of the MCC and accuracy scores, respectively. The last two rows of the table show the number of times that an ensemble has appeared as the top (#Best) and the number of times it appeared as the bottom (#Worst) of the performances in all experiments.

The MCCs achieved by each classifier ensemble are shown in Table 4.5 for experiments on the 10 datasets. Here, the EoC named `Stacking` produce 0.00 MCC scores for each dataset beacause it classifies all data into the majority label because of using the default parameter values. If we ignore the performance of `Stacking`, we find that `RandomCommittee` has been highlighted five times as the top classifier ensemble and never appeared as the worst in any experiments. The `AdaBoostM1` has never been highlighted for its performance on these dataset classifications, considering the MCC score. Moreover,

Dataset	AB	BG	RF	RC	Stack	DE-HEoC
appendicitis	0.51	0.51	0.56	**0.69**	0.00	0.62
australian	0.64	0.77	0.77	0.80	0.00	**0.81**
bupa	0.22	0.43	**0.59**	0.50	0.00	0.51
haberman	0.40	0.42	**0.67**	0.63	0.00	0.50
monk-2	0.88	**1.00**	**1.00**	**1.00**	0.00	**1.00**
pima	0.45	0.54	0.51	**0.56**	0.00	0.53
saheart	0.26	0.23	0.26	**0.40**	0.00	**0.40**
sonar	0.68	0.62	0.58	0.71	0.00	**0.78**
titanic	0.48	**0.54**	**0.54**	**0.54**	0.00	**0.54**
wdbc	0.91	**0.96**	0.92	0.94	0.00	**0.96**
#Best	0	3	4	5	0	6
#Worst	0	0	0	0	10	0

TABLE 4.5: Comparison of MCC scores achieved by DE-HEoC (average of 30 runs) and four state-of-the-art ensemble classifiers for 10 benchmarking datasets. Results are summarised as the number of best and worst performances.

it has been reported eight times as the worst performing ensemble in the experiments. The DE-HEoC has been highlighted six times as the top performing EoC considering the MCC scores achieved for the chosen 10 datasets and has never appeared as the worst performing EoC. Therefore, DE-HEoC produces better MCCs than all of the other classifier ensembles used in the experiments.

Considering the accuracy, the classification performances achieved by the EoCs are shown in Table 4.6. We can see that `Stacking` achieved the worst accuracy seven (7) times in experiments among all ensembles of classifiers. Moreover, it has never been highlighted for its accuracy scores in any of the dataset classifications. Conversely, both of the `RandomCommittee` and the DE-HEoC have been highlighted five (5) times as best performing classifier ensemble for experiments on the 10 datasets. These classifier ensembles never appeared as the worst in any of the test cases. Figure 4.6 shows the box-and-whisker plots for the accuracies achieved by the state-of-the-art EoCs and the DE-HEoC for the 10 benchmarking datasets. From the figure, it is clear that the median accuracy of DE-HEoC is higher than that of any other ensemble methods. Thus, DE-HEoC produces better generalisation than other EoCs for experiments on the 10 benchmark datasets in terms of accuracy.

Comparing both the MCC and accuracy measures on the benchmark of 10 datasets, it can be concluded that DE-HEoC exhibited the best performance among other state-of-the-art EoCs.

Dataset	AB	BG	RF	RC	Stack	DE-HEoC
appendicitis	84.91	84.91	86.79	**90.57**	86.49	88.68
australian	81.16	88.41	88.12	89.57	68.32	**90.72**
bupa	63.37	71.51	**79.65**	75.58	69.83	75.00
haberman	75.16	76.47	**83.66**	79.08	62.15	81.70
monk-2	93.52	**100.00**	**100.00**	**100.00**	66.23	**100.00**
pima	76.04	79.69	78.13	**80.47**	75.09	79.17
saheart	69.70	68.40	69.70	**74.46**	75.23	73.59
sonar	83.65	79.81	77.88	84.62	67.12	**88.46**
titanic	78.55	**80.00**	**80.00**	**80.00**	76.90	**80.00**
wdbc	95.77	**98.24**	96.48	97.18	73.37	**98.24**
#Best	0	3	4	5	0	5
#Worst	2	1	0	0	7	0

Table 4.6: Comparison of accuracy (%) achieved by DE-HEoC (average of 30 runs) and four state-of-the-art ensemble classifiers for 10 benchmarking datasets. Results are summarised as the number of best and worst performances.

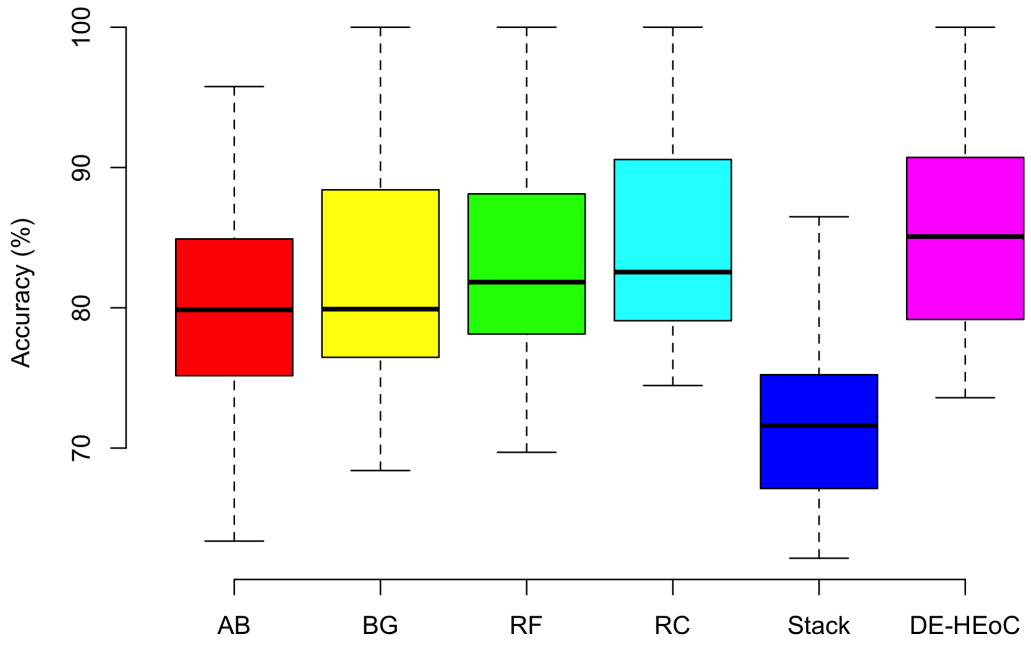


Figure 4.6: Comparison of accuracy scores achieved by state-of-the-art ensemble of classifiers and DE-HEoC (average of 30 runs) for 10 benchmarking datasets.

4.6 Case Study: Application in Heart Disease Prediction

In 2015, Bashir et al. [Bashir et al., 2015b] proposed a weighted-voting EoC system to predict heart disease. They used five binary-class benchmark datasets to verify the effectiveness of their weighted-voting EoC. These benchmark datasets are taken from the UCI-ML repository [Lichman, 2013], except for ERIC [Rakotomalala, 2013]. Descriptions of those datasets, including the test and train distribution, are tabulated in Table 4.7. We used the same set of data to compare the performance of our proposed weighted-voting-based EoC using optimising of the MCC score. We performed the experiments on each dataset 30 times.

After executing the DE-HEoC 30 times on each of the heart disease datasets, we plot the performance with the aid of box-and-whisker plots for Accuracy and MCC in Figure 4.7a and Figure 4.7b, respectively. Figure 4.7a shows the performances considering the MCC scores for each dataset. Here, we can see that the performances for 30 repetitions of DE-HEoC are consistent because the spread of boxes is very narrow. This phenomenon is also true for the accuracy scores shown in Figure 4.7b. From the performance analysis on heart disease prediction, we can claim that the DE-HEoC is a robust method for classification.

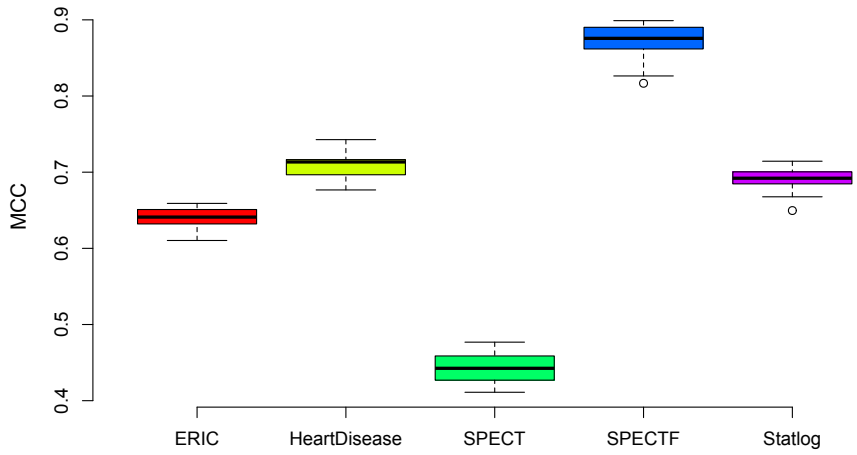
Dataset	**#Feat (R/I/N)**	**#Samp**	**#(Train, Test) Samp**
ERIC	7(0/3/4)	209	10-fold CV
HeartDisease	13(1/12/0)	303	10-fold CV
SPECT	22(0/44/0)	267	(80, 187)
SPECTF	44(0/44/0)	267	(80, 187)
Statlog	13(7/3/3)	270	10-fold CV

TABLE 4.7: Detailed characteristics of Heart Disease prediction datasets.

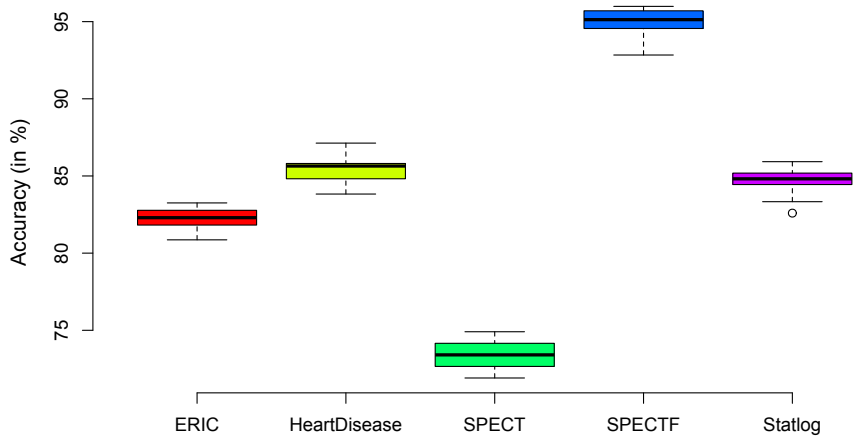
Dataset	**[Bashir et al., 2015b]**			**DE-HEoC**		
	F-Meas	Acc (%)	MCC	F-Meas	Acc (%)	MCC
ERIC	**78.51**	75.19	0.62	78.33 ± 1.14	**82.28 ± 0.67**	**0.64 ± 0.013**
HeartDisease	82.17	81.82	0.66	**87.10 ± 0.58**	**85.14 ± 0.84**	**0.70 ± 0.015**
SPECT	77.15	**80.75**	0.35	**78.52 ± 0.46**	73.85 ± 0.66	**0.45 ± 0.015**
SPECTF	73.00	72.73	0.27	**90.13 ± 1.84**	**95.00 ± 0.82**	**0.87 ± 0.021**
Statlog	**87.38**	**87.57**	**0.74**	**86.92 ± 0.60**	84.89 ± 0.80	0.69 ± 0.016

TABLE 4.8: Comparison of best classification performances by Bashir et al. (2015) and the average performances (± standard deviation) for 30 runs of the DE-HEoC for heart disease prediction datasets.

In Table 4.8, we report the average performances of DE-HEoC with standard deviations in different measures. To make a fair comparison, we compared our result for F-measure,



(a) MCC



(b) Accuracy

Figure 4.7: Box-and-whisker plots showing the classification performances achieved by DE-HEoC for 30 runs on heart disease prediction datasets considering the (a) MCC and (b) Accuracy scores.

accuracy and MCC scores calculated from the confusion matrices of Bashir et al. [Bashir et al., 2015b]. In terms of F-measure, our method achieved a better score for HeartDisease, SPECT and SPECTF datasets and similar (including the standard deviation) for the remaining datasets. While considering the accuracy, our method performed better than Bashir et al.'s for three datasets. In the case of the MCC score, the DE-HEoC produces better scores than theirs for all heart disease prediction datasets, except Statlog. The experimental outcomes of the proposed method clearly express that the proposed DE-HEoC performed better in general than Bashir et al. [Bashir et al., 2015b] for heart disease prediction.

4.7 Case Study: Churn Prediction

It is well-known amongst business and marketing managers that obtaining new customers is a more difficult than maintaining existing ones. It is also more costly and time-consuming to the business. Although customer relationship management (CRM) is not always easy either, hanging on to existing customers is usually more feasible than trying to get new customers (who may not even know about your business or product yet). Therefore, the concept of customer churn is an interesting research area. A customer is a 'churner' when he/she leaves the company as a customer. This concept is more applicable to service companies when long-term and ongoing relationships between the company and the consumer are more common, for instance, banks or phone or internet providers. Naturally, businesses with their increased technological capabilities today would like to be able to predict these churners or, at the very least, classify them from the customers who remain (non-churners) so that certain insights can be made into why these customers are churning. This scenario provides a prime example for the DE-HEoC algorithm to be tested in. Specifically, we have a dataset of consumers that includes churners and non-churners of a Telecom services company.

The Churn dataset is a Telecom service customer dataset and is publicly available online [1]. The objective is to predict 'churners' (those customers who left the company) and classify them from the 'non-churners' (those customers who stayed with the company). More details on this dataset can be found in [Obiedat et al., 2013]. It is a binary-class dataset containing 3333 samples and 20 features.

4.7.1 Classification Performances of DE-HEoC for Churn Prediction

The classification performance summary of DE-HEoC for 30 independent runs on churn dataset using 10-fold cross-validation is shown in Table 4.9. The performances are shown for four classification measures. For MCC score, the best performance exhibited by DE-HEoC with a score of 0.801. The average is 0.766 with 0.059 as standard deviation. The minimum MCC score attained by DE-HEoC during 30 independent runs is 0.656. The best accuracy of 95.30% is achieved by DE-HEoC. The lowest accuracy for DE-HEoC on churn dataset is 92.30%. The average with standard deviation is $94.6pm1.20\%$ for 30 independent runs of DE-HEoC. Table 4.9 also shows the same summary statistics for precision and F-Measure scores.

[1] http://www.dataminingconsultant.com/data/churn.txt

Statistic	Mean	St. Dev.	Min	Max
MCC	0.766	0.059	0.656	0.801
Accuracy	94.60	1.20	92.30	95.30
Precision	0.951	0.016	0.922	0.964
F-Measure	0.969	0.006	0.957	0.973

Table 4.9: Classification performances for 30 runs of the DE-HEoC on Churn datasets.

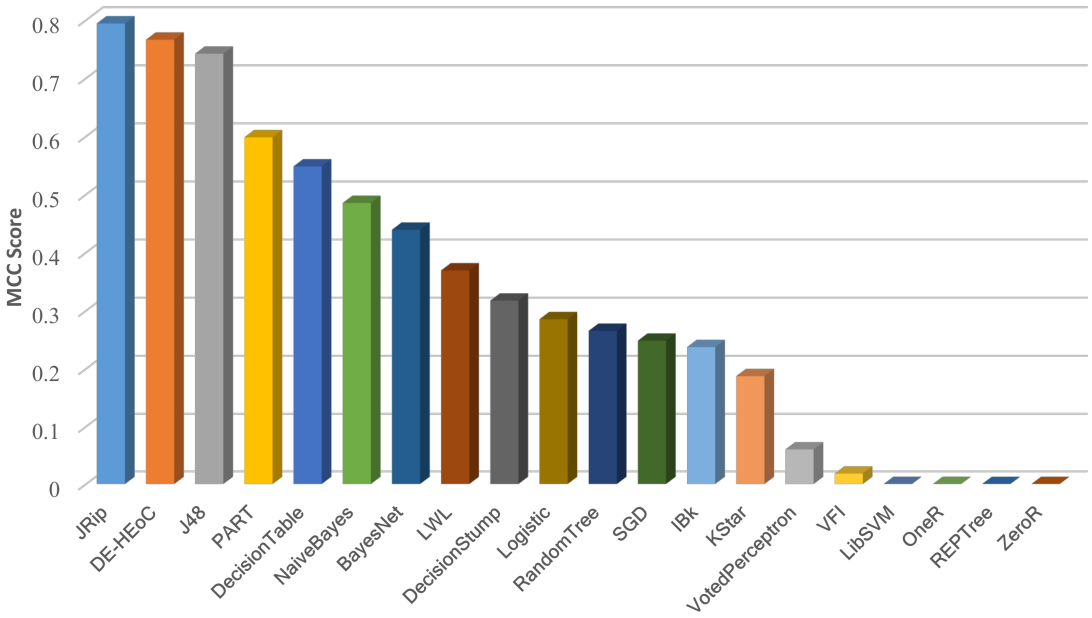


Figure 4.8: Comparison of classification performances order by MCC scores for churn prediction by base classifiers and avergae DE-HEoC

Classification Performance Comparison with Base Classifiers

Now we compare the classification performances of DE-HEoC with it's 20 base classifiers. The Figure 4.8 show the performances comparison of base classifiers and DE-HEoC for MCC scores. The average performance of DE-HEoC for 30 independent runs is compared against base classifier. For the MCC score, the `JRip` classifier is the best performing classifier and it outperform the DE-HEoC for Churn prediction. Here, the average performance of DE-HEoC is not better than the base classifiers. Detail performances of base classifiers is available in Table S22 in Appendix E for different performance measures.

Classification Performance Comparison with other Ensemble of Classifiers

The classification performance of DE-HEoC is now being compared with other state-of-the-art ensemble of classifiers. No parameter tuning is done for the execution of the ensemble classifiers. We used the default parameter values available in Weka software. Here, the DE-HEoC outperformed all ensemble of classifiers for churn prediction.

Classifier	MCC	Acc	Prec	F-Meas
AdaBoostM1	0.366	87.28	0.853	0.851
Bagging	0.000	85.51	0.731	0.788
RandomCommittee	0.268	86.74	0.862	0.821
RandomSubSpace	0.156	85.93	0.87	0.799
RandomForest	0.133	85.81	0.878	0.795
RandomTree	0.265	85.45	0.824	0.830
Stacking	0.000	85.51	0.731	0.788
DE-HEoC (avg)	**0.766**	**94.60**	**0.951**	**0.969**
DE-HEoC (stdev)	0.059	1.20	0.016	0.006

Table 4.10: Classification performances comparison of other ensemble of classifiers and the average performances for 30 runs of the DE-HEoC for Churn datasets.

The classification performance of DE-HEoC is not excellent for churn prediction. Some base classifier performed better than the DE-HEoC. However, it performed better than other ensemble methods on their default settings.

4.8 Conclusion

We propose a DE algorithm to optimise the weights of base classifiers used in a HEoC. These weights are optimised using the average MCC scores calculated in 10-fold CV of a training dataset. The performance of the weighted EoC has been evaluated on 10 benchmark datasets. The performance is compared with the results achieved by base classifiers and four other state-of-the-art EoCs. The overall classification performance achieved by the proposed method is found to be better in experiments on the 10 datasets. The experimental performances were compared with a recently proposed weighted-voting ensemble method for heart disease prediction datasets. The result comparison revealed the DE-HEoC as a better choice for predicting heart disease on those datasets. In addition, the proposed method exhibited an overall superiority in data classification over individual classifiers and other EoCs compared in the experiments.

Furthermore, in comparison of the result achieved by the weighted vote (DE-HEoC) with the majority vote (GA-EoC) we found that the DE-HEoC did not outperform the

GA-EoC. Let us considering the accuracy score achieved in common datasets for both experiments. It revealed that for bupa, DE-HEoC achieved 75.00% and GA-EoC achieved 75.72% accuracy. For pima dataset DE-HEoC achieved 79.17% and GA-EoC achieved 97.43% accuracy score. For the wbc dataset, DE-HEoC did not outperform the GA-EoC (accuracy for DE-HEoC is 98.24% and GA-EoC is 99.43%). Considering the MCC score, the DE-HEoC produced 0.51 MCC score for bupa datasets (GA-EoC achieved 0.50 for bupa). For, the pima and wdbc dataset DE-HEoC achieved 0.53 and 0.96 MCC score, respectively. GA-EoC achieved 0.94 and 0.99 MCC score for pima and wdbc datasets, respectively. Similarly, we found for the MCC score that DE-HEoC did not able to outperform the GA-EoC. Even tough DE-HEoC perform well in some real-world datasets in compared with base classifiers and other ensemble of classifiers, it unable to preform better in comparison to the majority voting approach using genetic algorithm. The performance of DE-HEoC could be boosted by using other measures as weight than the MCC score. Further research could explore this opportunity to optimise other classifier measure in weighted voting ensemble of classifier system.

This page intentionally left blank.

This chapter contains parts of the accepted chapter titled "A Multi-objective Meta-Analytic Method for Churn Prediction" in the 1st edition of "Business and Consumer Analytics: New Directions" book. Another manuscript is being prepared based on the computational outcomes of breast cancer dataset, METABRIC, followed by biological analysis and verification for IEEE Transaction.

5

Multi-objective Ensemble of Classifiers

In machine learning research, the primary objective in data classification is to enhance the generalisation ability. We have shown in the two previous chapters that the EoC approach is one useful technique to improve the generalisation capability in data classification. Although appropriately aggregated ensembles could outperform their members, constructing the members to meet the theoretical criterion (combining all accurate and diverse base classifiers) is not an easy task [Gu et al., 2015]. Because of the success of single-objective (mostly accuracy-based) optimisation, many researchers have turned their focus into multi-objective optimisation (MOO) (such as combining other factors as objectives) for further improvement of generalisation in the ensemble of classifiers [Chandra and Yao, 2006, Jin and Sendhoff, 2008, Kumar and Kumar, 2012a, Gu et al., 2015]. The multi-objective optimisation (MOO) could possibly handle with multiple conflicting criteria to optimise the outcome than single-objective ones.

The primary motivation for using a multi-objective evolutionary approach to find the base classifiers combination for the heterogeneous ensembles is that multi-objectivity forces the search process to find a set of near-optimal solutions instead of just a single solution. Obtaining a set of solutions necessarily means the decision maker will have options to choose near-optimal combinations of base classifiers. The decision-making process could be tailored based on the trade-off between multiple objectives. Therefore, incorporating

the use of multi-objective evolutionary optimisation for finding ensemble combinations is an attractive idea for both the homogeneous and heterogeneous types.

5.1 Multi-objective Optimisation

In real-world applications, most of the optimisation problems require the optimisation of more than one objective. The objectives found in real-world scenarios are often conflicting, e.g., maximise performance, minimise cost, maximise reliability, etc. In those cases, in general we cannot find a final solution that is optimal for all objective functions and the optimal solution of one objective will not necessary be the best solution for other objective(s). Therefore, many solutions will produce trade-offs among the various objectives and a set of solutions would be required to represent the optimal solutions of all objectives.

In general, a MOO problem (MOP) is defined as finding optimal solutions for optimising more than one conflicting objective (**min** $f(x)$ $s.t.$ $x \in S$ where f is a scalar function and S is the set of constraints). MOPs can be defined as:

$$\begin{aligned} (\textbf{Minimise})\, [f_1(x), f_2(x), \cdots, f_p(x)] \\ x \in \mathbf{S}, \end{aligned} \tag{5.1}$$

where f_i, with $i = 1, 2, \cdots, p$ is the i-th objective function and x is a solution from the set of solutions $\mathbf{S}$. In a MOO we obtain more than one solution for the optimisation problem. The set of solutions is called a Pareto-optimal solution.

A solution $x^* \in \mathbf{S}$ is called *Pareto-optimal* in a MOO problem if there does not exist another solution that dominates it. A solution is called *non-dominated* if all other solutions $x \in \mathbf{S}$ have a higher value for at least one of the objective functions f_i or have the same value for all of the objective functions [Caramia and Dell'Olmo, 2008].

For a two-objective problem, we can plot the objective values of solutions (also denoted as a *Pareto front*) in an x–y plane. An example of a Pareto curve is shown in Figure 5.1, where all of the points on the Pareto curve between $(f2(\hat{x}), f1(\hat{x}))$ and $(f2(\tilde{x}), f1(\tilde{x}))$ are called non-dominated points.

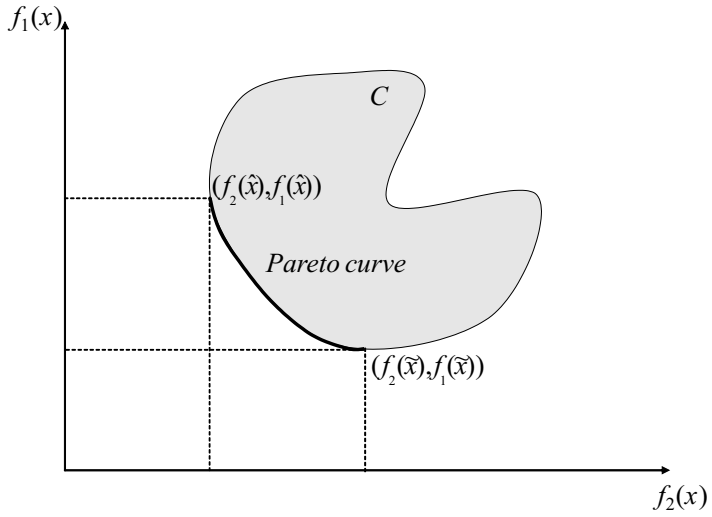


Figure 5.1: An example of a Pareto front. The figure is adapted from [Caramia and Dell'Olmo, 2008] with Permission Number 03832290290836.

5.2 Multi-Objective Ensemble of Classifiers (MO-EoC)

In this section, we are going to present an empirical study of multi-objective ensemble combination search approaches. We will investigate the effects of combining structural diversity, the number of base classifiers and classification performance (in MCC measure) to find the best combination of ensembles for the problem at hand.

5.2.1 Literature Review

The use of MOO has previously been proposed for the EoC. However, the approaches differ from one another based on the ensemble types, objective function, choosing the solution and underlying MOO algorithms. We can broadly divide MOO-based EoC methods into two categories: homogeneous and heterogeneous EoC. We present some recent work on combining ensemble learning with multi-objective evolutionary optimisation.

Dietterich [Dietterich, 1997] pointed out the prerequisites to formulate a multi-objective ensemble as the trade-off between the criteria of base classifiers needs to be better than random classifiers (accuracy) and they should make errors in different parts of the input spaces (diversity). From the optimisation point of view, improving the accuracy and increasing diversity of ensembles are likely to be conflicting objectives with each other [Opitz et al., 1996, Opitz and Maclin, 1999]. To find the right balance between these two objectives, researchers attempted to transform the MOO problem into single-objective problems by using hyperparameters [Gu et al., 2015, Jin and Sendhoff, 2008]. Another stream of

ensemble research explored the finding of better base classifiers for the problem. Zhou et al. [Zhou et al., 2002] formally showed that finding the most relevant subset of classifiers is more efficient regarding performance than combining all of the available classifiers. Therefore, selection of the ensemble is treated as an important task in creating EoCs.

[Dos Santos et al., 2006] proposed a multi-objective GA for the homogeneous EoC selection. They used three objective functions: the error rate (1−True Positive Rate), number of classifiers and diversity. They applied the MOO algorithm in two levels of the selection process. In the first level, they used non-dominated sorting GA II (NSGA-II [Deb et al., 2002]) to find the set of optimal solutions. To select a solution from the Pareto-optimal solutions, they applied the second level of the selection procedure. Then, they evaluated each solution using an independent validation dataset and selected a solution. From their analysis, they concluded the cardinality (number of base classifiers) in the ensemble has no effect in increasing the generalisation performance of an EoC as an individual or in pairs with other objectives. This homogeneous EoC was created with k-NN classifiers considering the crowding distance.

Later, [Ahmadian et al., 2007] also proposed a bi-level homogeneous EoC using a multi-objective EA. They used cardinality, accuracy, entropy and Q-statistics as objectives. In the first level, they generated a set of base classifiers based on producing the minimum classification error for each class. In the next level, they applied the NSGA-II algorithm to find the optimal ensemble combination from preselected base classifiers. In this level, they selected the ensemble with better accuracy. In the case of a tie, they chose the solution with the smaller ensemble size. They proposed the use of neural networks or k-NN as the underlying base classifier.

[Dos Santos et al., 2008] applied two Pareto spread quality measures to analyse the relationship between the three most important search criteria to select the EoC combinations, which are the ensemble error rate, ensemble size and diversity measures. The NSGA-II is applied to find the best pair of objective functions from diversity measures, error rate and ensemble size.

Next, [Kumar and Kumar, 2012b] proposed a multi-objective homogeneous EoC using ANNs for intrusion detection. In their two-phase MOO ensemble, first they used real-coded multi-objective GA (MOGA) for the training of base classifiers. They used detection rate (DR) of each class and false positive rate (FPR) as objectives in this phase. An NSGA-II-like archive of non-dominated sorting was adopted in the next step. Here, binary-coded MOGA was used for the selection of base classifiers. They formulated an ANN in the second phase MOO algorithm using the base classifiers generated from the first step. [Levesque et al., 2012] also used homogeneous multi-objective EA (MOEA) for generating

ensembles of classifiers. Class predictions (minimisation of FPRs and maximisation of true positive rates) are combined using different Boolean functions (AND, OR, XOR, etc.) to optimise the Receiver Operating Characteristics (ROC) curve. [Rahman and Verma, 2013] proposed cluster oriented homogeneous ensemble classifiers using the MOEA. The accuracy and diversity were two objectives to optimise. [Chiu and Verma, 2014] also used a MOEA for optimisation of the neural network ensemble classifier. The accuracy and diversity were used as the objective functions for the layers in the neural ensemble classifier. Conversely, they applied the MOO algorithm in two phases, which could be computationally very expensive for high-dimensional datasets.

[Oehmcke et al., 2015] analysed pattern and feature diversity methods for balancing ensemble classifiers. They used the evolutionary multi-objective algorithm (an adaptation of NSGA-II) for NN classifier ensembles and decision tree ensembles. The runtime (average training time and prediction time) and accuracy (combines the classification error and the corresponding standard deviation) are used as the objective functions to optimise.

Subsequently, [Nguyen et al., 2014] introduced a mechanism to learn optimal classifiers combining algorithms for an ensemble system by using a GA. The HEoC was formed with three base classifiers, namely LDA, NB and k-NN. The number of correctly classified observations, the number of selected features and the number of selected classifiers were the objective functions of their algorithm. [Bashir et al., 2015a] proposed a variant of the `Bagging` algorithm using multi-objective optimised weighted-voting ensemble for heart disease prediction. The F-Measure score was used as the voting weights of heterogeneous base classifiers (NB, LR, QDA, IBL, SVM). To select the final solution, they converted two objectives (Precision, Recall) into a single measure. [Sikdar et al., 2015] proposed multi-objective DE for FS and classifier ensemble. They also transformed precision and recall score into F-measure for selecting the best solution from the Pareto-optimal front.

The key characteristics of homogeneous and heterogeneous EoC using MOO are shown in Table 5.1 and Table 5.2, respectively. We can observe that most of the algorithms used different measures for classification performances and diversity as objective values. Many of them also used the size of the ensemble combination. In [Ahmadian et al., 2007, Dos Santos et al., 2006, Dos Santos et al., 2008], measures of diversity proposed by [Kuncheva and Whitaker, 2003] were used. Most of the time, the NSGA-II is used as the underlying MOO algorithm.

Algorithm	Base Classifiers	MOO Tool	Objective Functions	Solution Selection
[Dos Santos et al., 2006]	k-NN	NSGA-II	Error Rate, Size, Diversity	Validation error rate
[Dos Santos et al., 2008]	k-NN	NSGA-II	Error Rate, Size, Diversity	Not Mentioned
[Kumar and Kumar, 2012b]	ANN	MOGA & NSGA-II	Detection Rate (DR), FP Rate	Create ANN ensemble
[Levesque et al., 2012]	SVM	MOEA	FP Rate, TP Rate	ROC Curve
[Rahman and Verma, 2013]	SVM	MOEA	Accuracy, Diversity	Not Mentioned
[Chiu and Verma, 2014]	ANN	MOEA	Accuracy, Diversity	Not Mentioned
[Sikdar et al., 2015]	CRF	DE	Precision, Recall	F-Measure
[Nag and Pal, 2016]	Binary Tree	MOGP	FP, FN, size	net belongingness

Table 5.1: Key characteristics of multi-objective homogeneous ensemble of classifiers in chronological order.

Algorithm	Base Classifiers	MOO Tool	Objective Functions	Solution Selection
[Ahmadian et al., 2007]	MLP, kNN	NSGA-II	Size, Accuracy, Entropy, Q-Stat	Accuracy and size
[Nguyen et al., 2014]	LDA, NB, kNN	MOGA	TP, #selected features, Size	arithmetic
[Oehmcke et al., 2015]	k-NN, decision tree	NSGA-II	Runtime, error with stdev	None

Table 5.2: Key characteristics of multi-objective heterogeneous ensemble of classifiers in chronological order.

In the next subsections, we are going to describe the structure of the MO-EoC combination search approach. First, we will present some brief background of multi-objective ensembles including base classifiers, their diversity measurements and classification performances. Then, we will describe the structure of the proposed algorithm in detail.

5.2.2 Selection of Feature Selection Methods

We considered a total of eight feature quality estimation methods (also known as *rank-based FS method* or *feature evaluation methods*). Among these, six FS methods are implemented in the WEKA data mining software and Craig–Moscato's feature ranking 1 (CM1) [Marsden et al., 2013] and Craig-Moscato's feature ranking 2 (CM2) [Filiou et al., 2014] have been implemented using the Java 7 language. A list of ranking-based FS methods considered is shown in Table 5.3. We selected FS methods based on their running time and

FS Method	**Short Name**	**Reference**
Craig-Moscato's Feature Ranking 1	CM1	[Marsden et al., 2013]
Craig-Moscato's Feature Ranking 2	CM2	[Filiou et al., 2014]
Information Gain Attribute Evaluation	IG	[Hall et al., 2009]
Gain-Ratio Attribute Evaluation	GR	[Hall et al., 2009]
Correlation-based Attribute Evaluation	CR	[Hall et al., 2009]
OneR FS	OR	[Hall et al., 2009]
ReliefF Attribute Evaluation	RF	[Kira and Rendell, 1992]
Symmetric Uncertainty Attribute Evaluation	SU	[Hall et al., 2009]

Table 5.3: List of feature selection methods selected for experiments.

generalisation ability in data classification.

Datasets

To compare the goodness of participating FS methods, we used datasets from the NIPS 2003 challenge in the FS competition. The database contains five challenging real-world datasets. We applied FS methods on training datasets and calculated their generalisation performances with 39 classifiers on the validation datasets. Table 5.4 shows the number of features, the number of samples in training and validation datasets with their class distribution ratio (ratio of the number of positively labelled samples by the number of negatively labelled samples) per dataset.

Dataset	Domain	#Features	#Train Samples (Pos:Neg)	#Valid Samples (Pos:Neg)
arcene	Mass Spectrometry	10000	100 (1:1.3)	100 (1:1.3)
dexter	Text Classification	20000	300 (1:1)	300 (1:1)
dorothea	Drug Discovery	100000	800 (1:9.3)	350 (1:9.3)
gisette	Digit Recognition	5000	6000 (1:1)	1000 (1:1)
madelon	Artificial	500	2000 (1:1)	600 (1:1)

TABLE 5.4: Characteristics of datasets used for the selection of feature selection methods.

Evaluation of Selection Criterion

We report the running time (CPU time) required for each of the FS methods to select the top 100, 200, 300, 400 and 500 features from the datasets. We also report the generalisation performances achieved for the FS methods by 39 classification methods (listed in Table 5.5). Based on these performance comparisons, we will choose a set of FS methods.

Running Time: We executed all experiments on Dell PowerEdge III servers equipped with Dual Xeon 5550 2.67 GHz (eight cores) CPU and 32 GB RAM. Table 5.6 shows the experimental results in terms of CPU times required to select the top 100, 200, 300, 400 and 500 features. The best and worst running time by FS methods for selecting each of the top n-features are highlighted by bold and italic, respectively.

From the running times presented in Table 5.6, we observe that the CM1 FS method performed better than other approaches for selecting the top 300, 400 and 500 features from the arcene dataset. The Symmetric Uncertainty (SU) spent less CPU time for selecting the top 100 and 200 features from the arcene dataset. The OneR (OR) FS method appeared to be the worst FS method for this dataset regarding running times. For FS from the dexter dataset, CM2 outperforms all FS approaches. The OR FS method performed worst for this dataset also. Similarly, CM2 spent the least CPU time in FS for the dorothea dataset. RF performed worst in respect of execution times for the dorothea dataset. For the gisette dataset, CM2 spent least CPU time for selecting the top 100, 200 and 300 features. CM1 uses the least amount of CPU time for selecting 400 and 500 features. The RF spent most CPU time compared with other FS approaches. For the madelon dataset, CM1 uses least and RF uses most running time for FS. The Correlation (CR) FS method also uses the least amount of CPU time for selecting 100, 200, 300 and 400 features from this dataset. Table 5.7 summarises these results by showing average and standard deviations of CPU times spent for selecting the top 100, 200, 300, 400 and 500 features for each dataset. It is notable from here, CM1 and CM2 use lower CPU times than other

Type	Base Classifiers	Brief Description
Bayes	BayesNet	Bayes Network learning algorithm.
	NaiveBayes	Naive Bayes classifier using estimator [John and Langley, 1995].
Support Vector Machine	LibSVM	Support Vector Machine [Chang and Lin, 2011].
	SMO	Sequential Minimal Optimization for SVM [Platt, 1998].
	SPegasos	Stochastic variant of the Pegasos for SVM [Shalev-Shwartz et al., 2011].
Linear	Logistic	Logistic regression with ridge estimator [Le Cessie and Van Houwelingen, 1992].
	SimpleLogistic	'LogitBoost', the speed-up logistic regression [Sumner et al., 2005a].
	SGD	Stochastic gradient descent for learning linear models.
Neural Network	MLPClassifier	Trains a multilayer perceptron with one hidden layer.
	MultilayerPerceptron	A Classifier that uses backpropagation to classify instances.
	RBFNetwork	Normalized Gaussian radial basisbasis function network.
	VotedPerceptron	Voted perceptron algorithm [Freund and Schapire, 1998].
Decision Tree	ADTree	Generate an alternating decision tree [Freund and Mason, 1999].
	BFTree	Build a best-first decision tree classifier [Shi, 2007, Friedman et al., 2000].
	HoeffdingTree	Incremental, anytime decision tree algorithm [Hulten et al., 2001].
	J48	Generate a pruned or unpruned C4.5 decision tree [Quinlan, 2014].
	LADTree	Multi-class alternating decision tree using the LogitBoost [Holmes et al., 2001].
	REPTree	Fast decision tree learner uses information gain and prunes using reduced-error pruning.
	PART	PART decision list [Frank and Witten, 1998].
Trees	ExtraTree	Class for generating a single Extra-Tree [Geurts et al., 2006].
	FT	Classifier for building 'Functional trees' [Gama, 2004].
	LMT	Classifier for 'logistic model trees' [Landwehr et al., 2005b, Sumner et al., 2005b].
	NBTree	Decision tree with naive Bayes classifiers at the leaves [Kohavi, 1996].
	RandomTree	Tree-based classifier that considers K randomly chosen attributes at each node.
	SimpleCart	Class implementing minimal cost-complexity pruning [Breiman et al., 1984].
Nearest Neighbours	IBk	K-nearest neighbours classifier [Aha et al., 1991].
	NNge	Nearest-neighbor using non-nested generalized exemplars [Martin, 1995, Roy, 2002].
Instance-Based	KStar	An Instance-based Learner Using an Entropic Distance Measure [Cleary et al., 1995].
	LWL	Locally weighted learning [Christopher et al., 1997].
	HyperPipes	Class implementing a HyperPipe classifier.
Rule Learner	ConjunctiveRule	Class implements a single conjunctive rule learner.
	JRip	Propositional rule learner [Cohen, 1995].
	Ridor	Implementation of a RIpple-DOwn Rule learner [Gaines and Compton, 1995].
Decision Table	DecisionTable	Simple decision table majority classifier [Kohavi, 1995].
	DTNB	Decision table/naive bayes hybrid classifier [Hall and Frank, 2008].
Decision Stump	DecisionStump	Class for building and using a decision stump.
Fuzzy Rule	FURIA	Fuzzy Unordered Rule Induction Algorithm [Hühn and Hüllermeier, 2009].
Ordinal Learner	OLM	Implementation of the Ordinal Learning Method [Ben-David, 1992].
Feature Interval	VFI	Classification by voting feature intervals [Demiröz and Güvenir, 1997].

Table 5.5: List of base 39 classifiers considered for the experiments with their type and their short description. Classifiers without references in brief description are available in WEKA [Hall et al., 2009]

Dataset	#Sel.Feat	CM1	CM2	IG	GR	CR	OR	RF	SU
arcene	100	0.34	0.45	0.72	0.59	0.46	*18.91*	2.24	**0.27**
	200	0.30	0.47	0.69	0.44	0.48	*17.45*	2.39	**0.28**
	300	**0.24**	0.38	0.49	0.31	0.36	*17.96*	2.36	0.30
	400	**0.27**	0.35	0.67	0.30	0.31	*18.59*	2.29	0.30
	500	**0.29**	0.40	0.60	0.40	0.42	*20.73*	2.62	0.37
dexter	100	**0.74**	**0.74**	1.80	2.18	1.02	*50.54*	33.44	1.32
	200	0.70	**0.66**	1.54	2.11	1.00	*50.05*	34.24	1.30
	300	0.98	**0.67**	1.64	1.70	1.10	*50.90*	35.06	1.38
	400	0.69	**0.67**	1.57	1.76	0.84	*51.08*	35.67	1.40
	500	1.22	**1.03**	1.92	2.10	1.45	*49.81*	33.68	1.40
dorothea	100	13.19	**12.76**	25.83	30.72	13.21	852.04	*982.62*	31.69
	200	13.58	12.96	24.39	29.60	**12.89**	837.93	*975.96*	31.05
	300	14.30	**13.79**	28.94	35.22	14.88	1059.92	*1243.01*	36.38
	400	14.71	**14.03**	30.26	35.61	15.10	1083.24	*1321.26*	33.46
	500	13.27	**12.81**	25.77	31.49	14.60	851.37	*961.12*	31.22
gisette	100	1.36	**1.26**	11.64	12.58	1.37	74.20	*2447.58*	12.38
	200	1.16	**1.13**	11.60	13.68	1.49	74.42	*2409.74*	12.28
	300	1.18	**1.18**	11.49	13.41	1.28	83.42	*2565.09*	12.75
	400	**1.19**	1.24	11.07	13.11	1.34	81.40	*2545.88*	13.72
	500	**1.21**	1.22	10.57	12.42	1.33	74.66	*2412.82*	13.27
madelon	100	**0.03**	0.03	0.27	0.31	0.03	2.79	*29.41*	0.42
	200	**0.04**	0.05	0.27	0.31	**0.04**	2.80	*29.77*	0.38
	300	**0.04**	0.06	0.37	0.41	**0.04**	2.84	*29.69*	0.43
	400	**0.04**	0.04	0.33	0.35	**0.04**	3.12	*30.01*	0.35
	500	**0.03**	0.04	0.28	0.26	0.04	2.75	*29.35*	0.26

TABLE 5.6: CPU times (in s) required to select the different number of features (top 100, 200, 300, 400 and 500) by all feature selection methods (detail of FS methods shown in Table 5.3). The bold value represents the best CPU time and italic value represents the worst CPU time required for the FS method to select the specified number of features.

approaches on average. CM1 was better in respect of CPU time for FS from the arcene and madelon datasets, and CM1 performed better for dexter, dorothea and gisette datasets. If we consider the size of the datasets, we find that CM2 requires less CPU time than CM1 for FS for large-scale datasets.

Based on these results, we have removed OneR (OR) and ReliefF (RF), the two most expensive (considering the running times) techniques, from our pool of FS methods. Hence, we have six FS methods, namely CM1 score, CM2 score, Information Gain (IG), Gain Ratio (GR), Correlation (CR) and Symmetric Uncertainty (SU) to use in the proposed method.

Classification Performances: Now, we consider the classification performances of 39 base classifiers achieved for six filter-based FS methods. We used these FS methods

FS	arcene	dexter	dorothea	gisette	madelon
CM1	**0.29 ± 0.04**	0.87 ± 0.25	13.81 ± 0.69	1.22 ± 0.08	**0.03 ± 0.01**
CM2	0.41± 0.05	**0.75 ± 0.18**	**13.27 ± 0.59**	**1.21 ± 0.05**	0.04 ± 0.01
IG	0.63 ± 0.09	1.69 ± 0.18	27.04 ± 2.45	11.27 ± 0.45	0.30 ± 0.05
GR	0.41 ± 0.12	1.97 ± 0.22	32.53 ± 2.72	13.0 ± 0.54	0.33 ± 0.06
CR	0.40 ± 0.07	1.0 ± 0.26	14.14 ± 1.02	1.36 ± 0.08	0.04 ± 0.00
OR	*18.73 ± 1.25*	*50.47 ± 0.54*	936.90 ± 123.35	77.62 ± 4.43	2.86 ± 0.15
RF	2.38 ± 0.15	34.42 ± 0.88	*1096.79 ± 171.62*	*2476.22 ± 74.18*	*29.64 ± 0.27*
SU	0.30 ± 0.04	1.36 ± 0.05	32.76 ± 2.24	12.88 ± 0.61	0.37 ± 0.07

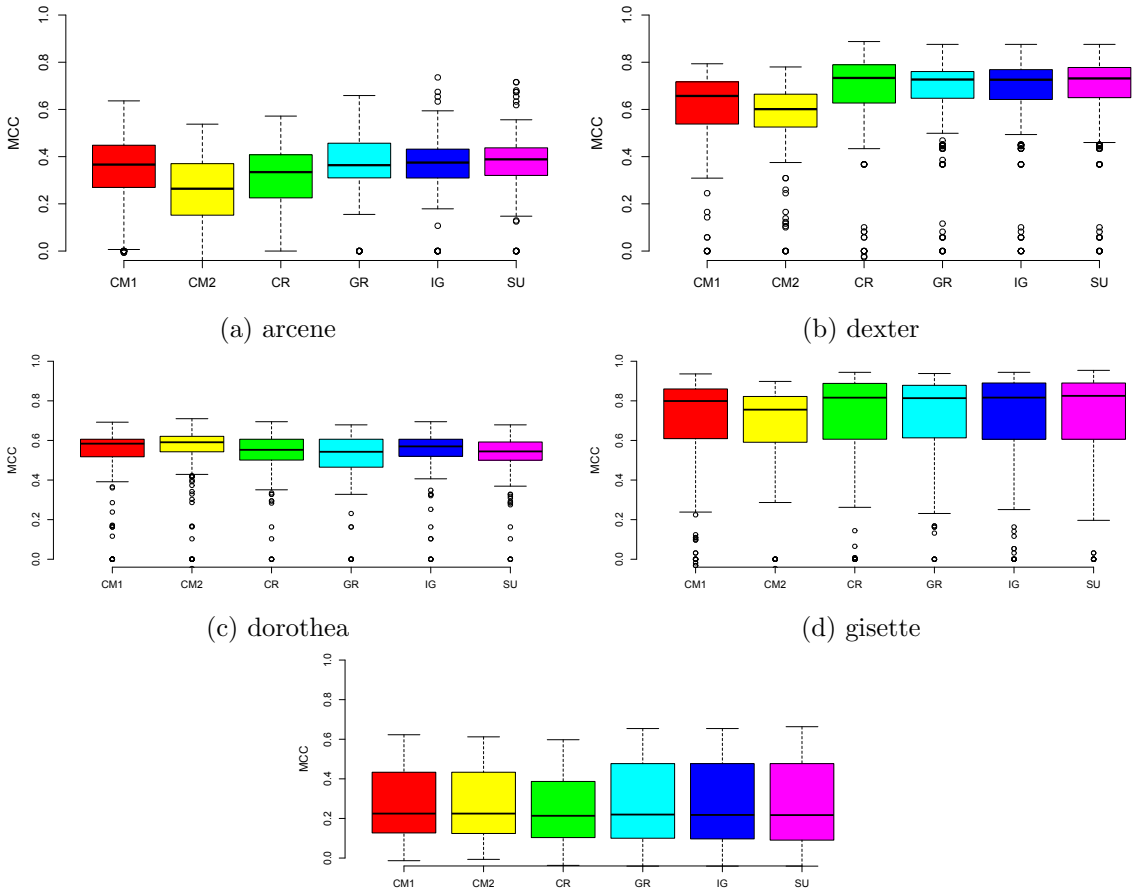
Table 5.7: Summary of running times required to select features from five datasets by all feature selection methods. The average (± standard deviation) is calculated for the CPU times (in s) spent in selecting the top 100, 200, 300, 400 and 500 features.

to select the different number of features from the datasets and we report the generalisation performances of base classifiers by the number of features used in their training phase. The summary of MCC scores achieved by base classifiers for different FS methods are shown in Table 5.8. For better visualising of the experimental outcomes, the classification performances achieved for the different number of features by the FS methods for all datasets are shown using box plots in Figure 5.2. From the box plots of each dataset, we can see that there is not a single FS method that performed as the worst in all cases. In contrast, we observe that the six selected FS methods performed almost the same for all datasets. Hence, we keep all six FS methods selected from the previous step.

FSMethod	Min	Avg	sd	Max
CM1	-0.12	0.50	0.25	0.94
CM2	-0.12	0.46	0.25	0.90
CR	-0.04	0.51	0.27	0.94
GR	-0.04	0.52	0.25	0.94
IG	-0.04	0.52	0.26	0.94
OR	0.00	0.54	0.25	0.94
RF	-0.06	0.49	0.24	0.95
SU	-0.04	0.52	0.26	0.95

Table 5.8: Classification performances summary (minimum (Min), average (Avg), standard deviation (sd) and maximum (MAX) MCC scores) for different feature selection methods for all experimental datasets.

After assessing the computational outcomes, the list of selected FS methods remaining in the FS pool is shown in Table 5.9.



(a) arcene

(b) dexter

(c) dorothea

(d) gisette

(e) madelon

FIGURE 5.2: Box-and-whisker plots of classification performances achieved using different feature selection methods for dataset (a) arcene, (b) dexter, (c) dorothea, (d) gisette and (e) madelon.

5.2.3 Selection of Base Classifiers

We took a total of 39 heterogeneous base classifiers available in the WEKA data mining software, version 3.7 [Hall et al., 2009]. The heterogeneity of those base classifiers with their count is shown in Table 5.5. To select the pool of base classifiers for the ensemble, we will consider their computational results.

The selection of base classifiers for the EoC will be carried out depending on the following criteria. If any base classifier failed to pass any one of those criteria, they would not be included in the base classifier pool. The selection criteria of base classifiers will be evaluated on the prediction performances achieved for datasets in Table 5.4.

FS Method	Short Name
Craig–Moscato's Feature Ranking 1	CM1
Craig–Moscato's Feature Ranking 2	CM2
Information Gain Attribute Evaluation	IG
Gain-Ratio Attribute Evaluation	GR
Correlation-based Attribute Evaluation	CR
Symmetric Uncertainty Attribute Evaluation	SU

Table 5.9: List of feature selection methods selected for experiments.

Classification Performances: Among various available classification performance measures, we will consider the MCC score. If a base classifier's Median (MCC) score $\leq$ 1st Quartile (Q1) of the base classifiers' MCC score for all experiments, we will remove that base classifier from the pool. To evaluate this selection criterion, we plot the MCC scores achieved by base classifiers in Figure 5.3. It is evident from the figure that the `LibSVM`, `KStar`, `HyperPipes` and `OLM` classifiers median MCC scores lie below the overall median MCC achieved by all base classifiers. Those classifiers will not be considered for inclusion in the base classifier pool.

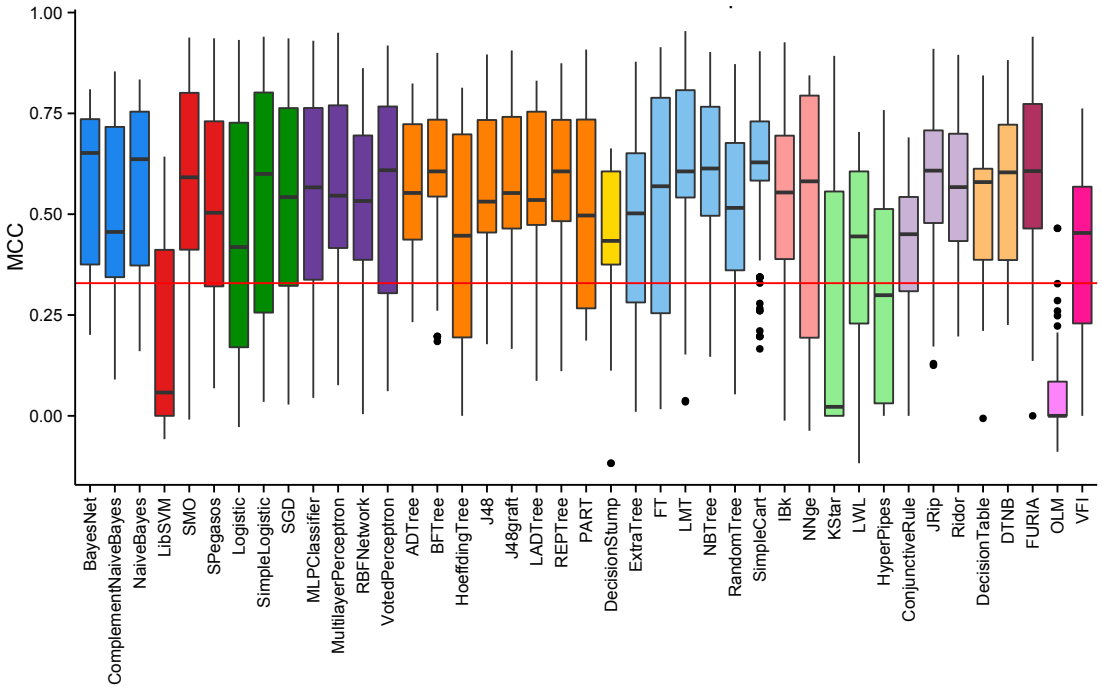


Figure 5.3: Box-and-whisker plot of classification performances (in MCC scores) achieved by base classifiers in the testing set. The red horizontal line shows the 1st Quartile (Q1) MCC score obtained by all base classifiers for all experiments.

Running Time: Running time of a base classifier is the total CPU time required for model building and prediction. Required running time per base classifier has a direct impact on the overall time requirement for evaluation of an ensemble combination. If any base classifier uses excessive CPU time for model building and evaluating, then we will remove that base classifier. To assess this criterion, we have plotted all running times (cropped to 600 seconds) required by each base classifier in Figure 5.4 per FS approach. The stacked bar plot in the figure shows the total running time required for all tests per FS methods. Here, we can observe that `MultilayerPerceptron`, `NBTree`, `NNge`, `Kstar`, `LWL`, `DTNB` and `FURIA` require more than 300 seconds running time independent of FS methods. We remove these seven resource-hungry base classifiers from our pool.

Type	**Count**	**Base Classifiers**
Bayes	2	`BayesNet` `NaiveBayes`
Support Vector Machine	2	`SMO` `SPegasos`

Linear	3	`Logistic`
		`SimpleLogistic`
		`SGD`

Neural Network	3	`MLPClassifier`
		`RBFNetwork`
		`VotedPerceptron`

Decision Tree	7	`ADTree`
		`BFTree`
		`HoeffdingTree`
		J48
		`LADTree`
		`REPTree`
		`PART`

Trees	5	`ExtraTree`
		`FT`
		`LMT`
		`RandomTree`
		`SimpleCart`

Nearest Neighbours	1	`IBk`

Rule Learner	3	`ConjunctiveRule`
		`JRip`
		`Ridor`

Decision Table	1	`DecisionTable`
Decision Stump	1	`DecisionStump`
Feature Interval	1	`VFI`

TABLE 5.10: List of the 29 selected base classifiers to be used for the experiments.

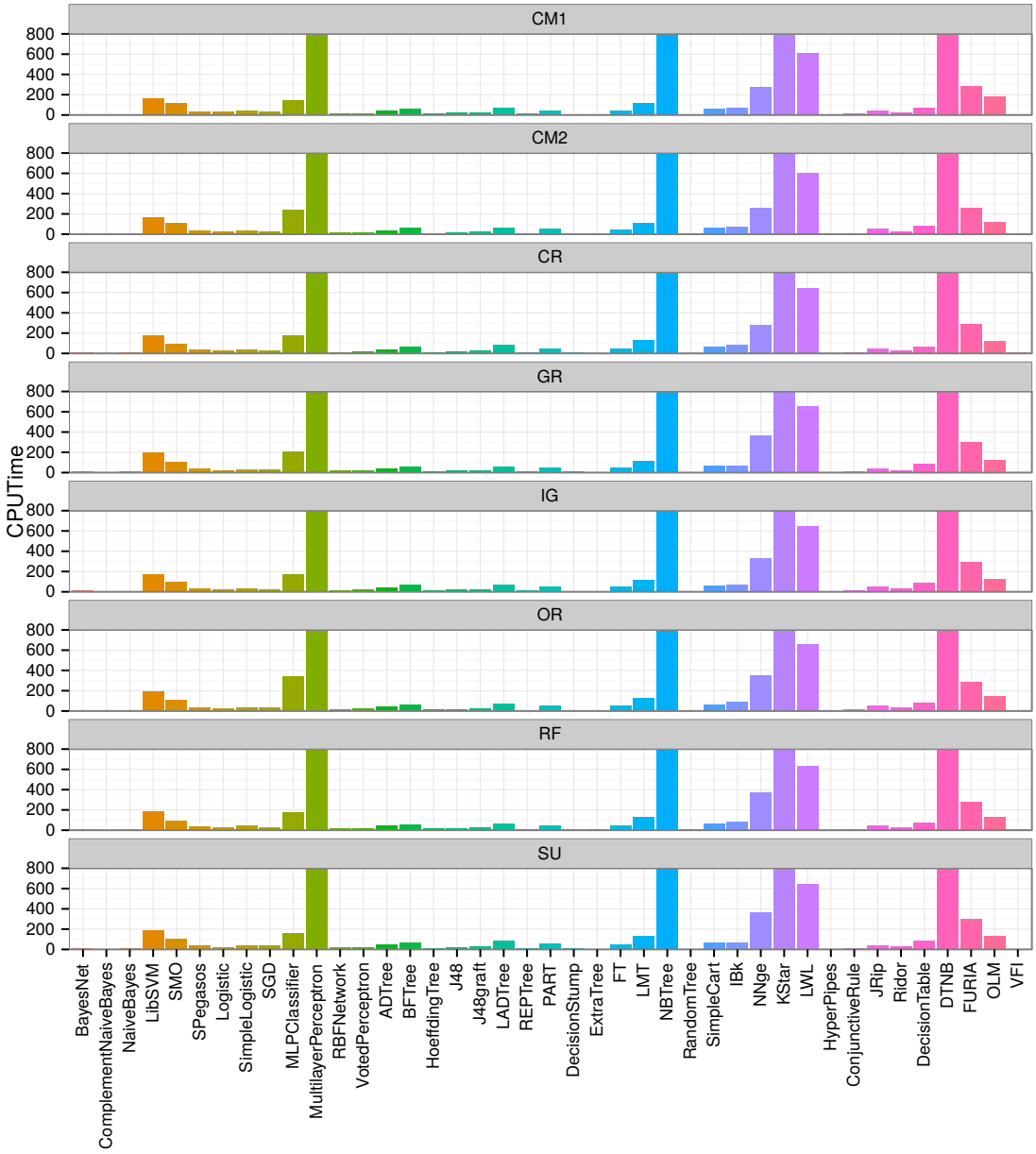


Figure 5.4: Stacked bar plot shows the base classifiers CPU time (cropped to 800 seconds) required for model building and validating per feature selection methods.

After considering those criteria, we have 29 base classifiers in the pool (shown in Table 5.10). We will use them for further computational experiments.

5.2.4 Objective Selection for Multi-Objective Optimisation

Most of the research on ensemble generation using multiple objectives is confined to a single scalar function using hyperparameters [Jin and Sendhoff, 2008]. It is common practice to tune the hyperparameter (λ) to balance between two commonly used objectives, accuracy (Obj_{acu}) and diversity (Obj_{div}) to obtain the scalar value of objectives Obj_{Sc} as:

$$Obj_{Sc} = Obj_{acu} + \lambda Obj_{div}. \tag{5.2}$$

The conversion of multi-objective into single objective usually introduces some weakness. A predefined hyperparameter is required to be set to achieve the right balance between accuracy and diversity. Moreover, this outcome transformed into single-objective approach is limited to a single classifier model. We do need the multi-objective problem to deal with multiple conflicting objectives (such as accuracy and diversity) optimisation. Best individuals from a multi-objective evolution will form a front called the Pareto front, of a size equal to the number of objectives.

Definition of Objective Functions

In our case, we will analyse three commonly used objective values in the literature, namely classification performance measure, diversity score and ensemble size. These objective values conflict with each other. Let us define these objective functions for our multi-objective ensemble of classifier (MO-EoC).

Classification Performance Measure: In the case of the classification performance measure, we will maximise the average MCC score for the EoC [Haque et al., 2016a]. Hence, the objective function can be written as:

Objective 5.1. ***MCC:*** *Given an ensemble $\mathbb{E}$ with k base classifiers. For each base classifier $\mathbb{C}_i$ in the ensemble,*

$$\begin{aligned} (\textbf{Maximise}) \quad Obj_{mcc}(\mathbb{E}) : \frac{1}{k}\sum_{i=1}^{k} MCC\left(\mathbb{C}_i\left(\mathbb{T}_s\right) \in \mathbb{E}\right) \\ subject\ to \quad MCC\left(\mathbb{E}\right) \geq 0.0, \end{aligned} \tag{5.3}$$

where the MCC score is calculated using a 60–40 split on the selected feature subset $\mathbb{T}_s$. A valid solution should not perform worse than a random predictor (with MCC score equal to at least 0.0).

The outcome of the objective will find the ensemble combination providing maximum average MCC score. Hence, the optimisation of Obj_{mcc} function will be a maximisation problem.

Diversity Measure of Base Classifiers: In the classification problem, the diversity score represents the agreement or disagreement among the base classifiers' decisions. The most diverse classifiers will predict different labels for the same input data. In the literature about the EoC, the diversity among base classifiers is not well defined as classification performance [Zhou and Li, 2010]. Further, [Kuncheva and Whitaker, 2003] experimented with available diversity measures and found some of them may mislead the classification learning. They categorised the diversity measures into two categories: pairwise and non-pairwise diversity.

In the case of pairwise diversity measures, the score is calculated for the pair of base classifiers. The calculation of pairwise diversity measure becomes computationally expensive for ensembles created with a large number of base classifiers and also for large datasets. Q-statistics, correlation coefficients and k-statistics are commonly used pairwise diversity scores [Ahmadian et al., 2007, Santana et al., 2006]. Conversely, a non-pairwise diversity score is calculated over a set of classifiers for training performance. The non-pairwise diversity measures can be calculated easily as a group for a large number of base classifiers than the calculation of pairwise measures considering each pair of base classifiers. Kohavi–Wolpert variance, inter-rater agreement, generalised diversity and entropy are commonly used non-pairwise measures of diversity [Chiu and Verma, 2014, Rahman and Verma, 2013]. In our case, we have 29 heterogeneous base classifiers, which is so far the largest EoC.

We will use the widely used entropy score for EoC as our measure of diversity [Kuncheva and Whitaker, 2003]. The value of diversity for an ensemble $\mathbb{E}$ on training dataset $\mathbb{T}_s$ is calculated as:

$$\frac{1}{m}\sum_{j=1}^{m}\left(\frac{1}{k-\lceil\frac{k}{2}\rceil}\right)\min\{l(z_j), k-l(z_j)\}, \tag{5.4}$$

where, k denotes the number of base classifiers in the ensemble, m is the number of samples in the training dataset, $l(z_j)$ denotes the number of base classifiers that recognise a sample z_j correctly and $\lceil\frac{k}{2}\rceil$ denotes the votes from base classifiers with the same class label.

Objective 5.2. ***Diversity:*** *Given an ensemble combination* $\mathbb{E}$ *with* k *base classifiers, the diversity is defined as:*

$$(\textbf{Maximise}) \quad Obj_{div}(\mathbb{E}) : Diversity\left(\mathbb{C}_i\left(\mathbb{T}_s\right) \in \mathbb{E}\right), \tag{5.5}$$

where the Diversity score is calculated on the training portion $\mathbb{T}_s$ *of the dataset.*

An ensemble will be assessed for the entropy score attained by the combination of base classifiers. Hence, the optimisation of Obj_{div} function will also be a maximisation problem.

Ensemble Size: In MOO of ensemble classifiers, many researchers used ensemble size (the number of base classifiers in the ensemble combination) as one objective value [Ahmadian et al., 2007, Nguyen et al., 2014, Dos Santos et al., 2006, Dos Santos et al., 2008]. They used minimisation of ensemble size or cardinality as an objective function. The objective function dealing with the minimisation of size is defined as:

Objective 5.3. ***Size:*** *Given an ensemble combination* $\mathbb{E}$ *with* k *length of binary string representing the combination of base classifiers. The objective function for the ensemble size is written as:*

$$\begin{aligned}(\textbf{Minimise}) \quad Obj_{sz}(\mathbb{E}) : |\mathbb{E}| \mapsto \sum_{i=1}^{k} \mathbb{E}_i \left[\forall_i, \mathbb{E}_i = 1\right] \\ subject\ to \quad |\mathbb{E}| \geq 2,\end{aligned} \tag{5.6}$$

where the ensemble size mapped into the number of elements $\mathbb{E}_i$ *having nonzero value in the respective position in the string of base classifiers. To be considered a valid solution, it needs to have at least two base classifiers.*

The size of the ensemble should be kept to a minimum.

5.2.5 The MO-EoC Framework

An MOEA deals with a mathematical optimisation problem that requires optimisation of more than one criterion simultaneously. MOEAs have been applied successfully in many areas where optimal decisions need to be taken in the presence of trade-offs between two or more conflicting objectives. Unlike single-objective optimisation, MOEA provides a number of Pareto-optimal solutions. A solution is called non-dominating if it is impossible to improve one objective without degrading some other objective value. For this reason, all

Pareto-optimal solutions are treated as equally good. However, a final solution is selected from the Pareto-optimal solutions depending on the subjective preference (trade-offs in satisfying different objectives) of the decision maker.

NSGA-II

Among many MOO algorithms, we have chosen the widely used NSGA-II [Deb et al., 2002]. It is an upgraded version of the NSGA algorithm [Srinivas and Deb, 1994]. The NSGA-II, a GA-based MOEA, possesses several advantages over its predecessor. It improved the NSGA by adopting the fast non-dominated sorting approach and the crowded comparison operator, which helps the algorithm to work faster, facilitates an elitist principle, emphasises non-dominating solutions, maintains population diversity and converges near the Pareto-optimal solutions. Hence, NSGA-II is capable of simultaneously optimising the maximisation and minimisation of multiple objective functions and it turned into one of the most famous Pareto-optimal solution algorithms. It has been successfully applied in various optimisation problems, including recently in resource allocation [Martínez-Vargas et al., 2016], data classification [Mane et al., 2016], biomarker discovery [Vafaee, 2016] and biological network analysis [Zickenrott et al., 2016].

Algorithm 9 provides pseudocode of the NSGA-II for optimising multiple objectives. Initially, a parent population $\mathbb{P}$ with Pop_{sz} number of individuals is randomly initialised. Each individual encodes the problem with a Prb_{sz} size of string. Then, genetic variation operations, namely *selection*, *recombination* and *mutation*, are applied to generate offspring population $\mathbb{Q}$ with the size of Pop_{sz}. The iteration of the algorithm starts with the merging of both populations into one population ($\mathbb{MP}$). Then, the combined population is transferred into the non-dominated sorting procedure with the `FastNon-dominatedSort` function (as shown in Algorithm 10). The `SortByRankAndDistance` function aligns the population into a hierarchy of non-dominated Pareto fronts. It assigns the best rank to a solution that is not dominated by any other solution in the Pareto fronts. The average distance between individuals in each front is calculated using `CrowdingDistanceAssignment` described in Algorithm 11. Then, a function for discriminating individuals in the population is used according to their rank that preserves elitism. The `SelectParentsByRankAnd-Distance` is used for ordering Pareto front solutions first by their dominance precedence and then by the distance within the front. Then, a new population of offspring is generated after applying genetic variation operators (recombination and mutation). These procedures are repeated until the stopping criteria are satisfied.

Algorithm 10 shows how we transform the population into a non-dominated sorting population. The crowding distances of the non-dominating solutions and their ranks are

Algorithm 9: Pseudocode of NSGA-II algorithm.

Input: Population Size Pop_{sz}, Individual String length Prb_{sz}, Recombiantion Rate R_χ, Mutation Rate R_μ
Output: New Population $\mathbb{Q}$

```
   //Initialise the Population ℙ
1  ℙ ← InitializePopulation(Pop_sz, Prb_sz)
   //Evaluate Population
2  EvaluateAgainstObjectiveFunctions(ℙ)
3  FastNon-dominatedSort(ℙ)
4  𝕊 ← SelectParentsByRank(ℙ, Pop_sz)
5  ℚ ← RecombinationAndMutation(𝕊, R_χ, R_μ)
6  while ¬StopCondition() do
7      EvaluateAgainstObjectiveFunctions(ℚ)
8      𝕄ℙ ← Merge(ℙ, ℚ)
9      𝔽 ← FastNon-dominatedSort(𝕄ℙ)
10     Parents ← ∅
11     F_L ← ∅
12     foreach F_i ∈ 𝔽 do
13         CrowdingDistanceAssignment(F_i)
14         if Size(Parents)+Size(F_i) > Pop_sz then
15             F_L ← i
16             Break()
17         else
18             Parents ← Merge(Parents, F_i)
19     if Size(Parents)<Pop_sz then
20         F_L ← SortByRankAndDistance(F_L)
21         for P_1 to P_{Pop_sz − Size(F_L)} do
22             Parents ← Pi
23     𝕊 ← SelectParentsByRankAndDistance(Parents, Pop_sz)
24     ℙ ← ℚ
25     ℚ ← RecombinationAndMutation(𝕊, R_χ, R_μ)
26 return ℚ
```

calculated and preserved (lines 4–13). The ranking process of each solution (shown in lines 8–10) in the population is calculated based on their dominance depth. Solutions in the population obtain the value of *rank* and *crowdingDistance* attributes.

The process of `CrowdingDistanceAssignment` is shown in Algorithm 11. The calculation of crowding distance is restricted by the size of the non-dominated solution front.

Algorithm 10: Pseudocode of FastNon-dominatedSort.

Input: The population $\mathbb{P}$
Output: Fronts $\mathbb{F}$

```
1  foreach ℙ_i ∈ ℙ do
2      ℝ.Add(ℙ_i)
3  rank ← 1
4  while ℝ ≠ ϕ do
5      𝔽 ← ϕ
6      foreach ℝ_i ∈ ℝ do
7          𝔽.Add(ℝ_i)
8      foreach F_i ∈ 𝔽 do
9          ℝ.Remove(F_i)
10         F_i.setRank(rank)
11     𝔽 ← CrowdingDistanceAssignment(𝔽)
12     rank ← rank +1
13     ℙ.Add(𝔽)
14 return ℙ;
```

It returns a positive infinity value for front size ≤ 3 (lines 2–4). Otherwise, we initialise each solution in the fronts with zero distance. For each objective, the crowding distance is calculated for the front (lines 9–20). Here, the front ($\mathbb{F}$) is sorted according to the objective value. For each solution in the front, we update the crowding distance based on its neighbourhood (lines 16–19). In general, solutions that are far away (not crowded) from other solutions are given a higher rank. The ranking is produced this way to generate a diverse solution set. The procedure returns the front with the associated crowding distances based on the objectives.

The parent selection method based on the rank and distance is shown in Algorithm 12. The parent selection process starts with a randomly selected solution. Then, it is compared against other *arity*-sized randomly selected candidates (lines 3–7) for rank and distance. The winning solution is preserved in the population of parents. The process is repeated until the number of parents has reached the arity value.

The MO-EoC Using NSGA-II

In our design, we selected 29 base classifiers (listed in Table 5.10) from the WEKA data mining software suite [Hall et al., 2009] to create the ensemble combinations. The method named `EvaluateAgainstObjectiveFunctions` will evaluate each pair of objectives listed

Algorithm 11: Pseudocode of CROWDINGDISTANCEASSIGNMENT.

Input: Input Fronts $\mathbb{F}$
Output: Output Fronts $\mathbb{F}$

```
1  n ← Size(𝔽)
2  if n ≤ 3 then
3      foreach F_i ∈ 𝔽 do
4          F_i.setDistance(+∞)
5  else
6      nObj ← getNumOfObjs(F_1)
7      foreach F_i ∈ 𝔽 do
8          F_i.setDistance(0)
9      foreach Obj_i ∈ nObj do
10         𝔽.Sort(Obj_i)
11         Obj_min ← F_1.getObj(Obj_i)
12         Obj_max ← F_n.getObj(Obj_i)
13         F_1.setDistance(+∞)
14         F_n.setDistance(+∞)
15         foreach F_i ∈ 𝔽 do
16             Dist_cr ← F_i
17             Dist_Range ← (Obj_max − Obj_min )
18             Dist_nbr ← ( F_{i+1}.getObj(Obj_i) − F_{i−1}.getObj(Obj_i) )
19             Dist_cr ← Dist_cr +( Dist_nbr /Dist_Range )
20             F_i.setDistance(Dist_cr)
21 return 𝔽;
```

in Section 5.2.4. We use a binary string to represent a solution and to encode a pair of objectives inside the MOO algorithm NSGA-II. The NSGA-II implementation is used from the `MOEA Framework`, version 2.7, available from `http://moeaframework.org/` [Hadka, 2014]. We use the binary presentation of the solution with `Half-uniformRecombination` (HUX) and `BitFlip` (BF) mutation operators as variation functions of the NSGA-II algorithm. In the case of the HUX recombination operation, half of the non-matching bits are swapped in between the two parents. The same bit-flip mutation operator shown in Algorithm 4 is used here. The probability rates of mutation (R_μ) and recombination (R_χ) are kept unchanged from the default value of the `MOEA Framework`. Table 5.11 shows the parameter values used for the execution of the MO-EoC algorithm.

Algorithm 12: Pseudocode of SelectParentsByRankAndDistance.

Input: Population $\mathbb{P}$, Tournamnet Size *arity*
Output: Parent Population Parents

```
for si ← 1 :arity do
    ℙ_win ← getRandomSoln(ℙ)
    for i ← 1 : Size(arity) do
        ℙ_cand ← getRandomSoln(ℙ)
        flag ← CompareRankDist(ℙ_win, ℙ_cand)
        if flag > 0 then
            ℙ_win ← ℙ_cand
    Parents_si ← ℙ_win
return Parents;
```

Parameter	Value
Individual Type	binary string
Individual length	29
Population Size	100
Maximum Evaluation	10000
Recombination Strategy	HUX
Recombination Rate (R_χ)	0.75
Mutation Strategy	BF
Mutation Rate (R_μ)	0.10
A pair of Objectives	{ Obj_{mcc}, Obj_{div} , Obj_{sz}}

Table 5.11: Parameter settings of the proposed multi-objective ensemble of classifier using NSGA-II.

5.2.6 Computational Experiments

We use the NSGA-II algorithm to optimise multiple objectives in pairs taken from the objectives of $\{Obj_{mcc}, Obj_{div}, Obj_{sz}\}$. In each execution setup, a pair of objectives is evaluated for the Pareto front comprised of non-dominated solutions. Then, we evaluate the goodness of those solutions to decide the best objective pairs to use in the MO-EoC.

Datasets

The performances of MO-EoC for different pairs of objective optimisations in the multi-objective EoC combination search are evaluated on the benchmarking datasets taken from UCI-ML and AD shown in Table 5.12. Details about these datasets have already been described in Section 3.3.2.

Short Name	Name of the Dataset	#Samps	#Feats
WBC	Wisconsin Breast Cancer (Original)	699 (458,241)	9
PIMA	Pima Indians Diabetes	768 (500,268)	8
BUPA	BUPA Liver Disorders Data Set	345 (145,200)	7
Ray-AD-Trn-18	Ray et al. - AD (18 Protein)	83 (43,40)	18
RMoscato-AD-Trn-5	Ravetti & Moscato - AD (5 Protein)	83 (43,40)	5

TABLE 5.12: Characteristics of the datasets used for experiments of selecting objectives in multi-objective ensemble of classifiers.

Computational Results

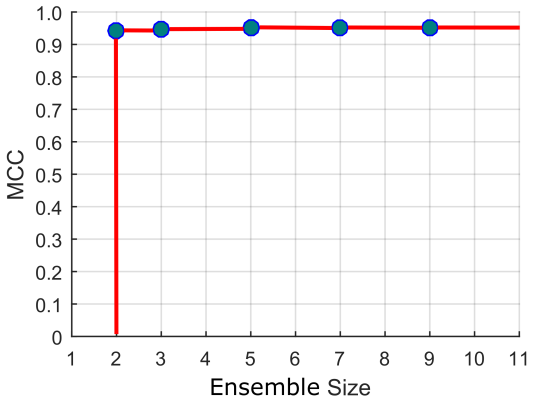
The outcome of MO-EoC is measured in pairs of objective evaluations. We wanted to know the effect of objectives on the generalisation outcome. Pairwise evaluation of them will give us an opportunity to explore the impact of objectives in a pair. Consequently, we can choose our pair of objectives to be used with the MO-EoC.

MCC vs Size

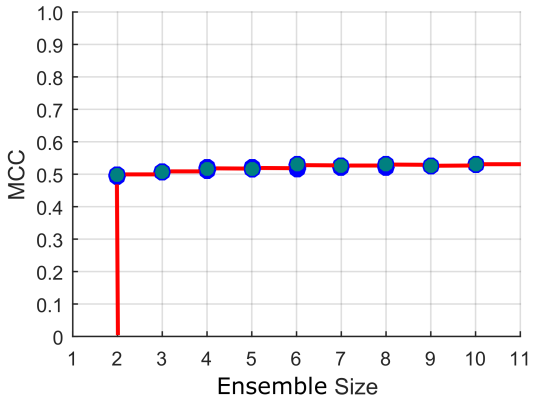
Now, we will evaluate the objective pairs (Obj_{mcc}, Obj_{size}) for our benchmark datasets described in Table 5.12. The values of objective pairs for each of the Pareto-optimal solutions for the datasets are plotted in Figure 5.5. Here, the X-axis expressed the ensemble size and the Y-axis denoted the MCC score of the non-dominating solution.

Figure 5.5a shows the scatter plot of objective scores pair (Obj_{mcc}, Obj_{size}) for the *WBC* dataset. Here, all of the solutions achieved approximately 0.90 MCC scores independent of the change in ensemble size (varies between 2 to 10). For the *PIMA* dataset, the solutions for optimising (Obj_{mcc}, Obj_{size}) are plotted in Figure 5.5b. The MCC score values of Pareto-optimal solutions varied between 0.35 and 0.55, irrespective of the size. In Figure 5.5c, the Pareto-optimal solutions for optimising (Obj_{mcc}, Obj_{size}) are shown for the *BUPA* dataset. In this case, the MCC scores remain steady for change in ensemble size. For the AD datasets, solutions were plotted for *RMoscato-AD-Trn-18* in Figure 5.5d. Only two sizes of ensemble (formed with two and three base classifiers) have been observed in the solution with two different MCC scores (0.70 and 0.85 for sizes two and three, respectively). Figure 5.5e shows a plot for changes in MCC scores for increasing the ensemble size of Pareto-optimal solutions. We observe no significant improvement of MCC when the size increased.

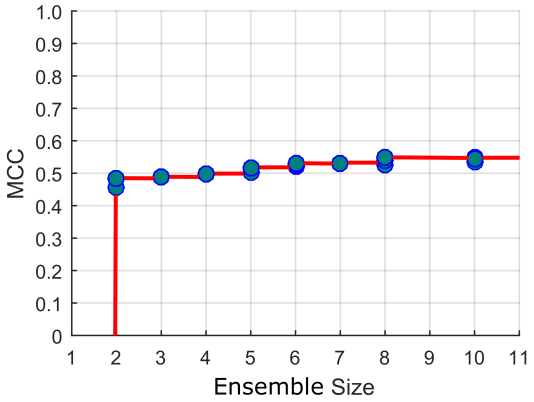
For these datasets, we found that the ensemble size and MCC score are not conflicting objectives and pairing them together did not produce any advantages to the enhancement of MCC scores. No previous study has ever used MCC scores as a performance measure in



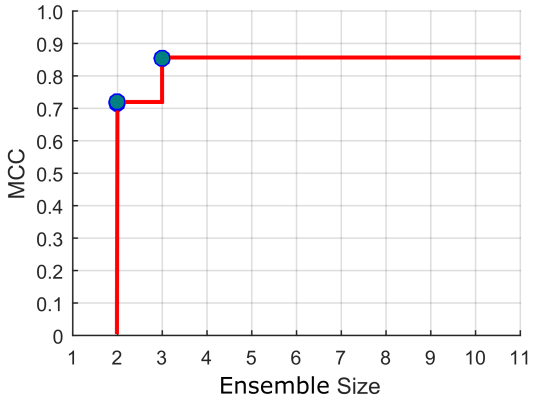
(a) WBC



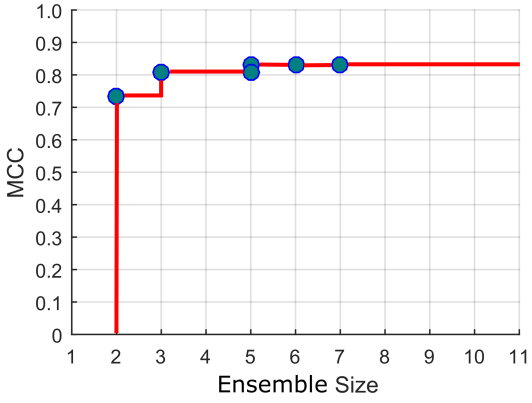
(b) PIMA



(c) BUPA



(d) Ray-AD-Trn-18



(e) RMoscato-AD-Trn-5

Figure 5.5: Scatter plots showing the Pareto-optimal solutions for optimising the objectives pair of (Obj_{mcc}, Obj_{size}) on (a) WBC, (b) PIMA, (c) BUPA, (d) RMoscato-AD-Trn-18 and (e) RMoscato-AD-Trn-5 datasets.

the MO-EoC. We observe for pairwise optimisation of MCC score and ensemble size, the change of MCC score for increasing the ensemble size is always parallel to the horizontal axis. Hence, the MCC score does not adhere to the conflicting objective of ensemble size in MOO. Therefore, the optimisation for the pair of objectives (Obj_{mcc}, Obj_{size}) will not be considered further in the MO-EoC.

Diversity vs Size

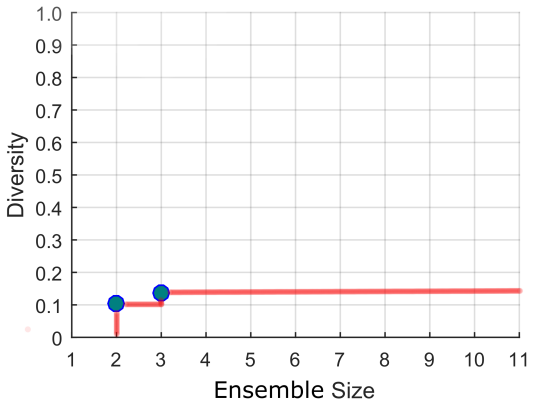
Now, we will evaluate the objective pairs (Obj_{div}, Obj_{size}) for our benchmark datasets described in Table 5.12. The values of objective pairs for each of the Pareto-optimal solutions for the datasets are plotted in Figure 5.6. Here, the X-axis expresses the ensemble size and the Y-axis denotes the diversity value of the non-dominating solutions.

Figure 5.6 shows the scatter plot of objective score pairs (Obj_{div}, Obj_{size}) for all datasets used for the experiment. Here, all solutions grouped into the ensemble of sizes two and three with two different diversity scores. The size three ensembles produced slightly better diversity scores than the size two ensembles. This phenomenon was observed in our experimental outcomes for optimising the (Obj_{div}, Obj_{size}) pair. Hence, the outcome has mostly been biased by the minimisation of the ensemble size. The disagreement between the base classifiers' decision in the solution ensemble with lower size will have greater diversity score compared with the disagreement by a single base classifier in a large ensemble. [Aksela and Laaksonen, 2006] also observed that pairwise optimisation of diversity and ensemble size always leads to minimising the number of base classifiers. Our result confirms their observation. Moreover, the optimisation of (Obj_{div}, Obj_{size}) pairs does not lead to a better generalisation. Both the diversity and size without pairing with an objective related to the classification performance will not lead to a better classification outcome. Hence, optimisation for the pair of objectives (Obj_{div}, Obj_{size}) will not be considered further in the MO-EoC.

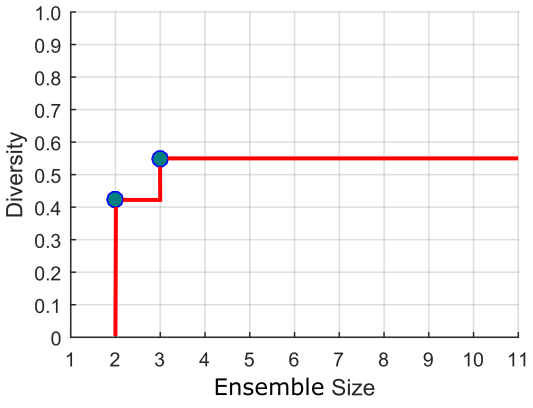
MCC vs Diversity

First, we evaluate the objective pairs (Obj_{mcc}, Obj_{div}) for our benchmark datasets described in Table 5.12. The values of objective pairs for each of the Pareto-optimal solutions for the datasets are plotted in Figure 5.7. The X-axis expresses the Diversity score and the Y-axis denotes the MCC score achieved by a non-dominating solution.

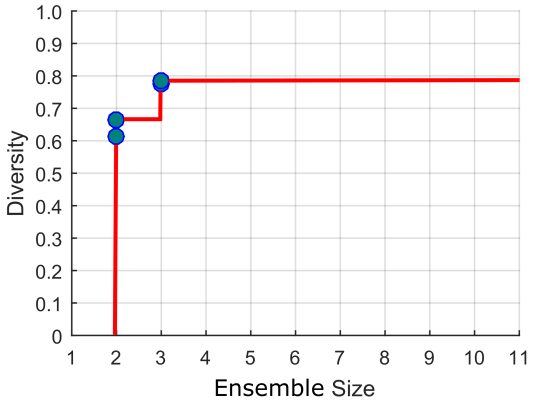
Figure 5.7a shows the scatter plot of objective score pairs (Obj_{mcc}, Obj_{div}) for the *WBC* dataset. Here, all of the solutions achieved approximately 0.90 MCC scores with the diversity scores in the range between 0.05 and 0.15. The WBC dataset classification



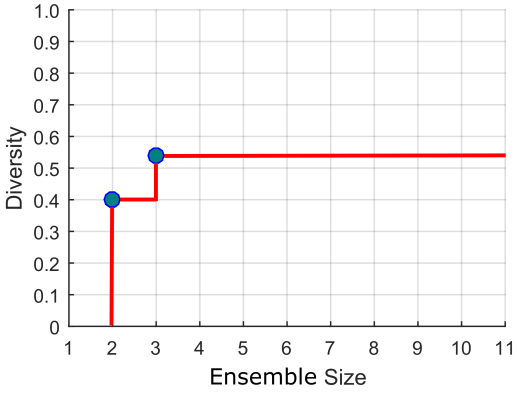
(a) WBC



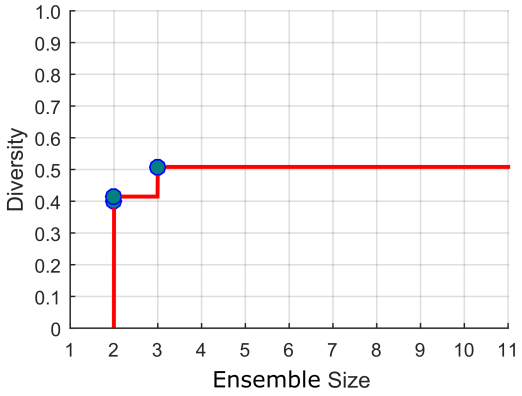
(b) PIMA



(c) BUPA

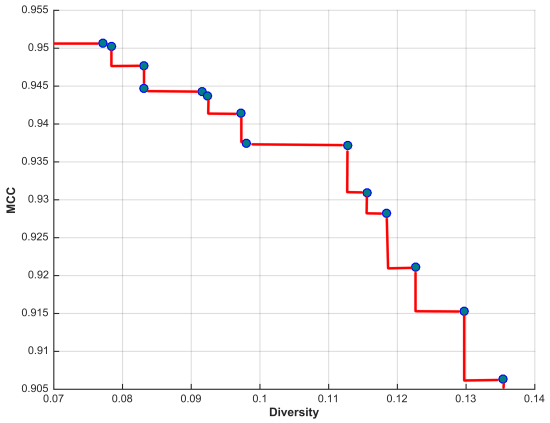


(d) Ray-AD-Trn-18

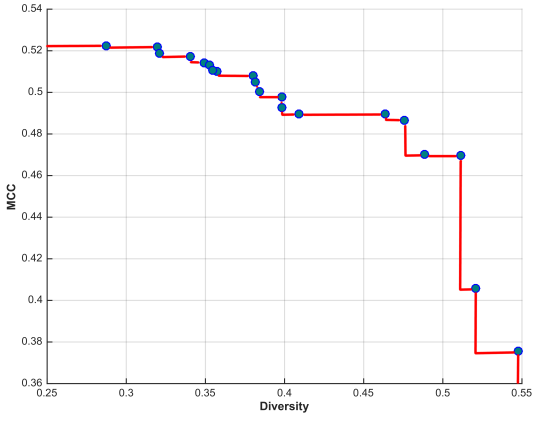


(e) RMoscato-AD-Trn-5

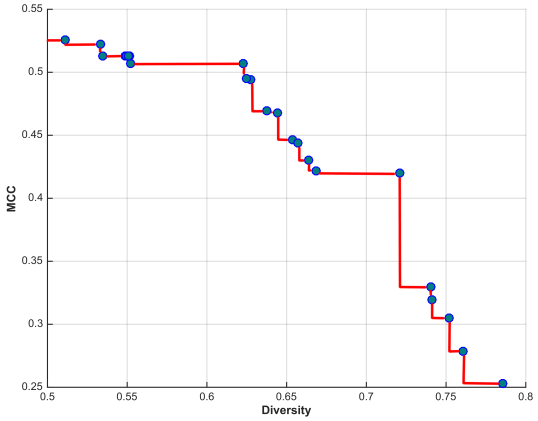
Figure 5.6: Scatter plots showing the Pareto-optimal solutions for optimising the objectives pair of (Obj_{div}, Obj_{size}) on the (a) WBC, (b) PIMA, (c) BUPA, (d) RMoscato-AD-Trn-18 and (e) RMoscato-AD-Trn-5 datasets.



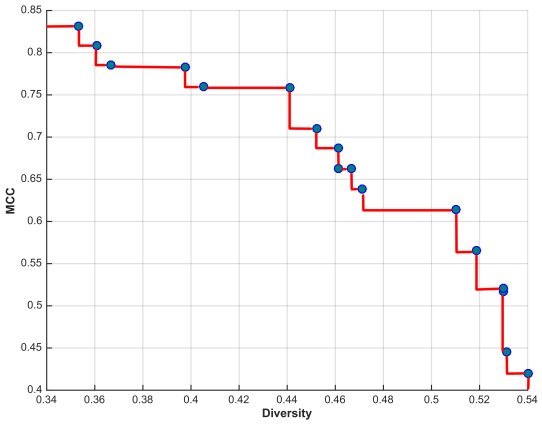
(a) WBC



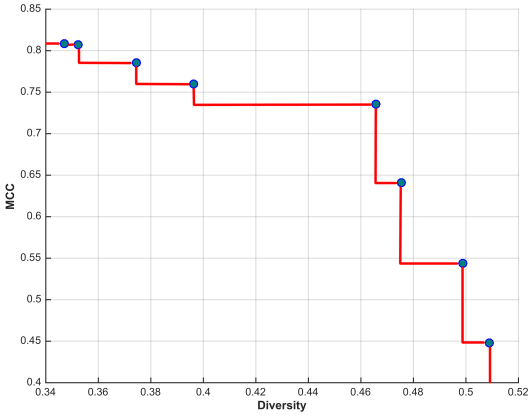
(b) PIMA



(c) BUPA



(d) Ray-AD-Trn-18



(e) RMoscato-AD-Trn-5

FIGURE 5.7: Scatter plots showing the Pareto-optimal solutions for optimising the objective pairs of (Obj_{mcc}, Obj_{div}) on the (a) WBC, (b) PIMA, (c) BUPA, (d) RMoscato-AD-Trn-18 and (e) RMoscato-AD-Trn-5 datasets.

problem is easier, so most of the solutions achieved similar MCC scores with lowest changes in diversity scores. For the *PIMA* dataset, the solutions for optimising (Obj_{mcc}, Obj_{div}) are plotted in Figure 5.7b. The diversity score varied between 0.20 and 0.55 for all solutions. The MCC score values of Pareto-optimal solutions varied between 0.35 and 0.55. In Figure 5.7c, the Pareto-optimal solutions for optimising (Obj_{mcc}, Obj_{div}) are shown for the *BUPA* dataset. Here, the diversity scores varied between 0.40 and 0.80 and the corresponding MCC scores decreased to within the range 0.55 to 0.15. For the AD datasets, solutions are plotted for the *RMoscato-AD-Trn-18* and *RMoscato-AD-Trn-5* datasets in Figure 5.7d and Figure 5.7e, respectively. The diversity score varied between 0.20 and 0.55 for both cases. Their MCC scores varied between 0.40 and 0.90. For these datasets, the slight increase of the diversity scores accompanied a sharp decrease in MCC score.

From these results, we can observe a common phenomenon of negative correlation between MCC scores with diversity in Pareto-optimal solutions. It is also noticeable that, for each dataset, the diversity score of solutions varies in a certain range. Most importantly, lower diversified solutions (within the range) produced better MCC scores. Thus, in the case of solution selection from the Pareto-optimal solutions (non-dominating solutions), we will choose a solution with higher MCC score but lower diversity value. From these outcomes, it is evident that the pair (Obj_{mcc}, Obj_{div}) is conflicting for MOO. We selected this pair of objectives to be used further in this thesis in Section 5.3.

5.2.7 Summary

The proposed method uses the GA to evaluate three pairs of objectives in multi-objective settings. The first pair was the optimisation of the MCC and diversity scores in the ensemble combinations. The two others were the optimisation of the ensemble size and MCC scores and the optimisation of the ensemble size and diversity scores. We have validated our selection of objective from the outcomes achieved in several real-world benchmarking datasets. The experimental outcomes revealed that optimisation of the ensemble's diversity and MCC score is the most suitable objective pair for the multi-objective ensemble of classifier combination search. The ensemble size does not have a significant effect on optimising the generalisation performances of the EoC in pairs with other objectives. So, we are only considering the EoC combination search in a MOO setup with the pair of (Obj_{mcc}, Obj_{div}) as best objectives for MO-EoC.

5.3 MO-EoC for Wrapper Feature Selection

FS is a crucial preprocessing procedure for high-dimensional data classification. Often FS methods can improve the generalisation performance of classifiers. The benefits of the FS method are several fold and it is an important process for biomarker discovery.

The techniques used for FS are mainly classified as the *filter*, *wrapper* and *embedded* approaches. The filter and embedded methods are relatively computationally inexpensive and commonly used. In contrast, wrapper methods iteratively evaluate a set of features with a classification method as an induction algorithm. Hence, Wrapper FS (WFS) methods are computationally expensive FS approaches. However, they have several advantages over other FS methods. They evaluate a set of features as a whole, unlike the alternative approaches that rank each feature based on their own merit like some filter approaches. This process helps WFS to learn vital relationships among multiple features. It is very important in biological datasets because a group of genes usually defines a disease in a better way than an individual gene and collectively contributes to the disease pathway. The iterative evaluation of features in WFS is more likely to discover the best set of features or biomarkers through an internal assessment by a classification algorithm. Hence, features selected by WFS are apparent to improve classification accuracy. As a consequence, WFS is famous for domain specialised data classification tasks where the performance is a crucial factor.

5.3.1 Literature Review of Ensemble of Feature Selection in Computing and Data Analysis

Many studies in the literature use GAs to search the conventional space created by the ensemble of FS methods. [Kuncheva and Jain, 2000] proposed two basic ways to utilise the power of GAs to design multiple classifier systems. Their approach started with a GA variation that selects disjoint feature subsets and the second variant selects an overlapping feature subset. The authors formed an ensemble with three-classifier systems and basic types of individual classifiers. [Oliveira et al., 2003] applied GAs to combine the predictions of different FS methods. They proposed an ensemble FS method based on a hierarchical multi-objective GA. In their first level, a set of robust classifiers is applied for FS. The next level of GA combines the features to build a powerful ensemble. The proposed method was used for handwritten digit recognition, using three different feature sets and neural networks (MLP) as classifiers. Experimental results showed the power of the proposed strategy. Later, [Minaei-Bidgoli et al., 2004] applied GA on FS to improve the prediction performance. First, they used the FS method to decrease computational cost and increase

classifier efficiency. The FS optimised the prediction accuracy of an ensemble using GA. Their approach was more efficient than a feature subset selection method, adaptable to analyse different attributes. Subsequently, [Oliveira et al., 2005] also treated FS using GAs for classification ensemble building in a similar manner to [Oliveira et al., 2003]. They experimented on the handwritten digit recognition problem and compared with a traditional `Bagging` and `Boosting` approach. The proposed method produced higher prediction accuracy.

Conversely, many researchers used different types of FS methods to create an ensemble to reduce the dimensionality problem in the dataset. Some researchers, such as [Cunningham and Carney, 2000], argue that feature subset selection has emerged as a useful technique for creating diversity in classification ensembles. They proposed an entropy measure of the outputs of the ensemble members as a useful measure of the ensemble diversity and evaluated this on a medical prediction problem. Experimental results showed the enhanced prediction performance of the ensembles and the entropy measure of diversity. In addition, it expressed a relationship with the change in diversity and breadth of the ensembles. Similarly, [Blanco et al., 2004] proposed an ensemble-based wrapper method for gene selection and classification in gene expression datasets. They employed the naive Bayes classification algorithm in a wrapper form. Their design reduces the number of genes selected with similar accuracy than conventional approaches. [Nikulin et al., 2009] introduced an ensemble that is capable of greater prediction accuracy than any of their individual members utilising selected features. They found a large number of relatively small and balanced subsets where representatives from the larger pattern are to be selected at random. They tested the ensemble strategy against datasets of the PAKDD-2007 data mining competition. Their ensemble method produced higher accuracy than single classifiers.

Several works in the literature have reported a problem arising from the class imbalance nature of some datasets. [Duangsoithong and Windeatt, 2010] presented a bootstrap FS for ensemble classifiers to manage the imbalance nature of datasets. They first selected optimal features from the full dataset before bootstrap-selected data. Then, they applied an ensemble classifier to evaluate the performance on a UCI machine learning repository and causal discovery datasets. The results showed that the bootstrap FS algorithm provides slightly better accuracy than the traditional FS for ensemble classifiers. [Yang et al., 2010b] also proposed a similar ensemble of FS methods for imbalanced data classification. [Turhal et al., 2013] has implemented different ensemble methods for FS for classification of colon cancer. They evaluated the performance improvement of each individual with ensemble classification methods. Their proposed ensemble method reduced the feature dimension

and improved the classification accuracy for the colon cancer dataset. Recently, [Yang et al., 2013] proposed an ensemble-based wrapper method (FS) suitable for highly imbalanced class distributions. First, they eliminated the imbalance nature of the dataset by sampling and creating multiple stable subsets of actual data. Then, they evaluated feature subsets using an ensemble of base classifiers, each trained on a balanced dataset. Their test results indicate that an ensemble-based FS using the wrapper method outperformed the original wrapper algorithms for imbalanced class data. [Nag and Pal, 2016] proposed a MOO approach for the selection of features to create a tree-based classifier using GP. They selected a feature from the dataset using the MOO algorithm. Then, they created a GP-based tree for classification learning. The method exhibited better performance in most of the datasets.

According to these previous contributions, we can conclude that FS methods can be a potential approach for resolving inherent problems in the dataset. In the context of biological data classification, imbalanced class and dimensionality are two challenges for classification algorithms that can be eliminated using FS methods. An ensemble of FS methods can select better features from a large number of features and consequently improve the quality of classifier training. Moreover, all of the works considered either feature subset selection methods using an ensemble of features or EoC creation using multi-objective methods. It requires a new approach to investigate both the feature subset selection and ensemble combination selection within the same framework. It will give us valuable insights about the EoC using MOO with feature subset selection together.

5.3.2 The Proposed MO-EoC Wrapper FS Method

In general, a wrapper-based FS algorithm consists of three main components, namely a search algorithm, a fitness function and an inductive algorithm [Kohavi and John, 1997], as shown in Figure 5.8. Here, input features came from the training dataset. A search algorithm is used to find the best subset of features. The fitness function controls the search algorithm to reach the optimal point. The evaluation of fitness function is done with an induction algorithm (a classifier is commonly used as an induction algorithm). The feature subset search and evaluation process advanced iteratively. Finally, the outcome is the selected subset of features from the input features.

Probably, the structure of MO-EoC-based Wrapper FS (MO-EoC-WFS) adheres to the generic wrapper FS method. The proposed *MO-EoC-WFS* has more components than generic WFS algorithms. Here, we are using a multi-objective GA, NSGA-II, as the search algorithm. The maximisation of the MCC score and diversity are treated as two objectives to be optimised. For evaluation of the fitness value, the MO-EoC algorithm

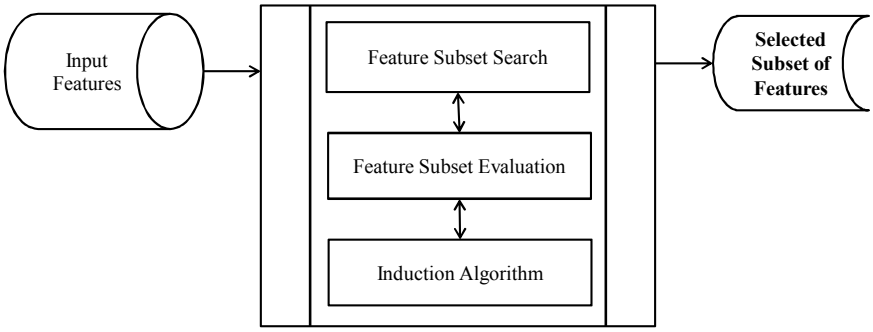


Figure 5.8: Generic architecture of a wrapper feature selection method (The figure is reproduced from [John et al., 1994] with permission Permission Number 3813910409023).

will evaluate the strength of feature subset selected by the wrapper using not only the generalisation capability (optimising the MCC score) but also optimising the diversity among base classifiers of the EoC. The structure of the proposed *MO-EoC-WFS* looks as in Figure 5.9.

5.3.3 Overview of MO-EoC-WFS

In this section, we will explain the elements of the GA. We will describe how the problem contexts are represented by the components of the GAs and how the evolution takes place. NSGA-II, a non-dominating sorting multi-objective *GA* has been used to search for the optimal combination of feature subsets (selected by a participating FS method) and finding the best set of base classifiers for creating the ensembles. NSGA-II has already been used successfully for large and complex optimisation problems. In our design, we selected 29 base classifiers (listed in Table 5.10) from the WEKA data mining software suite [Hall et al., 2009] to create the ensemble combinations (see Section 5.2.3). Based on the running time and generalisation ability in data classification, we selected six FS methods (see Section 5.2.2). The multi-objective-based EoC searching and feature subset selection using a wrapper approach will be denoted as *MO-EoC-WFS* (Figure 5.9). The components of MO-EoC-WFS are as follows:

Individual Representation: The efficiency and runtime of a GA depend on the **representation** of the individual and associated fitness function. This fitness function evaluates each individual within the population. We have used *39-bit binary encoding* for representation of the individual shown in Figure 5.10, consisting of three parts. The binary-coded decimal value from the first three bits represents the FS method.

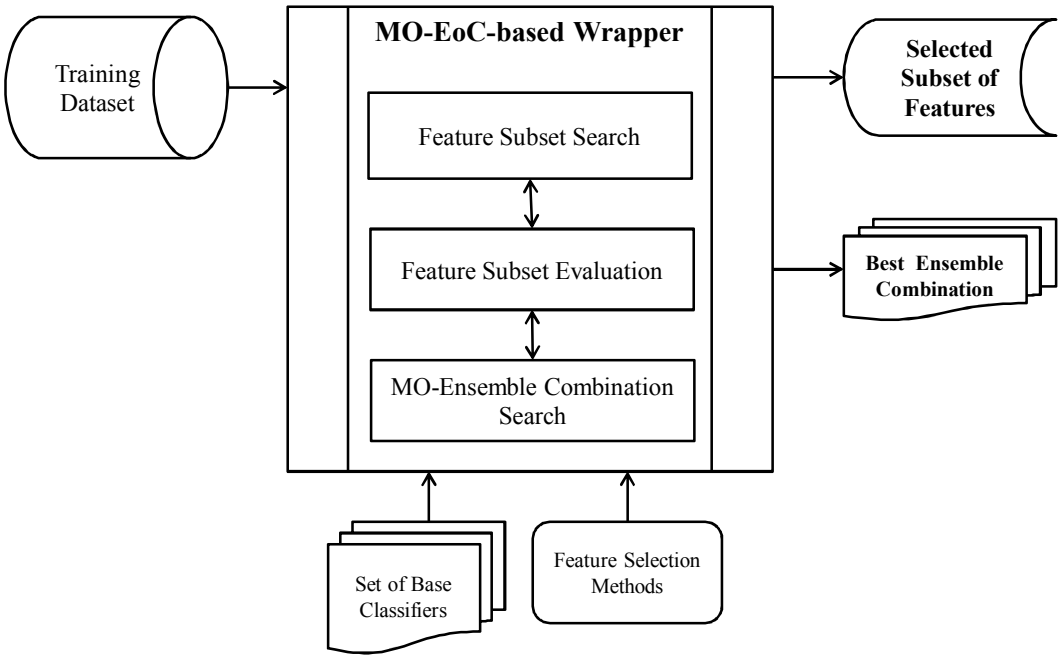
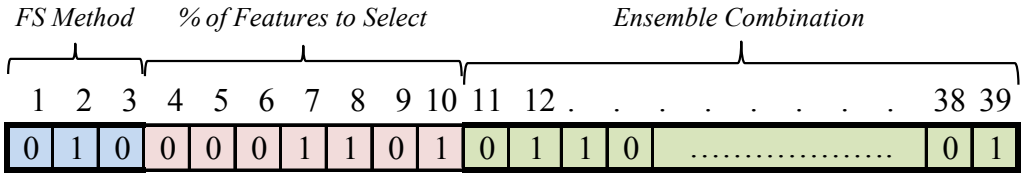


FIGURE 5.9: The architecture of the proposed *MO-EoC-WFS* algorithm for feature selection and finding the best ensemble combinations.

The next seven bits represent the binary-coded-decimal percentage of features to evaluate in the wrapper. The rest of the 29 bits represent the ensemble combination, where each bit position represents a particular classification algorithm. The selection of a particular classifier depends on the value of the corresponding bit in the individual.

Objectives used in the MO-EoC-WFS: The core idea of MO-EoC-WFS is to replace the *induction algorithm* of the generic WFS with a modified version of the MO-EoC (proposed in Chapter 5). In addition, it will use six rank-based filter FS methods (Table 5.9) as the component of a feature subset search in optimisation of the ensemble combination. We use the best pair of objectives (Obj_{mcc}, Obj_{div}) found from our previous computational experiments of MO-EoC. A detailed description of these objective functions is given in Section 5.2.4.

Individual Evaluation: Algorithm 13 provides a pseudocode of the individual evaluation for optimising multiple objectives. Initially, the individual representation string is split into an FS method string (Str_f), percentage of feature string (Str_p) and



Id	FS Method
1	CM1
2	CM2
3	IG
4	GR
5	CR
6	SU

% Feature	
Bin	1101
Dec	13

Idx	Ci	Base Classifier
11	1	BayesNet
12	2	NaiveBayes
13	3	SMO
14	4	Spegasos
.	.	
.	.	
.	.	
38	28	DecisionTable
39	29	VFI

Figure 5.10: Representation of an individual in the *MO-EoC-WFS* for wrapper feature selection and finding the best ensemble combination.

majority-vote ensemble combination string ($Str_{\mathbb{E}}$) (lines $1-3$). Then, corresponding strings are decoded to obtain the FS method index (id) and percentage value (p). After that, we obtain the subset of data from the training dataset ($\mathbb{T}$). The data structure $\mathbb{T}_F$ contains the score of each feature using all of the FS methods. The subset of features is selected using this information by selecting the top percentage of features of a particular FS method (line 7). Then, the majority-voting EoC (a variation of GA-EoC proposed in Section 3.2 with 29 base classifiers) is evaluated for MCC and Diversity score mapped by the ensemble combination string ($Str_{\mathbb{E}}$). Finally, the MCC and diversity score are saved in the individual and the evaluated individual returned.

To keep the running time to a reasonable limit, we have chosen a $60-40$ split validation of training data for evaluation of the objective functions, instead of the previously used 10-fold CV (GA-EoC in Chapter 3). To evaluate an individual with 10-fold CV, 10 models for each classifier need to be built and tested for calculating the objectives score. This CV process will increase the running time of the algorithm 10 times and will make it impractical for use in real-world datasets.

Solution Selection from the Pareto Set: In MOO, we usually obtain a set of solutions called a Pareto set. In MOO, we finally have to select a Pareto frontier that

Algorithm 13: Pseudocode of INDIVIDUALEVALUATION algorithm.

Input: Individual $\mathbb{I}$, Dataset $\mathbb{T}$, Ranked Features $\mathbb{T}_F$
Output: Evaluated Individual $\mathbb{I}$

```
//Split Individual String into different parts
Str_f ← SplitIndividual(I, 1, BitFSLen)
Str_p ← SplitIndividual(I, BitFSLen, BitPctLen)
Str_E ← SplitIndividual(I, BitPctLen, BitEnsLen)
//Decode String to Percentage, Features Id and FS Method
p ← BinaryToDecimal(Str_p)
id ← BinaryToDecimal(Str_f)
f ← F.GetFSMethod(id)
//Get subset of data T_fp with percentage of top features for the FS
  method
T_fp ← GetDataSubset( T_F, f, p)
//Evaluate majority-voting ensemble using 60 − 40 split of T_fp
E_mv ← EvaluateMajoVoteEnsemble(Str_E, T_fp)
//Set values of objective functions
I.SetObjMCC ← E_mv.GetMCC()
I.SetObjDiv ← E_mv.GetDiv()
return I
```

satisfies multiple objectives from the Pareto set. We thus need to choose a sufficiently representative solution that trade-offs our objectives from a set of non-dominating solutions to make the final classification decision. The process of choosing a solution turned into a difficult task for a large Pareto set. There are different approaches applied for solution selection in MOO, such as a priori, a posteriori, interactive and ranking-based selections. In this work, we used a weighted ranking-based approach for solution selection.

The major portion of the MO-EoC-WFS algorithm is similar to the MO-EoC algorithm for optimisation of objective pair (Obj_{mcc}, Obj_{div}). The parameters of the MO-EoC-WFS is listed in Table 5.13. The NSGA-II algorithm optimises the individuals of the population with size of 100. The individual evaluation process drives the core structure of the induction algorithm in the WFS method. The outcome of the algorithm is presented with the best ensemble combination and feature subset.

Parameter	Value
Individual Type	binary string
Individual length	39
Population Size	100
Maximum Evaluation	10000
Recombination Strategy	HUX
Recombination Rate (R_χ)	0.75
Mutation Strategy	BF
Mutation Rate (R_μ)	0.10
Objective Function	(Obj_{mcc}, Obj_{div})
Solution Selection	weighted ranking $(60\% - 40\%)$

Table 5.13: Parameter settings of the proposed MO-EoC-WFS.

5.3.4 Runtime Complexity Analysis of the MO-EoC-WFS

Now we will estimate the runtime behaviour of the proposed genetic algorithm for optimise the ensemble of classifier combination. The runtime complexity of NSGA-II is

$$\mathcal{O}(GMN^2), \tag{5.7}$$

where G is the number of generations, M is the number of objectives, and N is the population size [Jensen, 2003]. To estimate the runtime complexity of the MO-EoC-WFS, the evaluation of fitness for each of the individuals in the population will be considered with the NSGA-II.

At the initialisation operation, we generate the rank of features in the dataset for each of the participating FS methods. The runtime complexity of the FS rank initialisation is estimated by

$$\mathcal{O}(FSIni) = \mathcal{O}(m * f * FS), \tag{5.8}$$

where m is the number of feature in the dataset, f is the number of FS methods and FS denotes the estimation of runtime for a FS methods.

Line 1-6 in the Algorithm 13 has constant running time which can be ignored. The runtime required for getting the subset (percentage of top features) of data using the encoded FS method called in Line 7 is given by

$$\mathcal{O}(FSub) = \mathcal{O}(m * p), \tag{5.9}$$

where p denotes the percentage of top features to be selected from m number of features. The features could be fetched from already ranked features by the $m * p$ amount of time.

We can remove the constant valued running time from the approximation.

Now, we calculate the runtime estimation of the fitness evaluation in Line 8 of the algorithm. The base classifiers model for the ensemble are build and evaluated in runtime for a genetic individual. The dataset is separated using 60-40 split into training and validation for calculating the objective values. Let us assume, for each of the k base classifiers we require $\mathcal{O}(\mathbb{C}(D_{m,n}))$ runtime for model building and evaluation for training dataset $D_{m,n}$ with m features and n samples. The estimation of runtime requirement to evaluate an individual for each of the objectives is given by

$$\mathcal{O}(Obj) = \mathcal{O}(k * \mathbb{C}(D_{m,n})). \tag{5.10}$$

In our algorithm, the runtime estimation for evaluation of one generation consisting of MN^2 individuals is given by

$$\begin{aligned} \mathcal{O}(GenEval) &= \mathcal{O}(M * N^2) * (\mathcal{O}(Obj)) && (5.11)\\ &= \mathcal{O}(M * N^2) * \mathcal{O}(k * \mathbb{C}(D_{m,n})) && (5.12)\\ &= \mathcal{O}(M * N^2 * k * \mathbb{C}(D_{m,n})), && (5.13) \end{aligned}$$

where, N denotes the population size.

The total runtime estimation for worst case scenario for MO-EoC-WFS for G number of generations is estimated from the Equation 5.7 by

$$\begin{aligned} \mathcal{O}(MO\text{-}EoC\text{-}WFS) &= \mathcal{O}(FSIni) + \mathcal{O}(G * \mathcal{O}(GenEval)) && (5.14)\\ &= \mathcal{O}(m * f * FS) + \mathcal{O}\left(G * M * N^2 * k * \mathbb{C}(D_{m,n})\right) && (5.15) \end{aligned}$$

The constant runtime can be removed from the estimation. To get the upper bound of running time, we could ignore lower order of running times from the equation. The final equation can be formed as

$$\begin{aligned} \mathcal{O}(MO\text{-}EoC\text{-}WFS) &= \mathcal{O}(FS) + \mathcal{O}\left(G * M * N^2 * \mathbb{C}(D_{m,n})\right) && (5.16)\\ &= \mathcal{O}\left(G * M * N^2 * \mathbb{C}(D_{m,n})\right). && (5.17) \end{aligned}$$

The asymptotic growth of the runtime estimation for MO-EoC-WFS algorithm is given by Equation 5.17.

5.3.5 Computational Experiments

Datasets

We have taken eight binary-class datasets from different domains (see Table 5.14) for validating the robustness of the MO-EoC-Wrapper as FS and heterogeneous ensemble selection algorithm (MO-EoC-WFS). Some datasets have only training data and some have both training and testing data for the classifier. We have performed 10-fold CV on the datasets with only training data to measure the performance of MO-EoC-Wrapper. For the datasets with both training and testing data, we train the MO-EoC-Wrapper only using training samples and measure the generalisation performances using the testing samples. We use four datasets (namely breast cancer, leukemia, promoters and prostate cancer) from the UCI-ML Repository [Lichman, 2013] and another four (namely arcene, dexter, dorothea and madelon) from the NIPS2003 FS Challenge [Guyon et al., 2004]. Characteristics of datasets, including the source, features and sample distributions are shown in Table 5.14.

Dataset	Domain	#Features	#Train Samps.	#Test Samps.
dorothea	Drug Discovery	100000	805 (714:91)	345 (324:21)
dexter	Text Classification	20000	420 (205:215)	180 (95:85)
arcene	Mass Spectrometry	10000	100 (56:44)	100 (56:44)
Leukemia	Gene expression	7130	72 (47:25)	10-fold CV
Prostate	Gene expression	2135	102 (50:52)	10-fold CV
madelon	Artificial	500	1820 (912:908)	780 (388:392)
Promoters	Molecular Biology	57	106 (53:53)	10-fold CV
Breast	Image Feature	30	569 (212:357)	10-fold CV

Table 5.14: Characteristics of binary class datasets used for the experiment of MO-EoC-WFS method taken from UCI-ML Repository and NIPS 2003 Feature Selection Challenge, in order of their feature count.

Results & Discussion

We will report the computational results in three steps. To choose the solution from the Pareto front, we took 60% weight of MCC score and 40% of diversity score to rank Pareto-optimal solutions. The solution with the top weight will be chosen. In the beginning, we will report the classification performances of base classifiers achieved for all datasets used in the experiment. Then, we will report the generalisation performance of MO-EoC for all datasets. Finally, we will compare our experimental outcomes with the state-of-the-art EoCs. Datasets taken from UCI-ML Repository were used in the DragonWFS (a parallel GA-based wrapper FS tool) [Soufan et al., 2015]. This allows us directly to compare our

results with the state-of-the-art approach.

Classification Performances of Base Classifiers: We show the classification performances achieved by base classifiers for binary-class data classification in Table 5.15 and Table 5.16 for MCC and accuracy scores, respectively. We also have shown the number of times a base classifier has appeared as best (#B) and worst (#W). We apply entropy filtering to discretise the training dataset and discard features using the MDL principle; which has been used in many studies on biological datasets as the preprocessing step by Moscato et al. [Berretta et al., 2007b, Cotta et al., 2005, Ravetti and Moscato, 2008, Haque et al., 2016a]. After applying entropy filtering, the madelon and Promoters datasets do not retain enough features for training classifiers. So, we used their original feature set for further experiments. However, the numbers of features retained for other datasets are 893, 27, 87, 464, 1012 and 470 for arcene, Breast, dexter, dorothea, Leukemia and Prostate datasets, respectively.

While considering the accuracy scores in Table 5.16, we did not find even one classifier that performed the best in at least half of the datasets. Both of `NaiveBayes` and `SMO` appeared twice as the best classifiers among all base classifiers for classification of datasets. Conversely, `DecisionStump` appeared twice as the worst base classifier in terms of accuracy scores achieved for all binary-class data classifications, which is the highest frequency for a single base classifier appearing as the worst. In the case of MCC scores in Table 5.16, we found the same situation regarding a single classifier performing best for at least half of the datasets. `NaiveBayes` and `HoeffdingTree` appeared twice as the best classifiers for all datasets, which is the highest frequency for a single classifier performing as best classification. Conversely, both `DecisionStump` and `VFI` have shown the worst performances by achieving the least MCC scores for two binary-class data classifications. From these results, once again we observed that there does not exist a single classifier that performed best in every cases.

Training vs Testing Performances: One of the major goals of the proposed MO-EoC-WFS is to select a subset of features that best describe the dataset and that helps the optimised ensemble combination to achieve better generalisation performances. The classifier is trained on training data and tested for the performance on unknown testing data for measuring the generalisation capability. A well trained classifier will be able to perform similarly in testing data as did on training. To test the generalisation capability, we now compare the training MCC score of the MO-EoC-WFS with corresponding generalisation performances (MCC score on the testing

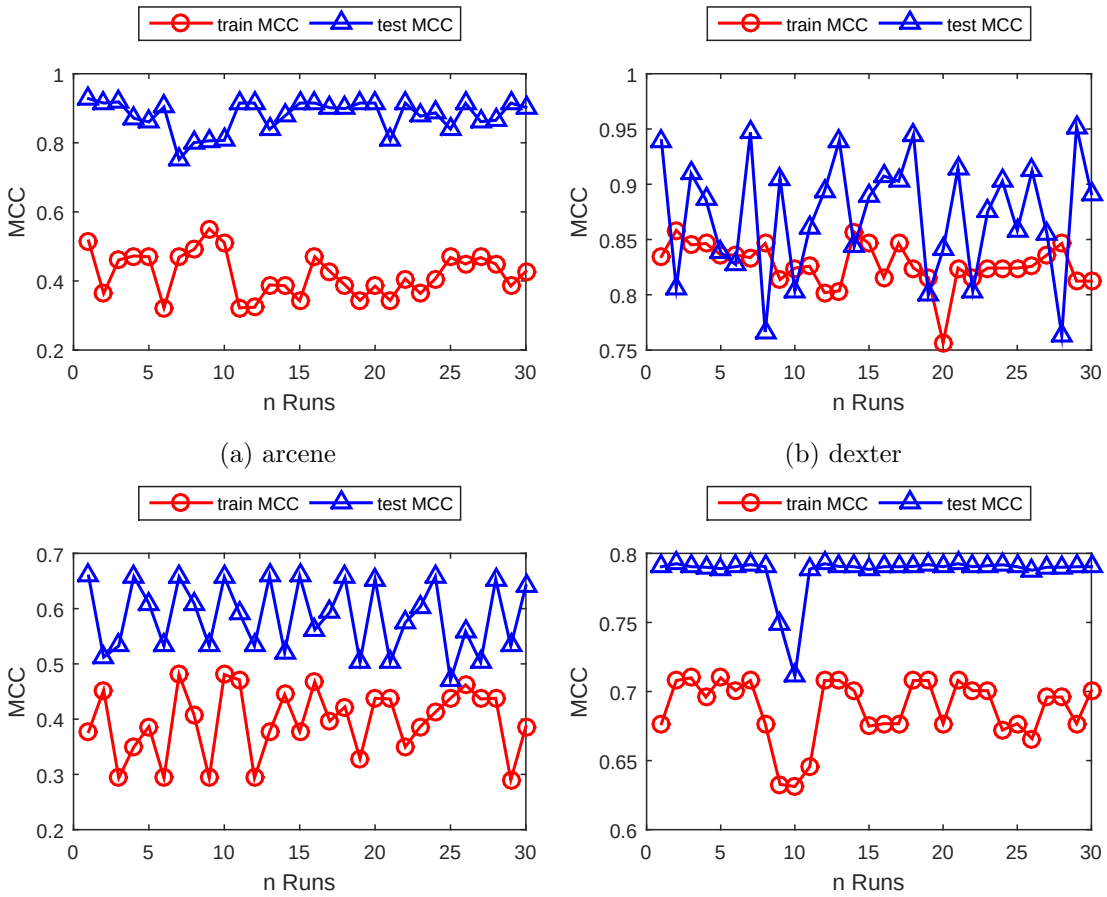
Base Classifier	arcene	Breast	dexter	dorothea	Leukemia	madelon	Promoters	Prostate	#B	#W
BayesNet	0.448	0.895	0.812	0.467	**0.970**	0.244	0.647	0.706	1	0
NaiveBayes	0.431	0.853	0.768	0.502	**0.970**	0.267	**0.796**	0.786	2	0
SMO	0.470	0.951	0.844	0.467	**0.970**	0.116	0.547	0.784	1	0
SPegasos	0.494	0.932	0.811	0.360	**0.970**	0.108	0.624	0.824	1	0
Logistic	0.203	0.881	0.725	0.221	0.829	0.077	0.661	0.747	0	0
SimpleLogistic	0.321	0.936	**0.845**	0.418	0.816	0.267	0.604	0.824	1	0
SGD	0.535	**0.955**	0.824	0.429	0.939	0.095	0.623	0.824	1	0
MLPClassifier	0.454	0.925	0.824	0.468	0.909	0.091	0.623	**0.844**	1	0
RBFNetwork	0.501	0.883	0.771	0.263	0.939	0.377	0.761	0.706	0	0
VotedPerceptron	0.437	0.792	0.778	**0.526**	0.846	0.206	0.557	0.726	1	0
ADTree	0.324	0.872	0.758	0.317	0.849	0.358	0.679	0.726	0	0
BFTree	0.344	0.849	0.711	0.387	0.722	0.519	0.453	0.611	0	0
HoeffdingTree	0.448	0.857	0.769	**0.526**	**0.970**	0.272	0.761	0.786	2	0
J48	0.363	0.851	0.737	0.368	*0.606*	0.359	0.509	0.648	0	1
LADTree	0.155	0.921	0.716	0.278	0.877	0.484	0.623	0.667	0	0
REPTree	0.384	0.834	0.773	0.417	0.626	0.541	0.472	0.628	0	0
PART	0.363	0.858	0.767	0.501	*0.606*	0.226	0.623	0.707	0	1
DecisionStump	0.392	*0.761*	*0.469*	0.481	0.694	0.272	0.437	0.578	0	2
ExtraTree	0.354	0.846	0.756	0.396	0.640	0.128	0.415	*0.530*	0	1
FT	0.299	0.951	0.800	0.403	0.816	0.121	0.547	0.804	0	0
LMT	0.321	0.936	**0.845**	0.418	0.816	0.480	0.643	0.824	1	0
RandomTree	0.501	0.884	0.722	0.312	0.666	0.208	*0.305*	0.686	0	1
SimpleCart	0.344	0.853	0.711	0.467	0.694	0.575	0.453	0.609	0	0
IBk	**0.578**	0.898	0.722	0.332	0.908	0.091	0.437	0.807	1	0
ConjunctiveRule	0.392	0.788	*0.469*	0.481	0.816	0.191	0.511	0.608	0	1
JRip	0.192	0.864	0.713	0.479	0.751	**0.593**	0.566	0.667	1	0
Ridor	0.321	0.845	0.669	0.439	0.821	0.388	0.510	0.667	0	0
DecisionTable	*0.008*	0.872	0.665	0.362	0.632	0.509	0.548	0.706	0	1
VFI	0.215	0.823	0.727	*0.212*	0.734	*-0.031*	0.452	0.696	0	2
Best	0.578	0.955	0.845	0.526	0.970	0.593	0.796	0.844		
Worst	0.008	0.761	0.469	0.212	0.606	-0.031	0.305	0.530		

Table 5.15: Base classifiers' MCC scores for all binary-class datasets used in the experiments of MO-EoC-WFS, including the number of times it performed as best (#B) and worst (#W).

Base Classifier	arcene	Breast	dexter	dorothea	Leukemia	madelon	Promoters	Prostate	#B	#W
BayesNet	72.00	95.08	90.56	93.62	**98.61**	62.18	82.08	85.29	1	0
NaiveBayes	71.00	93.15	88.33	93.91	**98.61**	63.33	**89.62**	89.22	2	0
SMO	74.00	97.72	**92.22**	93.04	**98.61**	55.77	77.36	89.22	2	0
SPegasos	75.00	96.84	90.56	91.30	**98.61**	55.38	81.13	91.18	1	0
Logistic	61.00	94.38	86.11	85.80	91.67	53.85	83.02	87.25	0	0
SimpleLogistic	67.00	97.01	**92.22**	93.04	91.67	63.33	80.19	91.18	1	0
SGD	77.00	**97.89**	91.11	91.88	97.22	54.74	81.13	91.18	1	0
MLPClassifier	73.00	96.49	91.11	91.59	95.83	54.49	81.13	**92.16**	1	0
RBFNetwork	75.00	94.55	88.33	*81.16*	97.22	68.85	87.74	85.29	0	1
VotedPerceptron	71.00	90.33	88.89	92.75	93.06	60.26	77.36	86.27	0	0
ADTree	67.00	94.02	87.78	93.91	93.06	67.82	83.96	86.27	0	0
BFTree	68.00	92.97	85.56	92.17	87.50	75.51	72.64	80.39	0	0
HoeffdingTree	72.00	93.32	88.33	92.75	98.61	63.59	87.74	89.22	0	0
J48	69.00	92.97	86.67	94.20	*81.94*	67.95	75.47	82.35	0	1
LADTree	59.00	96.31	85.56	92.46	94.44	73.72	81.13	83.33	0	0
REPTree	70.00	92.27	88.33	93.62	*81.94*	77.05	73.58	81.37	0	1
PART	69.00	93.32	88.33	**95.07**	*81.94*	61.28	81.13	85.29	1	1
DecisionStump	66.00	*88.93*	*68.33*	94.78	86.11	63.59	71.70	78.43	0	2
ExtraTree	68.00	92.79	87.78	90.72	83.33	56.41	70.75	*76.47*	0	1
FT	66.00	97.72	90.00	91.88	91.67	56.03	77.36	90.20	0	0
LMT	67.00	97.01	**92.22**	93.04	91.67	73.72	82.08	91.18	1	0
RandomTree	75.00	94.55	86.11	91.59	84.72	60.38	*65.09*	84.31	0	1
SimpleCart	68.00	93.15	85.56	93.62	86.11	78.72	72.64	80.39	0	0
IBk	**79.00**	95.25	86.11	93.62	95.83	54.49	71.70	90.20	1	0
ConjunctiveRule	66.00	90.16	*68.33*	94.78	91.67	57.05	75.47	80.39	0	1
JRip	60.00	93.67	85.56	93.91	88.89	**79.62**	78.30	83.33	1	0
Ridor	67.00	92.79	82.78	94.49	91.67	69.10	75.47	83.33	0	0
DecisionTable	*51.00*	94.02	83.33	93.62	83.33	73.85	77.36	85.29	0	1
VFI	61.00	91.74	86.11	94.20	86.11	*48.59*	66.98	84.31	0	1
Best	79.00	97.89	92.22	95.07	98.61	79.62	89.62	92.16		
Worst	51.00	88.93	68.33	81.16	81.94	48.59	65.09	76.47		

TABLE 5.16: Base classifiers' accuracy (in %) for all binary-class datasets used in the experiments of MO-EoC-WFS, including the number of times it performed as best (#B) and worst (#W).

dataset). We plot the line graph in Figure 5.11 of training and testing MCC achieved for 30 independent runs of MO-EoC-WFS for datasets taken from the NIPS 2003 FS challenge.



(a) arcene (b) dexter

(c) dorothea (d) madelon

Figure 5.11: Line plots showing the training and testing MCC scores of Pareto-optimal solutions for optimising the objectives pair of (Obj_{mcc}, Obj_{div}) on the (a) arcene, (b) dexter, (c) dorothea and (d) madelon datasets for 30 repetitions of MO-EoC-WFS.

From the experimental outcomes of NIPS 2003 FS challenge shown in Figure 5.11, it is evident that the feature subset selected by the MO-EoC-WFS is able to describe the dataset well. We can observe that the testing performance curve is better than the training curve for arcene, dexter and dorothea datasets for the MCC scores. For the madelon dataset, the testing MCCs are lower than the training MCC scores (Sub-figure 5.11d). However, the best classification performance achieved by the best base classifier for the madelon dataset is 0.593. The testing MCC score curve of

MO-EoC-WFS is above 0.65 for most of the cases. Moreover, the worst testing MCC scores attained for the 30 repetitions of MO-EoC-WFS is 0.58, which is comparable to the performance of the best base classifier. Hence, we can claim that the MO-EoC-WFS is able to select a better feature subset for diverse categories of datasets. Further, those selected features are sufficiently capable of describing the dataset well, and that leads to a good generalisation performance.

Classification Performances of MO-EoC-WFS: Now, we will discuss the classification performances achieved by the proposed method for 30 repeated runs in detail. We will first compare the training and testing MCC scores of MO-EoC-WFS, then compare its performances with other state-of-the-art classifier ensembles.

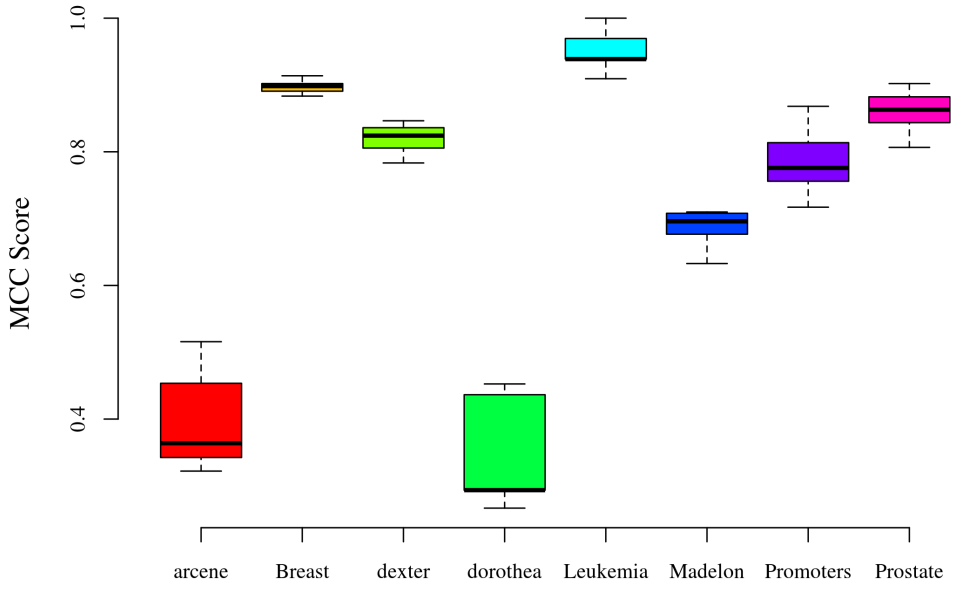
Dataset	Min	1st Qu.	Median	Mean	3rd Qu.	Max
arcene	0.322	0.343	0.364	0.388	0.454	0.516
Breast	0.827	0.891	0.898	0.896	0.902	0.921
dexter	0.783	0.806	0.824	0.821	0.836	0.846
dorothea	0.267	0.294	0.294	0.356	0.436	0.453
Leukemia	0.877	0.940	0.939	0.941	0.970	1.000
madelon	0.633	0.677	0.696	0.690	0.708	0.710
Promoters	0.717	0.756	0.776	0.786	0.814	0.868
Prostate	0.807	0.844	0.863	0.858	0.882	0.902

TABLE 5.17: Summary statistics of classification performances (in MCC scores) for 30 runs of MO-EoC-WFS for eight benchmarking datasets.

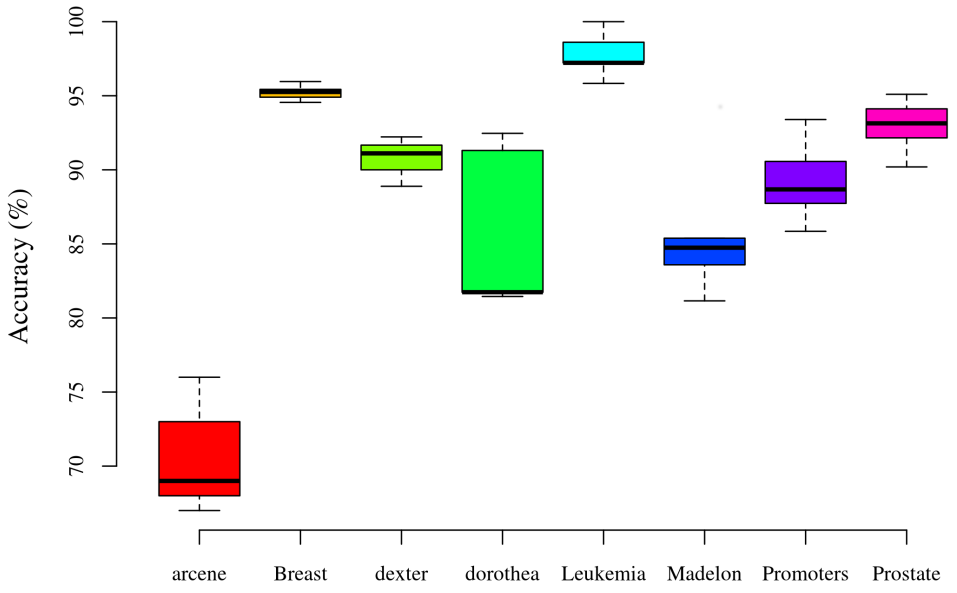
Dataset	Min	1st Qu.	Median	Mean	3rd Qu.	Max
arcene	67.00	68.00	69.00	70.07	73.00	76.00
Breast	91.92	94.90	95.25	95.14	95.43	96.31
dexter	88.89	90.00	91.11	90.90	91.67	92.22
dorothea	81.45	81.74	81.74	86.45	91.30	92.46
Leukemia	94.44	97.22	97.22	97.32	98.61	100.00
madelon	81.15	83.59	84.74	84.30	85.38	85.38
Promoters	85.85	87.74	88.68	89.20	90.57	93.40
Prostate	90.20	92.16	93.14	92.87	94.12	95.10

TABLE 5.18: Summary statistics of classification accuracies (in %) for 30 runs of MO-EoC-WFS for eight benchmarking datasets.

Table 5.17 and Table 5.18 show a summary of classification performances achieved by MO-EoC-WFS for eight datasets in the MCC and accuracy scores, respectively. The summary is computed from the results of 30 independent runs in each of the datasets.



(a) MCC



(b) Accuracy

Figure 5.12: Boxplot showing the classification performances achieved by MO-EoC-WFS for 30 runs on each of the eight benchmarking datasets in (a) MCC and (b) Accuracy measures.

From the table, we can observe that the maximum MCC scores achieved by the MO-EoC-WFS for Breast, Leukemia and Prostate are more than 0.90 in MCC score and above 90% in the accuracy measure. For the Leukemia data classification, it achieved the highest possible generalisation by perfectly classify all of the samples. The worst performance of MO-EoC-WFS is observed for the dorothea data classification, considering the MCC score. The dataset has the highest class imbalance ratio of $1:8$ and has been reported as one of the challenging datasets for classification [Guyon et al., 2007]. Recently, [Jayadeva et al., 2015] proposed an extreme learning machine approach that achieved 90% testing accuracy for the dorothea dataset classification task. The authors have not reported either the MCC score or the confusion matrix for the classification outcome. Hence, it is not possible actually to compare their performance in a more suitable measure (MCC score) for class-imbalanced data with our approach. However, considering the accuracy score, the generalisation accuracy reached a maximum of 92% by MO-EoC-WFS for the dorothea dataset, which outperforms the latest result in the literature.

Moreover, the box plots for MCC and Accuracy scores in Subfigure 5.12a and Subfigure 5.12b express the stability of performances of MO-EoC-WFS in benchmarking datasets. From these performances, it is evident that in most of the cases the heights of the boxes are small (except for the dorothea dataset), which argues for the stability in generalisation performance of the MO-EoC-WFS. Hence, we claim that MO-EoC is a steady approach for data classification and FS employing multi-objective EoC.

5.3.6 Statistical Comparison of Results

Let us determine if there is any classifiers whose performance can be regarded as significantly different with base classifiers and MO-EoC-WFS. Hence, we apply the classical Friedman test and a modification by Iman and Davenport. The Corrected Friedman's chi-squared value $Q = 79.581$ is found for the MCC scores in Table 5.15 and the best performances of MO-EoC-WFS in Table 5.17. The small $p\text{-}value = 1.334e\text{-}06$ indicates that there is one or more algorithm exist whose performance is significant for MCC scores. The Iman Davenport's correction of Friedman's rank sum test provide chi-squared value $Q = 65.675$ for the Accuracy scores in Table 5.16 and the best scores of MO-EoC-WFS in Table 5.18. The $p\text{-}value = 0.0001156$ indicates that there is one or more classifier exist whose performance is significant for accuracy scores. Hence, for both the MCC and Accuracy scores there exist at least one classifier whose performance is significantly different than others.

Classifier	p-value	
	MCC	*Accuracy*
BayesNet	**0.126**	**0.135**
NaiveBayes	**0.204**	**0.182**
SMO	**0.129**	**0.149**
SPegasos	**0.114**	**0.187**
Logistic	0.000838	0.00127
SimpleLogistic	**0.0516**	**0.0786**
SGD	**0.23**	**0.246**
MLPClassifier	**0.17**	**0.153**
RBFNetwork	**0.141**	**0.154**
VotedPerceptron	0.0177	0.00787
ADTree	0.0207	**0.0712**
BFTree	0.000584	0.000948
HoeffdingTree	**0.272**	**0.18**
J48	0.000426	0.00205
LADTree	0.00541	0.0215
REPTree	0.00233	0.00381
PART	0.00481	0.00641
DecisionStump	6.17E-05	1.73E-05
ExtraTree	3.12E-05	2.15E-05
FT	0.0083	0.0128
LMT	**0.186**	**0.266**
RandomTree	0.000222	0.000573
SimpleCart	0.0013	0.00146
IBk	0.0101	0.0307
ConjunctiveRule	0.000201	4.58E-05
JRip	0.00386	0.0033
Ridor	0.00176	0.00213
DecisionTable	0.000417	0.00129
VFI	4.08E-06	2.13E-05

Table 5.19: The *p-values* from statistical test of classification performances of base classifiers and MO-EoC-WFS for eight benchmarking datasets using post-hoc calculation of Friedman's Aligned Rank test with Iman Davenport's correction. The statistically similar base classifiers of MO-EoC-WFS are shown in bold face and statistically significant classifiers are shown in normal font face.

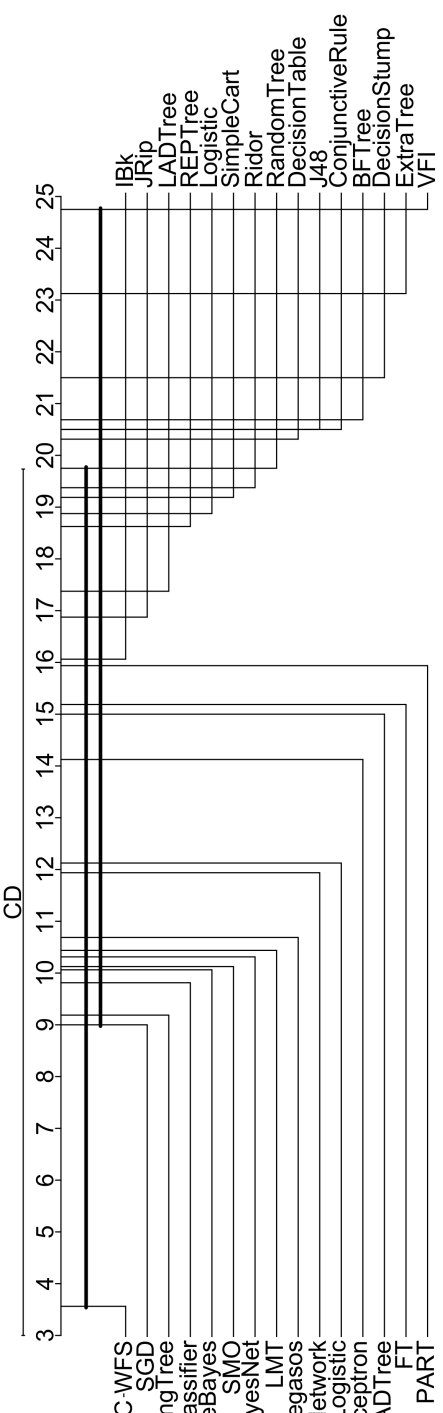


FIGURE 5.13: The critical difference (CD) plot shown the critically different base classifiers and MO-EoC-WFS over multiple datasets for MCC score. The critical distance is calculated at the significance level of 0.05 using Nemenyi test.

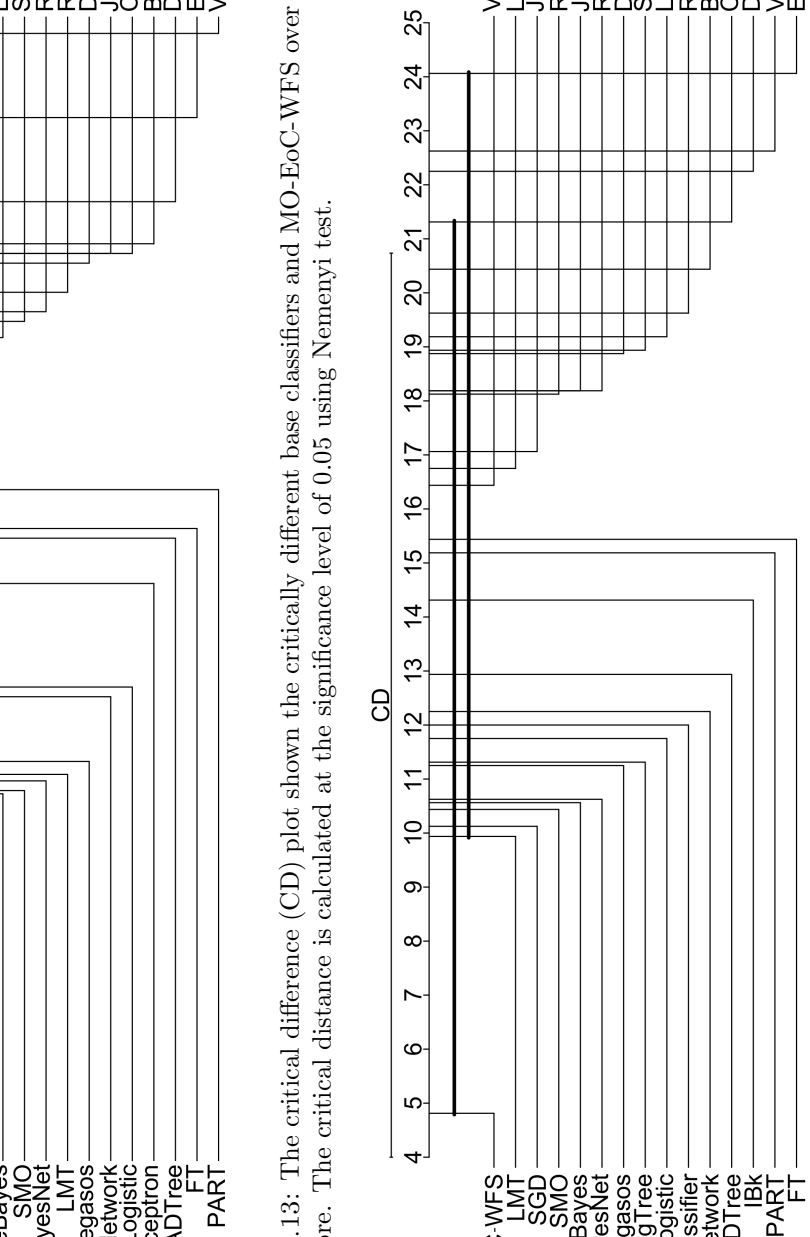


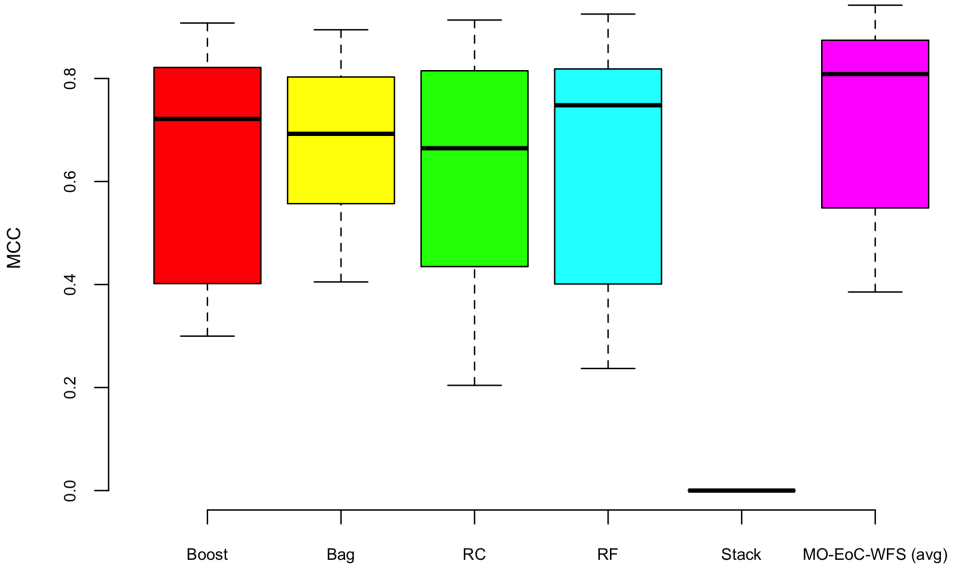
FIGURE 5.14: The critical difference (CD) plot shown the critically different base classifiers and MO-EoC-WFS over multiple datasets for Accuracy score. The critical distance is calculated at the significance level of 0.05 using Nemenyi test.

To visualise the critical difference (CD) among base classifiers and MO-EoC-WFS over a multiple problems (datasets), the plot of the CD is generated from Nemenyi test. Here we used $alpha = 0.05$ which regarded as the significance level at 95%. In Figure 5.13, the CD plot shows the average rank difference of all base classifiers and MO-EoC-WFS for MCC scores. The critical difference value is $CD = 16.735$ for MCC measures. The `DecisionTable`, `J48`, `COnjunctiveRule`, `BFTree`, `DecisionStump`, `ExtraTree` and `VFI` are in critical distance from MO-EoC-WFS for their MCC score. The CD plot for accuracy is shown in Figure 5.14. The critical difference value is $CD = 16.735$ for accuracy measures. The `ConjunctiveRule`, `DecisionStump`, `VFI` and `ExtraTree` are in critical distance from MO-EoC-WFS for their accuracy score.

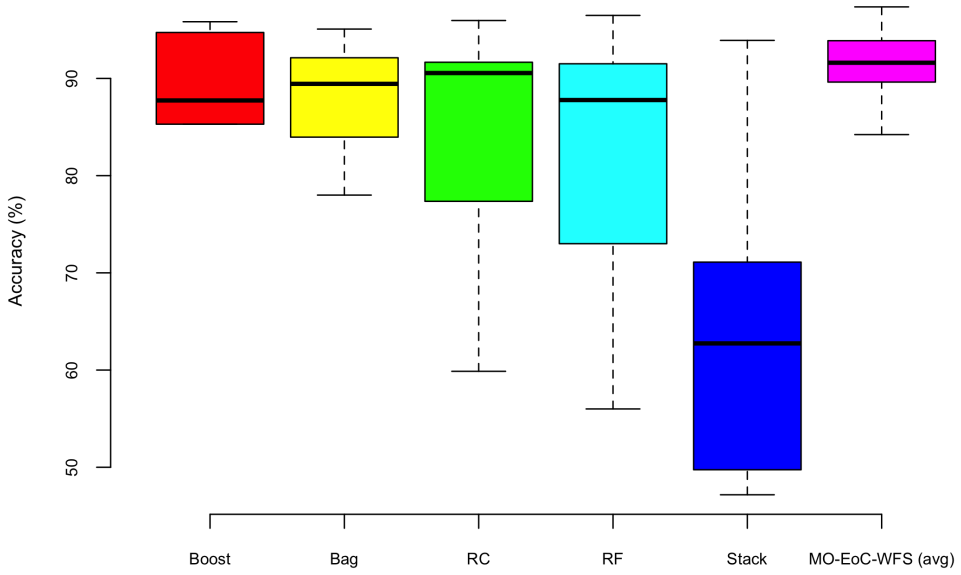
To better understand which base classifiers are statistically significant comparing the performance of MO-EoC-WFS, we conducted post-hoc calculation of *Friedman's Aligned Rank test with Iman Davenport's correction*. The *p-value* is shown for each of the base classifier and MO-EoC-WFS in the Table 5.19 for MCC and Accuracy scores. The *p-value* smaller than 0.05 in a row expressed that the performance of MO-EoC-WFS is statistically significant compared with the base classifier. Comparing the *p-value* for MCC scores, `BayesNet`, `NaiveBayes`, `SMO`, `SPegasos`, `SimpleLogistic`, `SGD`, `MLPClassifier`, `RBFNetwork`, `HoeffdingTree` and `LMT` are statistically similar to MO-EoC-WFS. The MO-EoC-WFS is statistically significantly better than Remaining 19 base classifiers for MCC scores. The `BayesNet`, `NaiveBayes`, `SMO`, `SPegasos`, `SimpleLogistic`, `SGD`, `MLPClassifier`, `RBFNetwork`, `ADTree`, `HoeffdingTree` and `LMT` exhibited the same performances of MO-EoC-WFS considering the accuracy score. Hence, the MO-EoC-WFS is statistically significantly better than remaining 18 base classifiers for accuracy score.

Form the statistical test on the results, we found that MO-EoC-WFS is significantly better approach than majority of the base classifiers (better than 19 base classifiers for MCC and 18 base classifiers for accuracy scores among 29 base classifiers).

Classification Performances of the State-of-the-Art Ensembles: We report the classification performances achieved by some state-of-the-art EoCs. We have used their default parameters and settings available in the WEKA frameworks. The classification performances could differ with optimisation of parameter values. We report the classification performances of `AdaBoostM1` (Boost), `Bagging` (Bag), `RandomCommittee` (RC), `RandomForest` (RF) and `Stacking` (Stack) for all datasets in BoxPlot of Figure 5.15.



(a) MCC



(b) Accuracy

FIGURE 5.15: Comparison of classification performances achieved by state-of-the-art ensemble of classifiers and MO-EoC-WFS (average of 30 runs) for eight benchmarking datasets for (a) MCC and (b) Accuracy measures.

Figure 5.15a shows a comparison of MCC scores achieved by four state-of-the-art EoCs and the proposed MO-EoC-WFS. The box-and-whisker plot for the average MCC scores achieved by MO-EoC-WFS for 30 runs has appeared as the top performing EoC. It has best 75^{th} percentile value and best mean score. In regard to the MCC measure, `Stacking` also performed worst among the EoCs for these dataset classifications.

Figure 5.15b shows a comparison of accuracy scores (in %) achieved by four state-of-the-art EoCs and the proposed MO-EoC-WFS. The box-and-whisker plot for the average accuracy scores achieved by MO-EoC-WFS for 30 runs has appeared as stable outcomes. The spread of the boxes is very low and their mean value is above other EoCs. `Stacking` performed as the worst among EoCs for these dataset classifications.

It is evident from the classification outcome comparison that the proposed MO-EoC-WFS outperforms the state-of-the-art EoCs for both MCC and Accuracy scores.

Feature Subset Size: We now report on the feature subset sizes selected by MO-EoC-WFS for all datasets. The feature subsets selected for 30 runs of the proposed approach on benchmarking datasets are summarised in Table 5.20 with the average, lowest and largest number of features.

Dataset	Min	1st Qu.	Median	Mean	3rd Qu.	Max	Orig	Reduce
arcene	26	35	35	70.55	62	438	893	96.08%
Breast	2	7	8	7.621	8	11	27	70.37%
dexter	36	41	43	41.52	43	43	87	50.57%
dorothea	190	190	190	195.5	195	227	464	59.05%
Leukemia	40	40	91	100.6	101	384	1012	91.01%
madelon	15	15	15	15.17	15	20	500	97.00%
Promoters	4	5	5	7.793	8	25	57	91.23%
Prostate	9	160	174	172.5	202	230	470	92.98%

Table 5.20: Summary statistics of selected Feature Subset Size by MO-EoC-WFS for eight benchmarking datasets. The corresponding reduction of feature size (Reduce) from the number of features (Orig) used in the MO-EoC-WFS is also shown for each dataset.

The summary Table 5.20 shows the minimum (min), first quartile (1st Qu.), median, mean, third quartile (3rd Qu.) and maximum (max) subset of features. It also presents the percentage reduction in feature size comparing the median feature subset size with the original number of features used in MO-EoC-WFS. For the madelon dataset, we notice the maximum reduction in feature size (97.00%) by MO-EoC-WFS. The least reduction of feature size (50.57%) is witnessed for the dexter dataset.

We can also observe that the average feature size reduced by 81.03% of the original size.

Thus, the MO-EoC-WFS performed well in reducing the feature size by applying the wrapper FS approach driven by GA search and EoC as the induction algorithm. However, if we use the method again on selected best feature subset for another level of optimisation, that could produce better feature subsets and perhaps this could lead to better generalisation performances.

5.4 Case Study: Multiclass Data Classification and Feature Selection in Breast Cancer

The datasets we have used so far for evaluating the performances of MO-EoC-WFS are binary classes. We will now apply the MO-EoC-WFS on multiclass datasets. We have to remove the `SPegasos`, `SGD`, `VotedPerceptron` and `ADTree` from the pool of base classifiers of multiclass variants of MO-EoC-WFS. Those classifiers cannot handle multiclass datasets. Therefore, the length of individual for the multiclass version of MO-EoC-WFS was reduced to 35, including 25 base classifiers. Other implementations and settings of the program remained unchanged.

To evaluate the performance of the multiclass MO-EoC-WFS for feature subset selection and data classification, we used a large breast cancer dataset from the METABRIC group that contains the gene expression of 2000 breast tumours with 48803 probes [Curtis et al., 2012]. METABRIC divides the dataset into discover and validation sets containing 997 and 989 samples, respectively. The samples were labelled using the original PAM50 method [Parker et al., 2009]. It divides the dataset samples into five categories namely, basal-like (Basal), HER2-enriched (Her2), luminal A (LumA), luminal B (LumB) and normal-like (Normal). Hence, the METABRIC dataset classification is a five-class problem.

5.4.1 Preprocessing of the METABRIC Dataset

We considered the original breast cancer dataset consisting of 2000 samples and 48803 probes with the PAM50 labelling of subtypes. First, we applied the entropy filtering described in Section 3.3.3. The entropy filtering leaves 15328 probes from the original 48803 probes. We will execute the multiclass MO-EoC-WFS on this subset of the dataset. The time required for model building and validation is very high for this dataset. Hence,

we limit to a single execution of MO-EoC-WFS for this multiclass dataset, instead of 30 repetitions used in other cases.

5.4.2 Training Performances

The training dataset (namely Discovery, consisting of 997 samples) is provided to the MO-EoC-WFS to find the best feature subset using the MOO approach. It considers the maximisation of both MCC and diversity scores. After executing the MO-EoC-WFS on the Discovery dataset, it returned with the eight Pareto-optimal solutions shown in Table 5.21. The average MCC score of the non-dominating solutions is 0.851 on the Discovery dataset.

Sol Id	MO-EoC-WFS Individual	MCC	Diversity	Rank Scr
1	00100001001001001001011100100101010	0.829	1.00	0.897
2	00000011001101101011110101011101000	0.873	0.00	0.524
3	00000001011101100001110111000101010	0.838	0.63	0.753
4	00100100011001000001100111100111001	0.852	0.40	0.671
5	00000010111101000100100110001101000	0.864	0.13	0.568
6	00100111010000000011110111000100100	0.848	0.50	0.709
7	00000011001101110001111100001101000	0.862	0.25	0.618
8	00000001100000101111100111001101100	0.844	0.60	0.746

Table 5.21: The Pareto-optimal solutions for the MO-EoC-WFS on the Discovery set of the METABRIC breast cancer dataset. The performance shows the Pareto-optimal solution with string representation, values of two objective functions and weighted score of objectives for solution selection.

Figure 5.16 shows the Pareto front for the Discovery dataset. Here, we found that the Pareto-optimal solution with lowest diversity (entropy score of 0) has the maximum MCC score of 0.873. Conversely, the solution with the utmost diversity score, 1.0, achieved the lowest MCC score of 0.829 on the training dataset. We find a linear descending relationship between the diversity and MCC score in MOO. The same phenomenon has been observed for the optimisation of (Obj_{mcc}, Obj_{div}) in Section 5.2.6. As before, increasing the diversity will not help the ensemble to learn well.

Figure 5.17 shows the types of base classifiers selected by the MO-EoC-WFS for the METABRIC Discovery dataset. From the figure, we find that `DecisionStump` and `IBk` have been selected for every solution in the Pareto-optimal solution set. The `SMO` classifiers have not been selected by any combinations of the Pareto-optimal solutions.

Now we consider the base classifiers by their types. We marked a type with light green if at least one base classifier from that category appeared in all of the Pareto-optimal

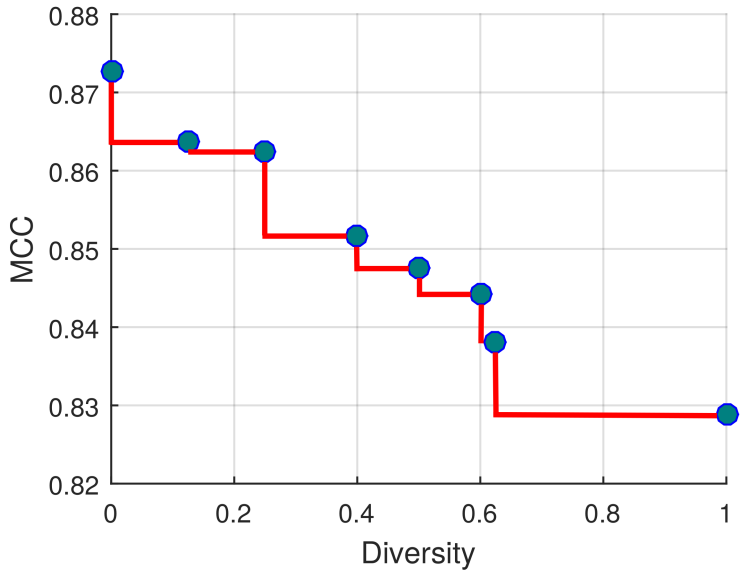


FIGURE 5.16: Plot of the Pareto-optimal solutions for the MO-EoC-WFS on the Discovery set of the METABRIC breast cancer dataset.

solutions. At least one base classifier from the Bayes, Decision Trees, Trees and Rule Learner categories have been selected while considering all of the eight Pareto-optimal solutions. The red colour is used for denoting the types of base classifier from which not a single classifier has been selected while considering all Pareto-optimal solutions. Only the SVM falls in this group. The black colour represents the group where a base classifier was selected by at least one solution, but not by all solutions. The Linear, Neural Network, Decision Table and Feature Interval types show this characteristic. The solid green coloured group of base classifiers denotes that each of the base classifiers have been selected by all of the Pareto-optimal solutions. Decision Stumps and k-NN are the two categories of base classifiers that appeared in all solutions.

5.4.3 Validation Performances

The best solution selected from the Pareto-optimal solutions consisted of 444 probes chosen by the CM2 FS method. We evaluated the classification performances of the ensemble created with the selected base classifiers combination on the validation dataset. We report the confusion matrix for the validation dataset in Table 5.22. The rows represent the original labels and columns represent the labels by the classification. Here, we can observe that many samples from other classes have been classified as LumA. 95 Normal samples,

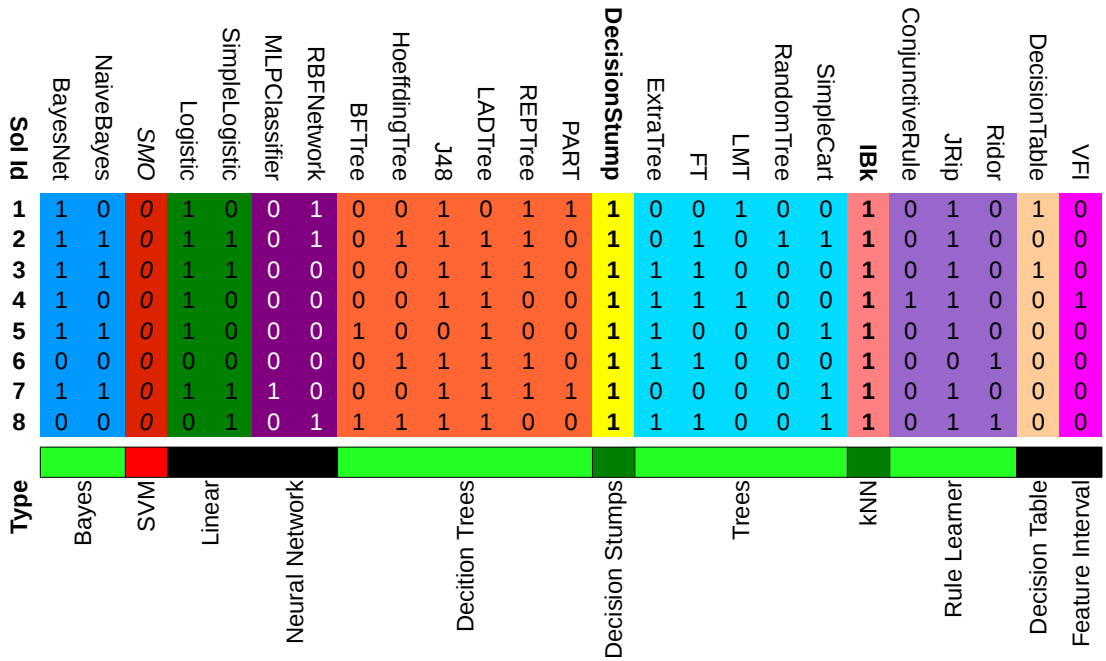


Figure 5.17: Patterns of base classifiers selection in the Pareto-optimal solutions for the METABRIC dataset. The light green coloured group denotes at least one base classifier appearing in all solutions. The red coloured group denotes that not a single base classifier from this group has been selected for a single solution. The solid green denotes a type of base classifier from where each of them have appeared in all solutions. The light green denotes that at least one base classifier has been selected for all solutions.

81 LumB samples, 32 Her2 samples and 12 Basal samples have been labelled as LumA. This class attracts the highest number of misclassified samples. A total of 44 Her2 samples have been classified as LumB in the validation dataset. The training performance of the ensemble combination on the Discovery dataset was quite good. But that performance does not reflect on the validation data for the samples labelled with the PAM50 method.

The inconsistency with the distribution of discovery and validation labels was also pointed out by the team of Prof. Pablo Moscato in [Milioli et al., 2015]. From the comparative performance analysis of the dataset with 24 classifiers using the PAM50 gene and CM1 score-based biomarker (consisting of 30 genes), they revealed better subtyping performances with new biomarkers. Later, based on the iterative analysis of subtype prediction performances, they relabelled the original METABRIC labels [Milioli et al., 2016]. The new labelling provided a more accurate prediction outcome and better agreement on patients' overall survival from the original dataset.

If we examine the validation performances of each of the Pareto-optimal solutions

	Basal	Her2	LumA	LumB	Normal
Basal	152	28	12	5	16
Her2	3	74	32	44	0
LumA	0	0	253	2	0
LumB	0	0	81	143	0
Normal	0	1	95	1	47

TABLE 5.22: Confusion matrix showing the validation output of the best ensemble selected by MO-EoC-WFS using the PAM50 subtyping labels.

Sol Id	FSName	#Feat	MCC	Acc	Prec	F-Meas	AUC	SEN	SPEC
1	CM2	444	0.612	67.64	0.735	0.667	0.787	0.676	0.898
2	CM1	1280	0.636	69.77	0.748	0.689	0.802	0.698	0.907
3	CM1	539	0.620	68.35	0.741	0.674	0.792	0.684	0.901
4	CM2	1662	0.617	68.15	0.739	0.671	0.790	0.681	0.899
5	CM1	1177	0.600	66.73	0.730	0.654	0.781	0.667	0.895
6	CM2	2718	0.613	67.54	0.744	0.664	0.786	0.675	0.897
7	CM1	1280	0.616	67.95	0.746	0.666	0.789	0.679	0.898
8	CM1	642	0.628	69.16	0.742	0.683	0.798	0.692	0.904

TABLE 5.23: Validation performances of all solutions from the Pareto front for the MO-EoC-WFS on the METABRIC breast cancer dataset with PAM50 subtype labelling.

in Table 5.23, we observe a correlation between their training and testing performance achievement. The training MCC scores vary between 0.83 and 0.87 and the testing MCC scores achieved in the range 0.60 to 0.64. Among the eight solutions, the CM1 [Marsden et al., 2013] and CM2 [Filiou et al., 2014] have appeared five and three times, respectively, from the considered six FS methods. This supports the novelty of the Craig–Moscato feature-ranking methods for feature subset selection from biological datasets. It is noticeable from the best-scored non-dominated solutions that the MO-EoC-WFS reduces the feature dimension of the METABRIC Discovery dataset (after entropy filtering) from 15328 to only 444 (which is a 97% reduction). The average dimensionality reduction rate considering the eight Pareto-optimal solutions is 92%. It is apparent that MO-EoC-WFS also performed well as a dimensionality reduction approach.

5.5 Summary

We proposed and evaluated a MOO of EoC and wrapper FS, namely MO-EoC-WFS. It uses NSGA-II, a non-dominated sorting GA, as the underlying MOO algorithm. It optimises the objective pair of (Obj_{mcc}, Obj_{div}). The individuals of the GA encode values

representing the selection of FS method, the subset of feature information for the wrapper approach and the choice of base classifiers for the EoC. The outcome of the algorithm has been evaluated on eight benchmarking datasets. Experimental results confirm that the proposed MO-EoC-WFS simultaneously serves the role of wrapper FS method and ensemble combination search approach. The selected subset of features reduces the dimensionality of the dataset by 81.03%. In the case of classification performances, the MO-EoC-WFS provides a consistent performance and outperforms other state-of-the-art EoCs.

The proposed method uses the different number of base classifiers to facilitate the multiclass data classification. The performance is assessed with a large real-world biological dataset, the METABRIC breast tumour dataset. The feature reduction performance of the MO-EoC-WFS is remarkable for the dataset with an average reduction ratio of 92%. The classification performance reveals the inconsistency with the class labelling of the dataset and agrees with the discovery of Moscato et al. [Milioli et al., 2015, Milioli et al., 2016]. Using Moscato et al.'s proposed labelling of the METABRIC dataset [Milioli et al., 2016] as the training dataset of MO-EoC-WFS and iteratively apply the MO-EoC-WFS multiple times could find more interesting results and contribute to new biomarker discovery. Applying the MO-EoC-WFS at a multilevel will reduce the dimensionality of the dataset dramatically and possibly come up with biomarkers with a small number of genes. Further investigation of this approach on the METABRIC dataset is time consuming and not suited to the allocated time of the thesis. However, this could be an interesting approach to be investigated in future.

This page intentionally left blank.

6

Conclusion and Final Remarks

We have investigated the ensemble combination search using GAs for biological datasets. Although the proposed methods can be utilised in any domain, we developed them for biological datasets. The source code of the projects is open-source and replacing the base classifier pool with domain-appropriate classifiers will be sufficient to work on other domains. One of the primary goals of machine learning and pattern classification is to enhance the generalisation performance of classification algorithms. The EoC is a popular way to improve the classification performance for biological datasets using GAs. We contributed to the selection of base classifiers from a large heterogeneous classifier pool to build the ensembles. The selection of base classifiers was driven through the GA to optimise the combination, instead of creating the ensemble with all base classifiers.

To develop the algorithm, we analysed both weighted and non-weighted majority schemes for ensemble decision fusion methods. Our experimental outcome revealed that the non-weighted-majority voting created with the MCC score performed better than weights optimised with MCC scores. The weight optimisation with base classifier's MCC score is unrelated to the generalisation performance of the weighted-vote EoC. The individual MCC score optimisation is suitable for use in the ensemble combination selection in non-weighted-majority voting. We have compared different approaches to the ensemble combination search method using both benchmarking and real-world biological datasets

to determine a better strategy.

Further, we extend our research from single-objective optimisation to MOO for the ensembles combination search. We evaluated the performance of MCC, diversity and ensemble size as objective functions and revealed that MCC and diversity are two conflicting objectives for finding the best ensemble combination. This approach has further extended to the wrapper-based FS method in addition to searching the ensembles' combination. Both approaches have been validated with the performances on benchmarking and real-world datasets.

We will discuss our contributions and future plans for this research in this chapter.

6.1 Research Contributions

The research has contributed to the search for base classifier combinations to create the ensembles. The usual practice is to search for an ensemble combination using some type of greedy approach. They add base classifiers to the pool one after another until a satisfactory criterion is met. The majority of ensemble methods use a set of base classifiers without searching for best combinations. In contrast, we propose a search method for finding the best combination from a set of heterogeneous base classifiers for a given classification problem. The overall aim is to improve the generalisation performance of the majority-vote ensemble of classifier. We use an EA, namely a GA, to identify the optimal combination of base classifiers for the ensemble. Unlike other approaches, we use the MCC to evaluate the performance of the different ensembles of classifiers that are created during the evolutionary search. This approach was developed to fulfil $Obj1$ and $Obj2$ mentioned in Section 1.6 of the thesis.

Obj 1: The first objective of this PhD research was to propose an EoC system that could achieve better performances than its predecessors. To pursue this, we have designed and tested such an algorithm. The system we present explored a complex and large search space posed by the 10-fold CV method using the complete training dataset for 20 base classifiers. We use a GA to explore the search space in parallel and evaluated the performances using the MCC metric. The best ensemble combination found after using the search method for complete training datasets was then used to test the generalisation performances on unseen testing data. Experimental results of the GA-based search for ensemble combination, reported in Chapter 3 are promising. To verify the effectiveness and accuracy of the proposed classification method, we have tested it on datasets from various application domains. The proposed algorithm

outperforms other state-of-the-art EoC methods, which answers research question $Q1$ stated in the Section 1.6.

Obj 2: The next objective of this PhD research is to study the applicability of the proposed ensemble method on high-dimensional datasets. A common feature of many modern classification problems is that the dimensionality of the feature vector is much larger than the available training sample size. Classifiers building on these high-dimensional datasets are very expensive in terms of running time and memory use. Moreover, classification methods using all of these features do not necessarily perform well due to the noise accumulation when estimating a large number of noise features [Fan et al., 2011]. Thus, FS is very important in high-dimensional classification. We applied Fayyad–Irani's entropy filtering method [Fayyad and Irani, 1993] to eliminate noisy features and eliminate redundant data. This well-established filtering method has been used in several articles as data preprocessing for removal of unnecessary features from datasets, especially biological [Ravetti and Moscato, 2008, Rocha de Paula et al., 2011] and big data [Ramírez-Gallego et al., 2016]. It has proven to be an effective filtering method from our empirical study. The resultant dataset provides better training of base classifiers, hence improves generalisation performance. The filtering not only removed noise but also enhanced the data description. To reduce the running time of GA, we distributed the population evaluation process to all cores in the machine. This high level of parallelism improves the overall running time of algorithms.

In the case of EoC methods, the quality of induced models is determined by various characteristics (e.g. accuracy, interpretability). The proposed method in Chapter 3 considers only one aspect (the MCC value of the classifier), while many factors could increase classification quality. Some research was directed to the weighted-voting method for ensemble decision fusion. Different measures were used by different authors as the voting weight of the base classifier. Second, we propose a search method to optimise the weight of heterogeneous base classifiers in a weighted-voting ensemble approach to find the answer to the second research question ($Q2$ in Section 1.6). The principal aim of this approach is to improve the generalisation performances using a weighted vote and was investigated in Chapter 3. The classification performance of weighted-voting EoC depends on the proper choice of weights for each base classifiers vote. We propose the use of a DE algorithm for the weight adjustment of base classifiers used in weighted-voting HEoC. The voting weights of base classifiers are optimised for the HEoC aiming to attain better generalisation performances on testing datasets. The experimental outcome revealed that

weights optimised using an EA, namely DE, optimised weights of the base classifier and the ensemble performed better than its base classifiers and state-of-the-art EoCs.

Then, we extended the quest to find the best combination of base classifiers for ensembles using MOO. Here, we had the opportunity to investigate other objective functions alongside the optimisation of MCC scores as used in the single-objective GAs. This investigation answered research question $Q4$ in Section 1.6. The multi-objective EoC for ensemble combination search was described in Chapter 5. We have increased the number of objective functions aligned as minimising the number of base classifiers used to create the ensemble, maximise the MCC and maximise the diversity among base classifiers. It has now become more capable of handling challenging and complex problems. Then, we extended the approach to meet our research objectives $Obj3$ and $Obj4$ by incorporating the FS method as a wrapper approach with the ensemble combination search.

Obj 3: The final objective of this PhD research is to offer an efficient and effective framework of an ensemble classification system for large-scale datasets. The component algorithms developed from previous objectives are integrated here. The integration of a GA-based ensemble of classification systems, with effective dimensionality reduction in the system and capability to handle imbalanced data, has put this framework into a level of an efficient and effective system. It has been demonstrated that the integrated system played a significant role in achieving higher generalisation performance than its base classifiers. We have evaluated the integrated method with some benchmarking datasets to test the capability of handling the imbalance nature and dimensionality problems in the biological datasets. The consolidated proposed techniques facilitate the framework for future research into data classification using the EoC method, powered by a GA-based search for better combinations.

Obj 4: FS is an integral part of the classification method. It plays a significant role in achieving the accuracy of the classification task. We have evaluated some available state-of-the-art FS techniques to reduce dimensionality problems in biological datasets as an integral part of a GA-based ensemble combination search in a multi-objective setting (MO-EoC-WFS). The consolidated proposed techniques facilitate the framework for future research into data classification by the wrapper-based FS with improved generalisation by EoCs.

To evaluate the system performance and classification accuracy, we need to integrate two methods (FS and classification) and form a test for interoperability. The consolidated proposed techniques will facilitate the framework for future research into biological data

mining using GA, and its variants. Incorporating FS methods will help the classifiers to obtain better training. Training with a good feature subset can improve the classification performance dramatically. So, we have investigated some state-of-the-art FS methods and incorporated those FS techniques as the wrapper of FS and EoC in multi-objective settings.

Most of the times machine learning approaches are criticised for *overfitting* problem. When a machine learning algorithm expresses significantly different between the training and testing performance is referred as the overfitting problem. The term 'overfitting' indicates that the problem originated from the learning algorithms fitting to peculiarities in the training data and does not adequately represent the problem domain. Which in turn produces the low generalisation performance in testing data. Several techniques are available to eliminate this problem. Widely used techniques is the use of a validation set for internal evaluation of performance of the algorithm. It might not be possible to eliminate the overfitting problem if adequate training samples not available for the problem. In case of the wrapper method, a large number of training samples also become computationally expensive for model building and evaluation of feature subset. To reduce the computational time, but to prevent the *overfitting* problem in MO-EoC-WFS, we separated validation data from the training set instead of using the popular 10-fold cross validation. According to [Cunningham, 2000], an ensemble of classifiers able to avoid the overfitting problem if the object of the wrapper feature selection is to build a better classifier, rather that provide insight into the relative importance of features in the domain. We worked towards the same goal of building a better ensemble of classifiers using the feature subset selected from the wrapper, not discovering the insight into the importance of features. Hence, the proposed MO-EoC-WFS method is free from the overfitting problem occurred in most of the machine learning algorithms.

6.2 Future Challenges

Improve the Runtime Performance: The main disadvantage of using GA is its computational cost. This computational overhead primarily arises from the computation of the fitness of individuals. We need to train each individual using the k-fold CV method on the training dataset, and evaluate for each test folds through the evolutionary process. Here, the training phase requires much time because of the k-fold CV training of several single classifiers and combining them. Parallel and distributed implementations are probably the most obvious approach to reduce this running time. To find the best ensemble combination, the algorithm has to evaluate

generations of individuals until the stopping criteria are satisfied. These expensive evaluations are the bottleneck of the current approach. To overcome these runtime performance bottlenecks, we utilised the fork–join framework of Java 7, which helped us to improve the running time performances by reducing the overall CPU times. This approach can be further improved in runtime performance if implemented for a distributed computing approach. Dynamic population sizing-based GA [Eskandari et al., 2007] could be another alternative approach to be used for reducing the running time.

High-dimensional Dataset: The proposed method requires the classification model to be built on the training dataset and evaluated for performance on a testing dataset. Usually, it takes a long time to train a classifier for high-dimensional datasets. The proposed method uses k-fold CV to evaluate the performance of an ensemble combination. It needs to build 10 models for each classifier in the combination. This leads to a massive runtime overhead. For a high-dimensional dataset, this will be a tough obstacle to be handled. In this regard, using an approximation method (like surrogate model [Jin, 2011, Rosales-Pérez et al., 2013]) instead of exact building and evaluation of the ensemble combination model, could be an alternative approach to be further investigated for large datasets.

Feature subset-select: FS or gene selection for biomarker discovery is one of the most challenging tasks to be performed. To select a suitable subset of the gene or candidate biomarker efficiently, we have used only rank-based FS in the wrapper candidate. To keep the computational time limited, we used feature-ranking approaches. A better result might be achievable in terms of FS from MO-EoC-WFS by incorporating other advanced search-based feature subset selection methods.

6.3 Conclusion

The experimental results suggest that the proposed method is a promising approach, and better than using single classifiers and other state-of-the-art ensemble approaches. We tested our method on both binary-class and multiclass datasets, for which it performed well. However, our method suffers because of the runtime required when dealing with large datasets. We can improve the runtime of our method by applying the grid computing parallelisation approach and/or adopting an approximation-based approach. The experimental results of the proposed method can be improved by adopting other manners because we have used very simple settings for generating the EoC. For example, better

classification accuracy can be achieved by fine-tuning the parameters of individual classifiers. In these ways, we can enhance the performance and robustness of the proposed method in the future. Moreover, we have extended the algorithm to simultaneous selection of a feature subset and optimisation of base classifier combination using a multi-objective approach. The integrated approach performed very well in feature reduction and classification for datasets taken from diverse domains, especially biological datasets.

References

[Afsari et al., 2013] Afsari, F., Eftekhari, M., Eslami, E., and Woo, P.-Y. (2013). Interpretability-based fuzzy decision tree classifier a hybrid of the subtractive clustering and the multi-objective evolutionary algorithm. *Soft Computing*, 17(9):1673–1686. 67, 68

[Aha et al., 1991] Aha, D. W., Kibler, D., and Albert, M. K. (1991). Instance-based learning algorithms. *Machine learning*, 6(1):37–66. 41, 121

[Ahmadian et al., 2007] Ahmadian, K., Golestani, A., Analoui, M., and Jahed, M. (2007). Evolving ensemble of classifiers in low-dimensional spaces using multi-objective evolutionary approach. In *Computer and Information Science, 2007. ICIS 2007. 6th IEEE/ACIS International Conference on*, pages 217–222. 116, 117, 118, 129, 130

[Aksela and Laaksonen, 2006] Aksela, M. and Laaksonen, J. (2006). Using diversity of errors for selecting members of a committee classifier. *Pattern Recognition*, 39(4):608–623. 138

[Allison et al., 2006] Allison, D. B., Cui, X., Page, G. P., and Sabripour, M. (2006). Microarray data analysis: from disarray to consolidation and consensus. *Nat Rev Genet*, 7(1):55–65. 4

[Babu, 2004] Babu, M. M. (2004). An Introduction to Microarray Data Analysis. In Grant, R. P., editor, *Computational Genomics: Theory and Application*, chapter 11, pages 225–249. Horizon Press, U.K. 3, 4

[Baraldi et al., 2011] Baraldi, P., Razavi-Far, R., and Zio, E. (2011). Classifier-ensemble incremental-learning procedure for nuclear transient identification at different operational conditions. *Reliability Engineering & System Safety*, 96(4):480 – 488. 14

[Basgalupp et al., 2015] Basgalupp, M. P., Barros, R. C., and Podgorelec, V. (2015). Evolving decision-tree induction algorithms with a multi-objective hyper-heuristic. In *Proceedings of the 30th Annual ACM Symposium on Applied Computing*, pages 110–117. ACM. 31

[Bashir et al., 2015a] Bashir, S., Qamar, U., and Khan, F. (2015a). BagMOOV: A novel ensemble for heart disease prediction bootstrap aggregation with multi-objective optimized voting. *Australasian Physical & Engineering Sciences in Medicine*, pages 1–19. 117

[Bashir et al., 2015b] Bashir, S., Qamar, U., and Khan, F. H. (2015b). A Multicriteria Weighted Vote-Based Classifier Ensemble for Heart Disease Prediction. *Computational Intelligence*. 106, 107

[Bashir et al., 2015c] Bashir, S., Qamar, U., and Khan, F. H. (2015c). Heterogeneous classifiers fusion for dynamic breast cancer diagnosis using weighted vote based ensemble. *Quality & Quantity*, 49(5):2061–2076. 88

[Bauer and Kohavi, 1999] Bauer, E. and Kohavi, R. (1999). An Empirical Comparison of Voting Classification Algorithms: Bagging, Boosting, and Variants. *Machine Learning*, 36(1-2):105–139. 87

[Bellman, 1961] Bellman, R. E. (1961). *Adaptive Control Processes*. Princeton University Press. 16

[Ben-David, 1992] Ben-David, A. (1992). Automatic generation of symbolic multiattribute ordinal knowledge-based dsss: methodology and applications. *Decision Sciences*, 23:1357–1357. 121

[Berretta et al., 2008] Berretta, R., Costa, W., and Moscato, P. (2008). Combinatorial optimization models for finding genetic signatures from gene expression datasets. *Bioinformatics: Structure, Function and Applications*, pages 363–377. 44

[Berretta et al., 2005] Berretta, R., Mendes, A., and Moscato, P. (2005). Integer programming models and algorithms for molecular classification of cancer from microarray data. In *Proceedings of the Twenty-eighth Australasian conference on Computer Science - Volume 38*, ACSC '05, pages 361–370, Darlinghurst, Australia, Australia. Australian Computer Society, Inc. 43, 56

[Berretta et al., 2007a] Berretta, R., Mendes, A., and Moscato, P. (2007a). Selection of discriminative genes in microarray experiments using mathematical programming. *Journal of Research and Practice in Information Technology*, 39(4):287–299. 44

[Berretta et al., 2007b] Berretta, R., Mendes, A., and Moscato, P. (2007b). Selection of discriminative genes in microarray experiments using mathematical programming. *Journal of Research and Practice in Information Technology*, 39(4):287–299. 152

[Bhadra et al., 2012] Bhadra, T., Bandyopadhyay, S., and Maulik, U. (2012). Differential Evolution Based Optimization of SVM Parameters for Meta Classifier Design. *Procedia Technology*, 4:50–57. 2nd International Conference on Computer, Communication, Control and Information Technology (C3IT-2012) on February 25-26, 2012. 88

[Blagus and Lusa, 2010] Blagus, R. and Lusa, L. (2010). Class prediction for high-dimensional class-imbalanced data. *BMC bioinformatics*, 11(1):523. 5

[Blanco et al., 2004] Blanco, R., Larrañaga, P., Inza, I., and Sierra, B. (2004). Gene selection for cancer classification using wrapper approaches. *International Journal of Pattern Recognition and Artificial Intelligence*, 18(08):1373–1390. 143

[Breiman, 1996] Breiman, L. (1996). Bagging Predictors. *Machine Learning*, 24(2):123–140. 13, 98

[Breiman, 2001] Breiman, L. (2001). Random Forests. *Machine Learning*, 45(1):5–32. 22, 36, 41, 98

[Breiman et al., 1984] Breiman, L., Friedman, J., Stone, C. J., and Olshen, R. A. (1984). *Classification and regression trees*. CRC press. 121

[Brown et al., 2005] Brown, G., Wyatt, J., Harris, R., and Yao, X. (2005). Diversity creation methods: a survey and categorisation. *Information Fusion*, 6(1):5 – 20. 32

[Burke et al., 2003] Burke, E., Kendall, G., Newall, J., Hart, E., Ross, P., and Schulenburg, S. (2003). Hyper-heuristics: An emerging direction in modern search technology. In *Handbook of metaheuristics*, pages 457–474. Springer. 31

[Burke et al., 2009] Burke, E. K., Hyde, M. R., Kendall, G., Ochoa, G., Ozcan, E., and Woodward, J. R. (2009). Exploring hyper-heuristic methodologies with genetic programming. In *Computational intelligence*, pages 177–201. Springer. 31

[Burke et al., 2007] Burke, E. K., McCollum, B., Meisels, A., Petrovic, S., and Qu, R. (2007). A graph-based hyper-heuristic for educational timetabling problems. *European Journal of Operational Research*, 176(1):177–192. 31

[Caramia and Dell'Olmo, 2008] Caramia, M. and Dell'Olmo, P. (2008). *Multi-objective Management in Freight Logistics: Increasing Capacity, Service Level and Safety with Optimization Algorithms*, chapter Multi-objective Optimization, pages 11–36. Springer London, London. 114, 115

[Chandra and Yao, 2006] Chandra, A. and Yao, X. (2006). Ensemble learning using multi-objective evolutionary algorithms. *Journal of Mathematical Modelling and Algorithms*, 5(4):417–445. 113

[Chang and Lin, 2011] Chang, C.-C. and Lin, C.-J. (2011). Libsvm: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27. 41, 121

[Chatterjee and Rakshit, 2004] Chatterjee, A. and Rakshit, A. (2004). Influential rule search scheme (IRSS) - a new fuzzy pattern classifier. *IEEE Transactions on Knowledge and Data Engineering*, 16(8):881–893. 67

[Chiachia et al., 2014] Chiachia, G., Falcao, A., Pinto, N., Rocha, A., and Cox, D. (2014). Learning person-specific representations from faces in the wild. *Information Forensics and Security, IEEE Transactions on*, 9(12):2089–2099. 55

[Chiu and Verma, 2014] Chiu, C.-Y. and Verma, B. (2014). Multi-objective evolutionary algorithm based optimization of neural network ensemble classifier. In *Signal Processing and Communication Systems (ICSPCS), 2014 8th International Conference on*, pages 1–5. 117, 118, 129

[Christopher et al., 1997] Christopher, A., Andrew, M., and Stefan, S. (1997). Locally weighted learning. *Artif Intell Rev*, 11(1-5):11–73. 121

[Cleary et al., 1995] Cleary, J. G., Trigg, L. E., et al. (1995). K*: An instance-based learner using an entropic distance measure. In *Proceedings of the 12th International Conference on Machine learning*, volume 5, pages 108–114. 121

[Cleofas et al., 2009] Cleofas, L., Valdovinos, R. M., García, V., Alejo, R., Universitario, C., and Valle, U. (2009). Use of Ensemble Based on GA for Imbalance Problem. In *6th International Symposium on Neural Networks, ISNN 2009 Wuhan, China, May 26-29, 2009 Proceedings, Part II*, pages 547–554. Springer Berlin Heidelberg. 26, 28, 36

[Cohen, 1995] Cohen, W. W. (1995). Fast effective rule induction. In *Proceedings of the twelfth international conference on machine learning*, pages 115–123. 41, 121

[Cohen, 2007] Cohen, W. W. (2007). *A Computer Scientist's Guide to Cell Biology*. Springer. 3

[Cotta et al., 2005] Cotta, C., Langston, M., and Moscato, P. (2005). Combinatorial and algorithmic issues for microarray data analysis. In *In: Handbook of Approximation Algorithms and Metaheuristics*. 152

[Cox and Pinto, 2011] Cox, D. and Pinto, N. (2011). Beyond simple features: A large-scale feature search approach to unconstrained face recognition. In *Automatic Face Gesture Recognition and Workshops (FG 2011), 2011 IEEE International Conference on*, pages 8–15. 55

[Cox, 2005] Cox, E. (2005). *Fuzzy modeling and genetic algorithms for data mining and exploration*. The Morgan Kaufmann series in data management systems. Elsevier, Amsterdam. 46, 50

[Crick, 1970] Crick, F. (1970). Central Dogma of Molecular Biology. *Nature*, 227:561–563. 2

[Cuevas et al., 2016] Cuevas, E., Zaldívar, D., and Perez-Cisneros, M. (2016). Image segmentation based on differential evolution optimization. In *Applications of Evolutionary Computation in Image Processing and Pattern Recognition*, pages 9–22. Springer. 88

[Cunningham, 2000] Cunningham, P. (2000). Overfitting and diversity in classification ensembles based on feature selection. *Trinity College Dublin, Dublin (Ireland), Computer Science Technical Report: TCD-CS-2000-07*. 175

[Cunningham and Carney, 2000] Cunningham, P. and Carney, J. (2000). Diversity versus quality in classification ensembles based on feature selection. In López de Mántaras, R. and Plaza, E., editors, *Machine Learning: ECML 2000*, volume 1810 of *Lecture Notes in Computer Science*, pages 109–116. Springer Berlin Heidelberg. 143

[Curtis et al., 2012] Curtis, C., Shah, S. P., Chin, S.-F., Turashvili, G., Rueda, O. M., Dunning, M. J., Speed, D., Lynch, A. G., Samarajiwa, S., Yuan, Y., et al. (2012). The genomic and transcriptomic architecture of 2,000 breast tumours reveals novel subgroups. *Nature*, 486(7403):346–352. 164

[Das and Suganthan, 2011] Das, S. and Suganthan, P. N. (2011). Differential Evolution: A Survey of the State-of-the-Art. *Evolutionary Computation, IEEE Transactions on*, 15(1):4–31. 89

[Deb et al., 2002] 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. 116, 131

[Demiröz and Güvenir, 1997] Demiröz, G. and Güvenir, H. A. (1997). Classification by voting feature intervals. In *European Conference on Machine Learning*, pages 85–92. Springer. 121

[Demšar, 2006] Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *Journal of Machine learning research*, 7(Jan):1–30. 65

[Dietterich, 2000] Dietterich, T. (2000). Ensemble methods in machine learning. In *Multiple Classifier Systems*, volume 1857 of *Lecture Notes in Computer Science*, pages 1–15. Springer Berlin Heidelberg. 22

[Dietterich, 1997] Dietterich, T. G. (1997). Machine Learning Research: Four Current Directions. *AI Magazine*, 18(4):97–136. 115

[Ding and Zhang, 2008] Ding, Y.-S. and Zhang, T.-L. (2008). Using Chou's pseudo amino acid composition to predict subcellular localization of apoptosis proteins: An approach with immune genetic algorithm-based ensemble classifier. *Pattern Recognition Letters*, 29(13):1887 – 1892. 35

[Dos Santos et al., 2006] Dos Santos, E., Sabourin, R., and Maupin, P. (2006). Single and Multi-Objective Genetic Algorithms for the Selection of Ensemble of Classifiers. *The 2006 IEEE International Joint Conference on Neural Network Proceedings*, pages 3070–3077. 116, 117, 118, 130

[Dos Santos et al., 2008] Dos Santos, E. M., Sabourin, R., and Maupin, P. (2008). Pareto analysis for the selection of classifier ensembles. In *Proceedings of the 10th Annual Conference on Genetic and Evolutionary Computation*, GECCO '08, pages 681–688, New York, NY, USA. ACM. 116, 117, 118, 130

[Duangsoithong and Windeatt, 2010] Duangsoithong, R. and Windeatt, T. (2010). Bootstrap feature selection for ensemble classifiers. In *Proceedings of the 10th industrial conference on Advances in data mining: applications and theoretical aspects*, ICDM'10, pages 28–41, Berlin, Heidelberg. Springer-Verlag. 143

[Durillo and Nebro, 2011] Durillo, J. J. and Nebro, A. J. (2011). jMetal: A Java framework for multi-objective optimization. *Advances in Engineering Software*, 42(10):760–771. 97

[Dutt and Madan, 2012] Dutt, R. and Madan, A. (2012). Predicting biological activity: Computational approach using novel distance based molecular descriptors. *Computers in Biology and Medicine*, 42(10):1026 – 1041. 9, 89

[Eiben and Smith, 2012] Eiben, A. E. and Smith, J. E. (2012). Evolutionary algorithms. In Neri, F., Cotta, C., and Moscato, P., editors, *Handbook of Memetic Algorithms*, volume 379 of *Studies in Computational Intelligence*, pages 9–27. Springer Berlin Heidelberg. 15

[Ekbal and Saha, 2011] Ekbal, A. and Saha, S. (2011). Weighted Vote-Based Classifier Ensemble for Named Entity Recognition: A Genetic Algorithm-Based Approach. *ACM Transactions on Asian Language Information Processing (TALIP)*, 10(2):9:1–9:37. 88, 89

[Eskandari et al., 2007] Eskandari, H., Geiger, C. D., and Lamont, G. B. (2007). *Evolutionary Multi-Criterion Optimization: 4th International Conference, EMO 2007, Matsushima, Japan, March 5-8, 2007. Proceedings*, chapter FastPGA: A Dynamic Population Sizing Approach for Solving Expensive Multiobjective Optimization Problems, pages 141–155. Springer Berlin Heidelberg, Berlin, Heidelberg. 176

[Espejo et al., 2010] Espejo, P., Ventura, S., and Herrera, F. (2010). A survey on the application of genetic programming to classification. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 40(2):121–144. 26

[Fan et al., 2011] Fan, J., Fan, Y., and Wu, Y. (2011). High-dimensional Classification. In Cai, T. and Shen, X., editors, *High-dimensional Data Analysis*, chapter 1, pages 3–37. World Scientific, New Jersey. 173

[Fayyad and Irani, 1993] Fayyad, U. and Irani, K. (1993). Multi-interval discretization of continuous-valued attributes for classification learning. In *International Joint Conference on Artificial Intelligence*, page 10221029. Morgan Kaufmann. 43, 55, 173

[Fernández et al., 2010] Fernández, A., García, S., Luengo, J., Bernadó-Mansilla, E., and Herrera, F. (2010). Genetics-based machine learning for rule induction: State of the art, taxonomy, and comparative study. *Trans. Evol. Comp*, 14(6):913–941. 60, 69

[Filiou et al., 2014] Filiou, M. D., Arefin, A. S., Moscato, P., and Graeber, M. B. (2014). 'Neuroinflammation' differs categorically from inflammation: transcriptomes of Alzheimer's disease, Parkinson's disease, schizophrenia and inflammatory diseases compared. *neurogenetics*, 15(3):201–212. 119, 168

[Frank and Witten, 1998] Frank, E. and Witten, I. H. (1998). Generating accurate rule sets without global optimization. pages 144–151. 41, 121

[Freund and Mason, 1999] Freund, Y. and Mason, L. (1999). The alternating decision tree learning algorithm. In *Proceeding of the Sixteenth International Conference on Machine Learning*, pages 124–133, Bled, Slovenia. 121

[Freund and Schapire, 1996] Freund, Y. and Schapire, R. E. (1996). Experiments with a new boosting algorithm. In *Thirteenth International Conference on Machine Learning*, pages 148–156, San Francisco. Morgan Kaufmann. 13, 22, 98

[Freund and Schapire, 1998] Freund, Y. and Schapire, R. E. (1998). Large margin classification using the perceptron algorithm. In *11th Annual Conference on Computational Learning Theory*, pages 209–217, New York, NY. ACM Press. 121

[Freund and Schapire, 1999] Freund, Y. and Schapire, R. E. (1999). Large margin classification using the perceptron algorithm. *Machine learning*, 37(3):277–296. 41

[Friedman et al., 2000] Friedman, J., Hastie, T., and Tibshirani, R. (2000). Additive logistic regression : A statistical view of boosting. *Annals of statistics*, 28(2):337–407. 121

[Gaber and Bader-El-Den, 2012] Gaber, M. M. and Bader-El-Den, M. (2012). Optimisation of Ensemble Classifiers using Genetic Algorithm. In Graña, M., Toro, C., Posada, J., Howlett, R. J., and Jain, L. C., editors, *Advances in Knowledge-Based and Intelligent Information and Engineering Systems*. IOS Press. 26, 28, 30

[Gabrys and Ruta, 2006] Gabrys, B. and Ruta, D. (2006). Genetic algorithms in classifier fusion. *Applied Soft Computing*, 6(4):337–347. 27, 29, 30

[Gaines and Compton, 1995] Gaines, B. R. and Compton, P. (1995). Induction of ripple-down rules applied to modeling large databases. *Journal of Intelligent Information Systems*, 5(3):211–228. 121

[Galar et al., 2012] Galar, M., Fernandez, A., Barrenechea, E., Bustince, H., and Herrera, F. (2012). A review on ensembles for the class imbalance problem: Bagging-, boosting-, and hybrid-based approaches. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 42(4):463–484. 24

[Galar et al., 2011] Galar, M., Fernndez, A., Barrenechea, E., Bustince, H., and Herrera, F. (2011). An overview of ensemble methods for binary classifiers in multi-class problems: Experimental study on one-vs-one and one-vs-all schemes. *Pattern Recognition*, 44(8):1761 – 1776. 23

[Gama, 2004] Gama, J. (2004). Functional trees. *Machine Learning*, 55(3):219–250. 121

[Georgiou et al., 2006] Georgiou, H., Mavroforakis, M., and Theodoridis, S. (2006). A Game-Theoretic Approach to Weighted Majority Voting for Combining SVM Classifiers. In Kollias, S., Stafylopatis, A., Duch, W., and Oja, E., editors, *Artificial Neural Networks - ICANN 2006*, volume 4131 of *Lecture Notes in Computer Science*, pages 284–292. Springer Berlin Heidelberg. 88

[Geurts et al., 2006] Geurts, P., Ernst, D., and Wehenkel, L. (2006). Extremely randomized trees. *Machine Learning*, 63(1):3–42. 121

[Giacinto and Roli, 2000] Giacinto, G. and Roli, F. (2000). Dynamic classifier selection. In *Multiple Classifier Systems*, pages 177–189. Springer. 14

[Gu et al., 2015] Gu, S., Cheng, R., and Jin, Y. (2015). Multi-objective ensemble generation. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 5(5):234–245. 113, 115

[Guyon et al., 2004] Guyon, I., Gunn, S., Ben-Hur, A., and Dror, G. (2004). Result analysis of the nips 2003 feature selection challenge. In *Advances in Neural Information Processing Systems*, pages 545–552. 151

[Guyon et al., 2007] Guyon, I., Li, J., Mader, T., Pletscher, P. A., Schneider, G., and Uhr, M. (2007). Competitive Baseline Methods Set New Standards for the NIPS 2003 Feature Selection Benchmark. *Pattern Recogn. Lett.*, 28(12):1438–1444. 158

[Hadka, 2014] Hadka, D. (2014). MOEA Framework: A Free and Open Source Java Framework for Multiobjective Optimization. 134

[Hall et al., 2009] Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. (2009). The WEKA Data Mining Software: An Update. *SIGKDD Explorations Newsletter*, 11(1):10–18. xxiii, 40, 41, 92, 97, 98, 103, 119, 121, 124, 133, 145

[Hall and Frank, 2008] Hall, M. A. and Frank, E. (2008). Combining naive bayes and decision tables. In *FLAIRS Conference*, volume 2118, pages 318–319. 121

[Hamilton et al., 1996] Hamilton, H. J., Shan, N., Cercone, N., Hamilton, H. J., Shan, N., Cercone, N., Hamilton, H. J., Shan, N., and Cercone, N. (1996). RIAC : A Rule Induction Algorithm Based on Approximate Classification. Technical report, Technical Report CS 96-06, University of Regina. 67

[Han et al., 2012] Han, M., Zhu, X., and Yao, W. (2012). Remote sensing image classification based on neural network ensemble algorithm. *Neurocomputing*, 78:133–138. 14

[Haque et al., 2016a] Haque, M. N., Noman, N., Berretta, R., and Moscato, P. (2016a). Heterogeneous ensemble combination search using genetic algorithm for class imbalanced data classification. *PLoS ONE*, 11(1):e0146116. vii, 128, 152

[Haque et al., 2016b] Haque, M. N., Noman, N., Berretta, R., and Moscato, P. (2016b). Optimising weights for heterogeneous ensemble of classifiers with differential evolution. In *2016 IEEE Congress on Evolutionary Computation (CEC)*, pages 233–240. vii

[Hart and Sim, 2016] Hart, E. and Sim, K. (2016). A hyper-heuristic ensemble method for static job-shop scheduling. *Evolutionary computation*. 31

[Haupt and Haupt, 2004] Haupt, R. L. and Haupt, S. E. (2004). *Practical Genetic Algorithms with CD-ROM*. Wiley-Interscience. 15

[Hegde et al., 2007] Hegde, C., Davenport, M. A., Wakin, M. B., and Baraniuk, R. G. (2007). Efficient machine learning using random projections. In *Neural Information Processing Systems (NIPS) Workshop on Efficient Machine Learning*, Whistler, BC. 17

[Hegerty et al., 2009] Hegerty, B., Hung, C.-C., and Kasprak, K. (2009). A comparative study on differential evolution and genetic algorithms for some combinatorial problems. In *Proceedings of 8th Mexican International Conference on Artificial Intelligence*, pages 9–13. 88

[Ho, 1998] Ho, T. K. (1998). The random subspace method for constructing decision forests. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 20(8):832–844. 13

[Holmes et al., 2001] Holmes, G., Pfahringer, B., Kirkby, R., Frank, E., and Hall, M. (2001). Multiclass alternating decision trees. In *ECML*, pages 161–172. Springer. 121

[Holte, 1993] Holte, R. C. (1993). Very simple classification rules perform well on most commonly used datasets. *Machine learning*, 11(1):63–90. 41

[Hong and Cho, 2006] Hong, J.-H. and Cho, S.-B. (2006). The classification of cancer based on DNA microarray data that uses diverse ensemble genetic programming . *Artificial Intelligence in Medicine*, 36(1):43 – 58. 33, 34

[Hossen et al., 2013] Hossen, J., Sayeed, S., Yusof, I., and Kalaiarasi, S. M. A. (2013). A Framework of Modified Adaptive Fuzzy Inference Engine (MAFIE) and Its Application. *International Journal of Computer Information Systems and Industrial Management Applications*, 5:662–670. 67

[Hourani et al., 2008] Hourani, M., Berretta, R., Mendes, A., and Moscato, P. (2008). Genetic signatures for a rodent model of parkinson's disease using combinatorial optimization methods. *Bioinformatics: Structure, Function and Applications*, pages 379–392. 44

[Hu et al., 2008] Hu, H., Li, J., Wang, H., and Daggard, G. (2008). Robustness analysis of diversified ensemble decision tree algorithms for Microarray data classification. *2008 International Conference on Machine Learning and Cybernetics*, pages 115–120. 24

[Hughes, 1968] Hughes, G. (1968). On the mean accuracy of statistical pattern recognizers. *IEEE Transactions on Information Theory*, 14(1):55–63. 17

[Hühn and Hüllermeier, 2009] Hühn, J. and Hüllermeier, E. (2009). Furia: an algorithm for unordered fuzzy rule induction. *Data Mining and Knowledge Discovery*, 19(3):293–319. 121

[Hulten et al., 2001] Hulten, G., Spencer, L., and Domingos, P. (2001). Mining time-changing data streams. In *ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining*, pages 97–106. ACM Press. 121

[Huong et al., 2015] Huong, D. T. T., Truong, V. V., and Lam, B. T. (2015). Forecasting of consumer price index using the ensemble learning model with multi-objective evolutionary algorithms: Preliminary results. In *Advanced Technologies for Communications (ATC), 2015 International Conference on*, pages 337–342. 14

[Icke and Rosenberg, 2011] Icke, I. and Rosenberg, A. (2011). Multi-objective genetic programming for visual analytics. In Silva, S., Foster, J., Nicolau, M., Machado, P., and Giacobini, M., editors, *Genetic Programming*, volume 6621 of *Lecture Notes in Computer Science*, pages 322–334. Springer Berlin Heidelberg. 67

[Ilonen et al., 2003] Ilonen, J., Kamarainen, J.-K., and Lampinen, J. (2003). Differential evolution training algorithm for feed-forward neural networks. *Neural Processing Letters*, 17(1):93–105. 88

[Ishibuchi, 2007] Ishibuchi, H. (2007). Multiobjective genetic fuzzy systems: Review and future research directions. In *Fuzzy Systems Conference, 2007. FUZZ-IEEE 2007. IEEE International*, pages 1–6. 35

[Jain et al., 2000] Jain, A. K., Duin, R. P. W., Mao, J., and Member, S. (2000). Statistical Pattern Recognition : A Review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(1):4–37. 32

[James et al., 2014] James, G., Witten, D., and Hastie, T. (2014). An introduction to statistical learning: With applications in r. 11

[Jang, 1993] Jang, J.-S. (1993). ANFIS: adaptive-network-based fuzzy inference system. *IEEE Transactions on Systems, Man and Cybernetics*, 23(3):665–685. 67, 68

[Jayadeva et al., 2015] Jayadeva, Soman, S., and Bhaya, A. (2015). The MC-ELM: Learning an ELM-like network with minimum VC dimension. In *Neural Networks (IJCNN), 2015 International Joint Conference on*, pages 1–7. 158

[Jensen, 2003] Jensen, M. T. (2003). Reducing the run-time complexity of multiobjective eas: The nsga-ii and other algorithms. *IEEE Transactions on Evolutionary Computation*, 7(5):503–515. 149

[Jin, 2011] Jin, Y. (2011). Surrogate-assisted evolutionary computation: Recent advances and future challenges. *Swarm and Evolutionary Computation*, 1(2):61–70. 176

[Jin and Sendhoff, 2008] 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. 113, 115, 128

[John et al., 1994] John, G. H., Kohavi, R., and Pfleger, K. (1994). Irrelevant features and the subset selection problem. In Hirsh, W. W. C., editor, *Machine Learning Proceedings 1994*, pages 121 – 129. Morgan Kaufmann, San Francisco (CA). 145

[John and Langley, 1995] John, G. H. and Langley, P. (1995). Estimating continuous distributions in bayesian classifiers. In *Proceedings of the Eleventh conference on Uncertainty in artificial intelligence*, pages 338–345. Morgan Kaufmann Publishers Inc. 41, 121

[Jordan and Jacobs, 1994] Jordan, M. I. and Jacobs, R. A. (1994). Hierarchical Mixtures of Experts and the EM Algorithm. *Neural Computation*, 6(2):181–214. 21

[Jurman et al., 2012a] Jurman, G., Riccadonna, S., and Furlanello, C. (2012a). A Comparison of MCC and CEN Error Measures in Multi-Class Prediction. *PLoS ONE*, 7(08):e41882. 93

[Jurman et al., 2012b] Jurman, G., Riccadonna, S., and Furlanello, C. (2012b). A comparison of mcc and cen error measures in multi-class prediction. *PLoS ONE*, 7(8):e41882. 9

[Karaboga, 2005] Karaboga, N. (2005). Digital iir filter design using differential evolution algorithm. *EURASIP Journal on Applied Signal Processing*, 2005:1269–1276. 88

[Karatsiolis and Schizas, 2012] Karatsiolis, S. and Schizas, C. (2012). Region based support vector machine algorithm for medical diagnosis on pima indian diabetes dataset. In *2012 IEEE 12th International Conference on Bioinformatics Bioengineering (BIBE)*, pages 139–144. 67

[Kim and Cho, 2008] Kim, K.-J. and Cho, S.-B. (2008). An evolutionary algorithm approach to optimal ensemble classifiers for dna microarray data analysis. *IEEE Transactions on Evolutionary Computation*, 12(3):377–388. 34, 36

[Kim et al., 2006] Kim, Y., Street, W. N., and Menczer, F. (2006). Optimal ensemble construction via meta-evolutionary ensembles. *Expert Systems with Applications*, 30(4):705 – 714. 26, 28, 30

[Kim and Oh, 2008] Kim, Y.-W. and Oh, I.-S. (2008). Classifier ensemble selection using hybrid genetic algorithms. *Pattern Recognition Letters*, 29(6):796 – 802. 25

[Kira and Rendell, 1992] Kira, K. and Rendell, L. A. (1992). A practical approach to feature selection. In Sleeman, D. H. and Edwards, P., editors, *Ninth International Workshop on Machine Learning*, pages 249–256. Morgan Kaufmann. 119

[Kleinberg, 2000] Kleinberg, E. (2000). On the algorithmic implementation of stochastic discrimination. *IEEE Transcations on Pattern Analysis and Machine Intelligence*, 22(5):473–490. 32

[Kobayashi et al., 2011] Kobayashi, H., Mark, B. L., and Turin, W. (2011). *Probability, Random Processes, and Statistical Analysis*, chapter 2, pages 26–29. Cambridge University Press. 38

[Kohavi, 1995] Kohavi, R. (1995). The power of decision tables. In *European conference on machine learning*, pages 174–189. Springer. 41, 121

[Kohavi, 1996] Kohavi, R. (1996). Scaling up the accuracy of naive-bayes classifiers: A decision-tree hybrid. In *KDD*, volume 96, pages 202–207. Citeseer. 121

[Kohavi and John, 1997] Kohavi, R. and John, G. H. (1997). Wrappers for feature subset selection. *Artificial Intelligence*, 97(12):273 – 324. 144

[Kotsiantis et al., 2006] Kotsiantis, S., Zaharakis, I., and Pintelas, P. (2006). Machine learning: a review of classification and combining techniques. *Artificial Intelligence Review*, 26(3):159–190. 22

[Koutanaei et al., 2015] Koutanaei, F. N., Sajedi, H., and Khanbabaei, M. (2015). A hybrid data mining model of feature selection algorithms and ensemble learning classifiers for credit scoring. *Journal of Retailing and Consumer Services*, 27:11 – 23. 25, 36

[Krawczyk et al., 2016] Krawczyk, B., Galar, M., Jeleń, Ł., and Herrera, F. (2016). Evolutionary undersampling boosting for imbalanced classification of breast cancer malignancy. *Applied Soft Computing*, 38:714–726. 22, 35

[Krawczyk et al., 2013] Krawczyk, B., Schaefer, G., and Wozniak, M. (2013). An evaluation of classifier ensembles for class imbalance problems. *2013 International Conference on Informatics, Electronics and Vision (ICIEV)*, pages 1–4. 33

[Kumar and Kumar, 2012a] Kumar, G. and Kumar, K. (2012a). The Use of Multi-Objective Genetic Algorithm Based Approach to Create Ensemble of ANN for Intrusion Detection. *International Journal of Intelligence Science*, 2(October):115–127. 113

[Kumar and Kumar, 2012b] Kumar, G. and Kumar, K. (2012b). The use of multi-objective genetic algorithm based approach to create ensemble of ann for intrusion detection. *International Journal of Intelligence Science*, 2(4A):115–127. 116, 118

[Kuncheva and Jain, 2000] Kuncheva, L. and Jain, L. (2000). Designing classifier fusion systems by genetic algorithms. *IEEE Transactions on Evolutionary Computation*, 4(4):327–336. 4, 142

[Kuncheva and Whitaker, 2003] Kuncheva, L. and Whitaker, C. (2003). Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine Learning*, 51(2):181–207. 32, 117, 129

[Kuncheva, 2004] Kuncheva, L. I. (2004). *Combining Pattern Classifiers: Methods and Algorithms*. John Wiley & Sons, Inc. 5, 6, 14, 22, 86

[Kuncheva, 2014] Kuncheva, L. I. (2014). *Combining Pattern Classifiers: Methods and Algorithms, 2nd Edition*. John Wiley & Sons, Inc. 11

[Kuncheva and Rodríguez, 2014] Kuncheva, L. I. and Rodríguez, J. J. (2014). A weighted voting framework for classifiers ensembles. *Knowledge and Information Systems*, 38(2):259–275. 88

[Landwehr et al., 2005a] Landwehr, N., Hall, M., and Frank, E. (2005a). Logistic model trees. *Machine Learning*, 59(1-2):161–205. 41

[Landwehr et al., 2005b] Landwehr, N., Hall, M., and Frank, E. (2005b). Logistic model trees. *Machine Learning*, 95(1-2):161–205. 121

[Le Cessie and Van Houwelingen, 1992] Le Cessie, S. and Van Houwelingen, J. C. (1992). Ridge estimators in logistic regression. *Applied statistics*, pages 191–201. 41, 121

[Lee et al., 2007] Lee, C., Zaknick, A., and Braunl, T. (2007). An Adaptive T-S type Rough-Fuzzy Inference System (ARFIS) for Pattern Classification. In *Fuzzy Information Processing Society, 2007. NAFIPS '07. Annual Meeting of the North American*, pages 117–122. 67

[Lekkas and Mikhailov, 2010] Lekkas, S. and Mikhailov, L. (2010). Evolving fuzzy medical diagnosis of pima indians diabetes and of dermatological diseases. *Artificial Intelligence in Medicine*, 50(2):117 – 126. 67

[Lertampaiporn et al., 2013] Lertampaiporn, S., Thammarongtham, C., Nukoolkit, C., Kaewkamnerdpong, B., and Ruengjitchatchawalya, M. (2013). Heterogeneous ensemble approach with discriminative features and modified-SMOTEbagging for pre-miRNA classification. *Nucleic acids research*, 41(1):e21. 27, 29, 30

[Levesque et al., 2012] Levesque, J.-C., Durand, A., Gagne, C., and Sabourin, R. (2012). Multi-objective evolutionary optimization for generating ensembles of classifiers in the roc space. In *Proceedings of the 14th Annual Conference on Genetic and Evolutionary Computation*, GECCO '12, pages 879–886, New York, NY, USA. ACM. 116, 118

[Li, 2007] Li, C. (2007). Classifying imbalanced data using a bagging ensemble variation (BEV). In *Proceedings of the 45th annual southeast regional conference*, ACM-SE 45, pages 203–208, New York, NY, USA. ACM. 32, 36

[Li et al., 2004] Li, X., Rao, S., Wang, Y., and Gong, B. (2004). Gene mining: a novel and powerful ensemble decision approach to hunting for disease genes using microarray expression profiling. *Nucleic Acids Research*, 32(9):2685–2694. 33

[Liao et al., 2014] Liao, J. J., Shih, C. H., Chen, T. F., and Hsu, M. F. (2014). An ensemble-based model for two-class imbalanced financial problem. *Economic Modelling*, 37:175–183. 14

[Lichman, 2013] Lichman, M. (2013). UCI Machine Learning Repository. 52, 53, 97, 106, 151

[Lin et al., 2013] Lin, C., Zou, Y., Qin, J., Liu, X., Jiang, Y., Ke, C., and Zou, Q. (2013). Hierarchical classification of protein folds using a novel ensemble classifier. *PloS one*, 8(2):e56499. 33, 36

[Liu and Xu, 2009] Liu, K.-H. and Xu, C.-G. (2009). A genetic programming-based approach to the classification of multiclass microarray datasets. *Bioinformatics (Oxford, England)*, 25(3):331–7. 10

[Liu et al., 2014] Liu, N., Cao, J., Lin, Z., Pek, P. P., Koh, Z. X., and Ong, M. E. H. (2014). Evolutionary Voting-Based Extreme Learning Machines. *Mathematical Problems in Engineering*, 2014:1–7. 88, 89

[Liu et al., 2006] Liu, Y., An, A., and Huang, X. (2006). Boosting prediction accuracy on imbalanced datasets with SVM ensembles. In *Advances in Knowledge Discovery and Data Mining*, pages 107–118. Springer. 67

[Lovell and Bradley, 1996] Lovell, B. and Bradley, A. (1996). The multiscale classifier. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(2):124–137. 67

[Maghsoudi et al., 2006] Maghsoudi, Y., Alimohammadi, A., Zoej, M. V., and Mojaradi, B. (2006). Weighted combination of multiple classifiers for the classification of hyperspectral images using a genetic algorithm. In *ISPRS Commission VII Mid-term Symposium on Remote Sensing: From Pixels to Processes*. 88

[Mane et al., 2016] Mane, S., Sonawani, S., and Sakhare, S. (2016). Hybrid multi-objective optimization approach for neural network classification using local search. In *Innovations in Computer Science and Engineering*, pages 171–179. Springer. 131

[Mangasarian et al., 1995] Mangasarian, O. L., Street, W. N., and Wolberg, W. H. (1995). Breast cancer diagnosis and prognosis via linear programming. *OPERATIONS RESEARCH*, 43:570–577. 53

[Mao et al., 2015] Mao, S., Jiao, L., Xiong, L., Gou, S., Chen, B., and Yeung, S.-K. (2015). Weighted classifier ensemble based on quadratic form. *Pattern Recognition*, 48(5):1688–1706. 87

[Marsden et al., 2013] Marsden, J., Budden, D., Craig, H., and Moscato, P. (2013). language individuation and marker words: Shakespeare and his maxwells demon. *PLoS One*, 8:e66813. 119, 168

[Martin, 1995] Martin, B. (1995). Instance-based learning: Nearest neighbor with generalization. Master's thesis, University of Waikato, Hamilton, New Zealand. 121

[Martínez-Vargas et al., 2016] Martínez-Vargas, A., Domínguez-Guerrero, J., Andrade, Á. G., Sepúlveda, R., and Montiel-Ross, O. (2016). Application of NSGA-II algorithm to the spectrum assignment problem in spectrum sharing networks. *Applied Soft Computing*, 39:188–198. 131

[Matthews, 1975] Matthews, B. W. (1975). Comparison of the predicted and observed secondary structure of T4 phage lysozyme. *Biochim. Biophys. Acta*, 405(2):442–451. 89

[Mezura-Montes, 2008] Mezura-Montes, E. (2008). Nature-inspired algorithms evolutionary and swarm intelligence approaches. *A Tutorial in MICAI*, 2008. 88

[Milioli et al., 2015] Milioli, H. H., Vimieiro, R., Riveros, C., Tishchenko, I., Berretta, R., and Moscato, P. (2015). The discovery of novel biomarkers improves breast cancer intrinsic subtype prediction and reconciles the labels in the metabric data set. *PloS one*, 10(7):e0129711. 167, 169

[Milioli et al., 2016] Milioli, H. H., Vimieiro, R., Tishchenko, I., Riveros, C., Berretta, R., and Moscato, P. (2016). Iteratively refining breast cancer intrinsic subtypes in the metabric dataset. *BioData mining*, 9(1):1. 167, 169

[Minaei-Bidgoli et al., 2004] Minaei-Bidgoli, B., Kortemeyer, G., and Punch, W. (2004). Optimizing classification ensembles via a genetic algorithm for a web-based educational system. In Fred, A., Caelli, T., Duin, R., Campilho, A., and de Ridder, D., editors, *Structural, Syntactic, and Statistical Pattern Recognition*, volume 3138 of *Lecture Notes in Computer Science*, pages 397–406. Springer Berlin Heidelberg. 142

[Montazeri et al., 2015] Montazeri, M., Baghshah, M. S., and Enhesari, A. (2015). Hyper-heuristic algorithm for finding efficient features in diagnose of lung cancer disease. *arXiv preprint arXiv:1512.04652*. 31

[Moscato et al., 2005] Moscato, P., Berretta, R., Hourani, M., Mendes, A., and Cotta, C. (2005). *Genes Related with Alzheimer's Disease: A Comparison of Evolutionary Search, Statistical and Integer Programming Approaches*, pages 84–94. Springer Berlin Heidelberg, Berlin, Heidelberg. 44

[Moscato et al., 2007] Moscato, P., Mendes, A., and Berretta, R. (2007). Benchmarking a memetic algorithm for ordering microarray data. *Biosystems*, 88(1):56–75. 44

[Nag and Pal, 2016] Nag, K. and Pal, N. R. (2016). A multiobjective genetic programming-based ensemble for simultaneous feature selection and classification. *IEEE Transactions on Cybernetics*, 46(2):499–510. 118, 144

[Namsrai et al., 2013] Namsrai, E., Munkhdalai, T., Li, M., Shin, J.-H., Namsrai, O.-E., and Ryu, K. H. (2013). A Feature Selection-based Ensemble Method for Arrhythmia Classification. *Journal of Information Processing Systems*, 9(1):31–40. 23, 36

[Nguyen et al., 2014] Nguyen, T. T., Liew, A.-C., Pham, X. C., and Nguyen, M. P. (2014). Optimization of ensemble classifier system based on multiple objectives genetic algorithm. In *Machine Learning and Cybernetics (ICMLC), 2014 International Conference on*, volume 1, pages 46–51. 117, 118, 130

[Nikulin et al., 2009] Nikulin, V., Mclachlan, G. J., and Ng, S. K. (2009). Ensemble Approach for the Classification of Imbalanced Data. In *AI 2009: Advances in Artificial Intelligence*, pages 291–300. Springer. 143

[Obiedat et al., 2013] Obiedat, R., Alkasassbeh, M., Faris, H., and Harfoushi, O. (2013). Customer churn prediction using a hybrid genetic programming approach. *Scientific Research and Essays*, 8(27):1289–1295. 108

[Oehmcke et al., 2015] Oehmcke, S., Heinermann, J., and Kramer, O. (2015). Analysis of diversity methods for evolutionary multi-objective ensemble classifiers. In Mora, A. M. and Squillero, G., editors, *Applications of Evolutionary Computation*, volume 9028 of *Lecture Notes in Computer Science*, pages 567–578. Springer International Publishing. 117, 118

[Oh and Gray, 2013] Oh, D.-Y. and Gray, J. B. (2013). GA-Ensemble: a genetic algorithm for robust ensembles. *Computational Statistics*, 28(5):2333–2347. 26, 28, 30

[Oh et al., 2011] Oh, S., Lee, M. S., and Zhang, B. T. (2011). Ensemble learning with active example selection for imbalanced biomedical data classification. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 8:316–325. 14

[Oliveira et al., 2005] Oliveira, L., Morita, M., Sabourin, R., and Bortolozzi, F. (2005). Multi-objective genetic algorithms to create ensemble of classifiers. In Coello Coello, C. A., Hernández Aguirre, A., and Zitzler, E., editors, *Evolutionary Multi-Criterion Optimization*, volume 3410 of *Lecture Notes in Computer Science*, pages 592–606. Springer Berlin Heidelberg. 143

[Oliveira et al., 2003] Oliveira, L. S., Sabourin, R., Bortolozzi, F., and Suen, C. Y. (2003). Feature selection for ensembles: A hierarchical multi-objective genetic algorithm approach. In *Proceedings of the Seventh International Conference on Document Analysis and Recognition - Volume 2*, ICDAR '03, pages 676–, Washington, DC, USA. IEEE Computer Society. 142, 143

[Opitz and Maclin, 1999] Opitz, D. and Maclin, R. (1999). Popular ensemble methods: An empirical study. *Journal of Artificial Intelligence Research*, pages 169–198. 115

[Opitz et al., 1996] Opitz, D. W., Shavlik, J. W., et al. (1996). Generating accurate and diverse members of a neural-network ensemble. *Advances in neural information processing systems*, pages 535–541. 115

[Osareh and Shadgar, 2013] Osareh, A. and Shadgar, B. (2013). An Efficient Ensemble Learning Method for Gene Microarray Classification. *BioMed Research International*, 2013:1–10. 24

[Oza, 2006] Oza, N. C. (2006). Ensemble data mining methods. In Wang, J., editor, *Encyclopedia of Data Warehousing and Mining*, volume 1, pages 448–453. Idea Group Reference. 10

[Oza and Tumer, 2008] Oza, N. C. and Tumer, K. (2008). Classifier ensembles: Select real-world applications. *Information Fusion*, 9(1):4 – 20. 23

[Özcan et al., 2012] Özcan, E., Mısır, M., Ochoa, G., and Burke, E. K. (2012). A reinforcement learning: Great-deluge hyper-heuristic. *Modeling, Analysis, and Applications in Metaheuristic Computing: Advancements and Trends: Advancements and Trends*, page 34. 31

[Pappa et al., 2014] Pappa, G. L., Ochoa, G., Hyde, M. R., Freitas, A. A., Woodward, J., and Swan, J. (2014). Contrasting meta-learning and hyper-heuristic research: the role of evolutionary algorithms. *Genetic Programming and Evolvable Machines*, 15(1):3–35. 31

[Parker et al., 2009] Parker, J. S., Mullins, M., Cheang, M. C., Leung, S., Voduc, D., Vickery, T., Davies, S., Fauron, C., He, X., Hu, Z., et al. (2009). Supervised risk predictor of breast cancer based on intrinsic subtypes. *Journal of clinical oncology*, 27(8):1160–1167. 164

[Peng, 2006] Peng, Y. (2006). A novel ensemble machine learning for robust microarray data classification. *Computers in Biology and Medicine*, 36(6):553 – 573. 33

[Pinto et al., 2011] Pinto, N., Stone, Z., Zickler, T., and Cox, D. (2011). Scaling up biologically-inspired computer vision: A case study in unconstrained face recognition on facebook. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2011 IEEE Computer Society Conference on*, pages 35–42. 53, 55

[Platt, 1998] Platt, J. (1998). Sequential minimal optimization: A fast algorithm for training support vector machines. 121

[Polikar, 2006] Polikar, R. (2006). Ensemble based systems in decision making. *Circuits and Systems Magazine, IEEE*, 6(3):21–45. 13, 14, 22

[Price, 1999] Price, K. V. (1999). An introduction to differential evolution. In Corne, D., Dorigo, M., Glover, F., Dasgupta, D., Moscato, P., Poli, R., and Price, K. V., editors, *New ideas in optimization*, pages 79–108. McGraw-Hill Ltd., UK, Maidenhead, UK, England. 87, 90

[Quinlan, 2014] Quinlan, J. R. (2014). *C4. 5: programs for machine learning*. Elsevier. 41, 121

[Rahman and Verma, 2013] Rahman, A. and Verma, B. (2013). Cluster oriented ensemble classifiers using multi-objective evolutionary algorithm. In *Neural Networks (IJCNN), The 2013 International Joint Conference on*, pages 1–6. 117, 118, 129

[Rakotomalala, 2013] Rakotomalala, R. (2013). Heart Disease Male. (Date last accessed on 9-Nov-2015). 106

[Ramírez-Gallego et al., 2016] Ramírez-Gallego, S., García, S., Mouriño-Talín, H., Martínez-Rego, D., Bolón-Canedo, V., Alonso-Betanzos, A., Benítez, J. M., and Herrera, F. (2016). Data discretization: taxonomy and big data challenge. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 6(1):5–21. 173

[Ranawana and Palade, 2006] Ranawana, R. and Palade, V. (2006). Multi-Classifier Systems: Review and a roadmap for developers. *International Journal of Hybrid Intelligent Systems*, 3(1):35–61. 25

[Ravetti and Moscato, 2008] Ravetti, M. G. and Moscato, P. (2008). Identification of a 5-protein biomarker molecular signature for predicting Alzheimer's disease. *PloS one*, 3(9):e3111. 54, 70, 71, 72, 152, 173

[Ray et al., 2007] Ray, S., Britschgi, M., Herbert, C., Takeda-Uchimura, Y., Boxer, A., Blennow, K., Friedman, L. F., Galasko, D. R., Jutel, M., Karydas, A., Kaye, J. a., Leszek, J., Miller, B. L., Minthon, L., Quinn, J. F., Rabinovici, G. D., Robinson, W. H., Sabbagh, M. N., So, Y. T., Sparks, D. L., Tabaton, M., Tinklenberg, J., Yesavage, J. a., Tibshirani, R., and Wyss-Coray, T. (2007). Classification and prediction of clinical Alzheimer's diagnosis based on plasma signaling proteins. *Nature medicine*, 13(11):1359–62. 53, 54, 70

[Rocha de Paula et al., 2011] Rocha de Paula, M., Gmez Ravetti, M., Berretta, R., and Moscato, P. (2011). Differences in abundances of cell-signalling proteins in blood reveal novel biomarkers for early detection of clinical alzheimer's disease. *PLoS ONE*, 6(3):1–14. 173

[Rokach, 2009] Rokach, L. (2009). Ensemble-based classifiers. *Artificial Intelligence Review*, 33(1-2):1–39. 12

[Roli et al., 2001] Roli, F., Giacinto, G., and Vernazza, G. (2001). Methods for designing multiple classifier systems. In Kittler, J. and Roli, F., editors, *Multiple Classifier Systems*, volume 2096 of *Lecture Notes in Computer Science*, pages 78–87. Springer Berlin Heidelberg. 22

[Rosales-Pérez et al., 2013] Rosales-Pérez, A., Coello, C. A. C., Gonzalez, J. A., Reyes-Garcia, C. A., and Escalante, H. J. (2013). A hybrid surrogate-based approach for evolutionary multi-objective optimization. In *Evolutionary Computation (CEC), 2013 IEEE Congress on*, pages 2548–2555. 176

[Rosso et al., 2009] Rosso, O. A., Mendes, A., Berretta, R., Rostas, J. A., Hunter, M., and Moscato, P. (2009). Distinguishing childhood absence epilepsy patients from controls by the analysis of their background brain electrical activity (ii): a combinatorial optimization approach for electrode selection. *Journal of neuroscience methods*, 181(2):257–267. 44

[Roy, 2002] Roy, S. (2002). Nearest neighbor with generalization. 121

[Ruta and Gabrys, 2005] Ruta, D. and Gabrys, B. (2005). Classifier selection for majority voting. *Information fusion*, 6(1):63–81. 14

[Rutkowski and Cpalka, 2003] Rutkowski, L. and Cpalka, K. (2003). Flexible neuro-fuzzy systems. *IEEE Transactions on Neural Networks*, 14(3):554–574. 67

[Santana et al., 2006] Santana, A., Soares, R., Canuto, A., and Souto, M. C. P. d. (2006). A dynamic classifier selection method to build ensembles using accuracy and diversity. In *Neural Networks, 2006. SBRN '06. Ninth Brazilian Symposium on*, pages 36–41. 129

[Shah and Kusiak, 2007] Shah, S. and Kusiak, A. (2007). Cancer gene search with data-mining and genetic algorithms. *Computers in Biology and Medicine*, 37(2):251 – 261. 34

[Shalev-Shwartz et al., 2011] Shalev-Shwartz, S., Singer, Y., Srebro, N., and Cotter, A. (2011). Pegasos: Primal estimated sub-gradient solver for svm. *Mathematical programming*, 127(1):3–30. 121

[Shen and Chou, 2006] Shen, H.-B. and Chou, K.-C. (2006). Ensemble classifier for protein fold pattern recognition. *Bioinformatics*, 22(14):1717–1722. 14, 35

[Shen and Chou, 2007] Shen, H.-B. and Chou, K.-C. (2007). Using ensemble classifier to identify membrane protein types. *Amino Acids*, 32(4):483–488. 35

[Shi, 2007] Shi, H. (2007). Best-first decision tree learning. Master's thesis, University of Waikato, Hamilton, NZ. COMP594. 121

[Shi et al., 2010] Shi, L., Campbell, G., Jones, W., Campagne, F., Walker, S., Su, Z., et al. (2010). The MAQC-II Project: A comprehensive study of common practices for the development and validation of microarray-based predictive models. *Nature biotechnology*. 9

[Sikdar et al., 2015] Sikdar, U., Ekbal, A., and Saha, S. (2015). MODE: multiobjective differential evolution for feature selection and classifier ensemble. *Soft Computing*, pages 1–21. 117, 118

[Sim et al., 2012] Sim, K., Hart, E., and Paechter, B. (2012). A hyper-heuristic classifier for one dimensional bin packing problems: Improving classification accuracy by attribute evolution. In *International Conference on Parallel Problem Solving from Nature*, pages 348–357. Springer. 31

[Soufan et al., 2015] Soufan, O., Kleftogiannis, D., Kalnis, P., and Bajic, V. B. (2015). DWFS: A Wrapper Feature Selection Tool Based on a Parallel Genetic Algorithm. *PLoS ONE*, 10(2):e0117988. 151

[Srimani and Koti, 2013] Srimani, P. K. and Koti, M. S. (2013). Medical Diagnosis Using Ensemble Classifiers - A Novel Machine-Learning Approach. *Journal of Advanced Computing*, pages 9–27. 24

[Srinivas and Deb, 1994] Srinivas, N. and Deb, K. (1994). Muiltiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary computation*, 2(3):221–248. 131

[Stehman, 1997] Stehman, S. V. (1997). Selecting and interpreting measures of thematic classification accuracy. *Remote sensing of Environment*, 62(1):77–89. 7

[Storn and Price, 1997] Storn, R. and Price, K. (1997). Differential Evolution-A Simple and Efficient Heuristic for global Optimization over Continuous Spaces. *Journal of Global Optimization*, 11(4):341–359. 87, 89, 90

[Sumner et al., 2005a] Sumner, M., Frank, E., and Hall, M. (2005a). Speeding up logistic model tree induction. In *European Conference on Principles of Data Mining and Knowledge Discovery*, pages 675–683. Springer. 41, 121

[Sumner et al., 2005b] Sumner, M., Frank, E., and Hall, M. (2005b). Speeding up logistic model tree induction. In *9th European Conference on Principles and Practice of Knowledge Discovery in Databases*, pages 675–683. Springer. 121

[Sun et al., 2005] Sun, Y., Kamel, M. S., and Wong, A. K. (2005). Empirical Study on Weighted Voting Multiple Classifiers. In Singh, S., Singh, M., Apte, C., and Perner, P., editors, *Pattern Recognition and Data Mining*, volume 3686 of *Lecture Notes in Computer Science*, pages 335–344. Springer Berlin Heidelberg. 36, 87

[Sun et al., 2009] Sun, Y., Wong, A. K. C., and Kamel, M. S. (2009). Classification of Imbalanced Data: A Review. *International Journal of Pattern Recognition and Artificial Intelligence*, 23(04):687–719. 32

[Thammasiri and Meesad, 2012] Thammasiri, D. and Meesad, P. (2012). Ensemble Data Classification based on Diversity of Classifiers Optimized by Genetic Algorithm. *Advanced Materials Research*, 433-440:6572–6578. 27, 29, 30

[Tsoumakas et al., 2004] Tsoumakas, G., Katakis, I., and Vlahavas, I. (2004). Effective voting of heterogeneous classifiers. In *Machine Learning: ECML 2004*, pages 465–476. Springer. 13

[Tsymbal et al., 2003] Tsymbal, A., Puuronen, S., and Patterson, D. W. (2003). Ensemble feature selection with the simple bayesian classification. *Information Fusion*, 4(2):87 – 100. 23

[Tulyakov et al., 2008] Tulyakov, S., Jaeger, S., Govindaraju, V., and Doermann, D. (2008). Review of classifier combination methods. In Marinai, S. and Fujisawa, H., editors, *Machine Learning in Document Analysis and Recognition*, volume 90 of *Studies in Computational Intelligence*, pages 361–386. Springer Berlin Heidelberg. 23

[Turhal et al., 2013] Turhal, U., Babur, S., Avci, C., and Akbas, A. (2013). Performance improvement for diagnosis of colon cancer by using ensemble classification methods. In *Technological Advances in Electrical, Electronics and Computer Engineering (TAEECE), 2013 International Conference on*, pages 271–275. 143

[Vafaee, 2016] Vafaee, F. (2016). Using multi-objective optimization to identify dynamical network biomarkers as early-warning signals of complex diseases. *Scientific Reports*, 6:22023. 131

[Valdovinos and Sánchez, 2009] Valdovinos, R. and Sánchez, J. (2009). Combining Multiple Classifiers with Dynamic Weighted Voting. In Corchado, E., Wu, X., Oja, E., Herrero, A., and Baruque, B., editors, *Hybrid Artificial Intelligence Systems*, volume 5572 of *Lecture Notes in Computer Science*, pages 510–516. Springer Berlin Heidelberg. 88

[Valentini and Masulli, 2002] Valentini, G. and Masulli, F. (2002). Ensembles of learning machines. In Marinaro, M. and Tagliaferri, R., editors, *Neural Nets*, volume 2486 of *Lecture Notes in Computer Science*, pages 3–20. Springer Berlin Heidelberg. 22

[Vapnik, 1999] Vapnik, V. N. (1999). *The Nature of Statistical Learning Theory*. Springer, 2nd edition. 7

[W. et al., 2011] W., D., R., A., and K., G. (2011). Computational intelligence laboratory. 67, 68

[Wang and Wang, 2006] Wang, X. and Wang, H. (2006). Classification by evolutionary ensembles. *Pattern Recognition*, 39(4):595 – 607. 25

[Wang et al., 2008] Wang, Y., Miller, D. J., and Clarke, R. (2008). Approaches to working in high-dimensional data spaces: gene expression microarrays. *British journal of cancer*, 98(6):1023–8. 4

[Wolpert, 1996] Wolpert, D. H. (1996). The lack of a priori distinctions between learning algorithms. *Neural computation*, 8(7):1341–1390. 10

[Woods et al., 1997] Woods, K., Kegelmeyer, Jr., W. P., and Bowyer, K. (1997). Combination of multiple classifiers using local accuracy estimates. *IEEE Trans. Pattern Anal. Mach. Intell.*, 19(4):405–410. 14

[Wozniak, 2008] Wozniak, M. (2008). Classifier fusion based on weighted voting- analytical and experimental results. In *Intelligent Systems Design and Applications, 2008. ISDA '08. Eighth International Conference on*, volume 2, pages 687–692. IEEE. 88

[Woźniak et al., 2013] Woźniak, M., Graña, M., and Corchado, E. (2013). A survey of multiple classifier systems as hybrid systems. *Information Fusion*. 33

[Wozniak and Jackowski, 2009] Wozniak, M. and Jackowski, K. (2009). Some Remarks on Chosen Methods of Classifier Fusion Based on Weighted Voting. In Corchado, E., Wu, X., Oja, E., Herrero, A., and Baruque, B., editors, *Hybrid Artificial Intelligence Systems*, volume 5572 of *Lecture Notes in Computer Science*, pages 541–548. Springer Berlin Heidelberg. 87

[Xu and He, 2008] Xu, R. and He, L. (2008). GACEM: Genetic Algorithm Based Classifier Ensemble in a Multi-sensor System. *Sensors*, 8(10):6203–6224. 27, 29, 30

[Yang and Wang, 2013] Yang, L. and Wang, L. (2013). A class of semi-supervised support vector machines by DC programming. *Advances in Data Analysis and Classification*, pages 1–17. 67, 68

[Yang et al., 2010a] Yang, P., Hwa Yang, Y., B Zhou, B., and Y Zomaya, A. (2010a). A review of ensemble methods in bioinformatics. *Current Bioinformatics*, 5(4):296–308. 34

[Yang et al., 2013] Yang, P., Liu, W., Zhou, B., Chawla, S., and Zomaya, A. (2013). Ensemble-based wrapper methods for feature selection and class imbalance learning. In Pei, J., Tseng, V., Cao, L., Motoda, H., and Xu, G., editors, *Advances in Knowledge Discovery and Data Mining*, volume 7818 of *Lecture Notes in Computer Science*, pages 544–555. Springer Berlin Heidelberg. 144

[Yang et al., 2010b] Yang, P., Liu, W., Zhou, B. B., Chawla, S., and Albert, Y. (2010b). Ensemble-based wrapper methods for feature selection and class imbalance learning. In *Advances in Knowledge Discovery and Data Mining*, pages 544–555. Springer. 143

[Zhang et al., 2015] Zhang, L., Wang, X., and Moon, W. M. (2015). PolSAR images classification through GA-based selective ensemble learning. In *Geoscience and Remote Sensing Symposium (IGARSS), 2015 IEEE International*, pages 3770–3773. 27

[Zhang and Zhou, 2011] Zhang, L. and Zhou, W.-D. (2011). Sparse ensembles using weighted combination methods based on linear programming. *Pattern Recognition*, 44(1):97–106. 88

[Zhang and Bhattacharyya, 2004] Zhang, Y. and Bhattacharyya, S. (2004). Genetic programming in classifying large-scale data: an ensemble method. *Information Sciences*, 163(1–3):85–101. 25

[Zhang et al., 2014] Zhang, Y., Zhang, H., Cai, J., and Yang, B. (2014). A Weighted Voting Classifier Based on Differential Evolution. *Abstract and Applied Analysis*, 2014. 88, 89

[Zhang et al., 2010] Zhang, Z., Li, J., Hu, H., and Zhou, H. (2010). A robust ensemble classification method analysis. In Arabnia, H. R., editor, *Advances in Computational Biology*, volume 680 of *Advances in Experimental Medicine and Biology*, pages 149–155. Springer New York. 24

[Zhou, 2012] Zhou, Z.-H. (2012). *Ensemble Methods: Foundations and Algorithms*. Chapman & Hall/CRC Machine Learning & Pattern Recognition. Chapman and Hall/CRC. 5

[Zhou and Jiang, 2004] Zhou, Z.-H. and Jiang, Y. (2004). NeC4.5: neural ensemble based C4.5. *Knowledge and Data Engineering, IEEE Transactions on*, 16(6):770–773. 67

[Zhou and Li, 2010] Zhou, Z.-H. and Li, N. (2010). Multi-information ensemble diversity. In *Multiple Classifier Systems: 9th International Workshop, MCS 2010, Cairo, Egypt, April 7-9, 2010. Proceedings*, pages 134–144. Springer Berlin Heidelberg, Berlin, Heidelberg. 129

[Zhou et al., 2002] Zhou, Z.-H., Wu, J., and Tang, W. (2002). Ensembling neural networks: Many could be better than all. *Artificial Intelligence*, 137(12):239 – 263. 116

[Zickenrott et al., 2016] Zickenrott, S., Angarica, V., Upadhyaya, B., and Del Sol, A. (2016). Prediction of disease–gene–drug relationships following a differential network analysis. *Cell Death & Disease*, 7(1):e2040. 131

[Zielinski et al., 2005] Zielinski, K., Peters, D., and Laur, R. (2005). Run time analysis regarding stopping criteria for differential evolution and particle swarm optimization. In *Proc. of the 1st International Conference on Experiments/Process/System Modelling/Simulation/Optimization*. 96

A

Readme File for GA-EoC

How to Use the Software

The GA-EoC: Genetic Algorithm-based Search Method for Heterogeneous Ensemble Classifiers has been implemented using Java programming Language. The source code is available at https://sourceforge.net/projects/geneticensembleclassifier/ under GNU General Public License version 3.0 (GPLv3). The first beta release of the system uses pre-built CV datasets and their models for finding the best base classifiers combination to create the ensemble based on training dataset.

Required Libraries:

To compile and execute the program, it requires some software libraries. They are:

- **WEKA:** We use Waikato Environment for Knowledge Analysis (WEKA) version 3.7.10. The jar file for required version has been included in the 'lib' directory. Alternatively, you can download it from publisher's website at http://www.cs.waikato.ac.nz/ml/weka/index.html
- **libsvm for WEKA:** For using the LibSVM classifier with weka, we have used Wrapper class for the libsvm library by Chih-Chung Chang and Chih-Jen Lin (http://www.csie.ntu.edu.tw/~cjlin/libsvm/). The original wrapper, named WLSVM, was developed by Yasser EL-Manzalawy (Yasser EL-Manzalawy (2005). WLSVM. URL http://www.cs.iastate.edu/~yasser/wlsvm/). We have included the required wrappers for using the LibSVM inside the 'lib' directory.
- **Java:** we used JDK 1.6 for the development of the souce code. If you run the program in a multi-core computer, it will leave 2 processor cores free and use rest cores for this program.

Features:

This beta release of the system consists of following features:

- Generate Cross Validation folds and save datasets into disk for future usage in ARFF format.
- Generate and serialise Classifier Models into the disk for all cross validation Training Folds for use by GA-EoC.
- Generate and serialise Base Classifiers Model into disk using the Full Training dataset.
- Search for best ensemble combinations to create heterogeneous ensemble of classifiers using k-fold cross validation on training dataset (using pre-generated CV dataset and models).
- Evaluate the performance of best ensemble combination on unknown Testing Data (use pre-generated models using full training data).

How to run the executable:

To execute the program it is required to download and unzip the compressed program from companion website. It is required to:

1. Unzip the GA-EoC.zip file into a directory.
2. To execute or run the program from the command line, go to the 'lib' folder and type the following command depending on your intended usage:

   - **Usage 1:** Generate CV Models for use in GA-EoC.

     ```
     java -jar GA-EoC.jar <options optValue>
     -t = input trainFileName ../path/name.
     -b = enable flag Build CV Models (no values required).
     -d = output CV Data path.
     -m = output Prebuilt CV Models path.
     -f = CV folds (optional, default 10-fold cv).
     ```

   - **Usage 2:** Generate Full Models for use in GA-EoC.

     ```
     java -jar GA-EoC.jar <options optValue>
     -t = input trainFileName ../path/name.
     -M = output Path for Full Models.
     ```

   - **Usage 3:** Use Prebuilt Models to find Best Ensemble Combination.

```
java -jar GA-EoC.jar <options optValue>
-t = input trainFileName ../path/name.
-d = input CV Data path.
-m = input Prebuilt Model path.
-r = Repeat the Process (optional, default 50 repeatations).
-l = output Run Logs.
-f = CV folds (optional, default 10-fold cv).
```

- **Usage 4:** Evaluate the performance of an Ensemble Combination.

```
java -jar GA-EoC.jar <options optValue>
-t = input trainFileName ../path/name.
-T = input TestFileName ../path/name.
-M = input Path for Full Models.
-e = input Ensemble Combination.
```

However, this program does not guarantee to be free from bugs. Bug reporting will highly appreciated for future development and stability of the program through the website.

B

Readme File for DE-HEoC

We propose the use of Differential Evolution algorithm for the weight adjustment of base classifiers used in weighted voting heterogeneous ensemble of classifier. Average Matthews Correlation Coefficient (MCC) score, calculated over 10-fold cross-validation, has been used as the measure of quality of an ensemble. `DE/rand/1/bin` algorithm has been utilised to maximise the average MCC score calculated using 10-fold cross-validation on training dataset. The voting weights of base classifiers are optimised for the heterogeneous ensemble of classifiers aiming to attain better generalisation performances on testing datasets.

How to run the executable:

To execute the program it is required to download and unzip the compressed program from companion website https://sourceforge.net/projects/de-heoc/. It is required to:

1. Unzip the GA-EoC.zip file into a directory.
2. To execute or run the program from the command line, go to the 'lib' folder and redirect to settings file with `-i <settingsFileName>` keys for your intended usage. An example of settings file for execution of `DE-HEoC` for execution and evaluation is shown here:

- Eexcute DE-EoC to find the Best Individual:

```
expName:
trnFile:
cvDataPath:
cvModelPath:
```

```
#fold: default=10
#maxGen: default=1000
#repeat: default=30
```

- Evaluate the Performance of Best Individuals:

```
expName:
modelPath:
trnFile:
tstFile:
eval:
#repeat: default=30
```

C

Permissions for Copyrighted Materials

This chapter in the appendix includes the first page of each acquired licenses of the figures and excerpt of papers used in the thesis.

Permission Number 3711660699232

JOHN WILEY AND SONS LICENSE TERMS AND CONDITIONS

Feb 21, 2016

This Agreement between Mohammad N Haque ("You") and John Wiley and Sons ("John Wiley and Sons") consists of your license details and the terms and conditions provided by John Wiley and Sons and Copyright Clearance Center.

License Number	3711660699232
License date	Sep 17, 2015
Licensed Content Publisher	John Wiley and Sons
Licensed Content Publication	Wiley oBooks
Licensed Content Title	Fundamentals of Pattern Recognition
Licensed Content Author	Ludmila I. Kuncheva
Licensed Content Date	Aug 29, 2014
Pages	48
Type of Use	Dissertation/Thesis
Requestor type	University/Academic
Format	Print and electronic
Portion	Figure/table
Number of figures/tables	1
Original Wiley figure/table number(s)	Fig 1.1 The pattern recognition cycle.
Will you be translating?	No
Title of your thesis / dissertation	Genetic Algorithm-Based Ensemble Method for Large-Scale Biological Data Classification
Expected completion date	Feb 2016
Expected size (number of pages)	200
Requestor Location	Mohammad N Haque CIBM The University of Newcastle University Drive Callaghan, Australia 2308 Attn: Mohammad N Haque
Billing Type	Invoice
Billing Address	Mohammad N Haque CIBM The University of Newcastle University Drive Callaghan, Australia 2308 Attn: Mohammad N Haque
Total	**0.00 AUD**
Terms and Conditions	

TERMS AND CONDITIONS

This copyrighted material is owned by or exclusively licensed to John Wiley & Sons, Inc. or one of its group companies (each a"Wiley Company") or handled on behalf of a society with which a Wiley Company has exclusive publishing rights in relation to a particular work (collectively "WILEY"). By clicking ‘accept’ in connection with

Permission Number 3711660886137

JOHN WILEY AND SONS LICENSE TERMS AND CONDITIONS

Feb 21, 2016

This Agreement between Mohammad N Haque ("You") and John Wiley and Sons ("John Wiley and Sons") consists of your license details and the terms and conditions provided by John Wiley and Sons and Copyright Clearance Center.

License Number	3711660886137
License date	Sep 17, 2015
Licensed Content Publisher	John Wiley and Sons
Licensed Content Publication	Wiley oBooks
Licensed Content Title	Fundamentals of Pattern Recognition
Licensed Content Author	Ludmila I. Kuncheva
Licensed Content Date	Aug 29, 2014
Pages	48
Type of Use	Dissertation/Thesis
Requestor type	University/Academic
Format	Print and electronic
Portion	Text extract
Number of Pages	1
Will you be translating?	No
Title of your thesis / dissertation	Genetic Algorithm-Based Ensemble Method for Large-Scale Biological Data Classification
Expected completion date	Feb 2016
Expected size (number of pages)	200
Requestor Location	Mohammad N Haque CIBM The University of Newcastle University Drive Callaghan, Australia 2308 Attn: Mohammad N Haque
Billing Type	Invoice
Billing Address	Mohammad N Haque CIBM The University of Newcastle University Drive Callaghan, Australia 2308 Attn: Mohammad N Haque
Total	**0.00 AUD**
Terms and Conditions	

TERMS AND CONDITIONS

This copyrighted material is owned by or exclusively licensed to John Wiley & Sons, Inc. or one of its group companies (each a"Wiley Company") or handled on behalf of a society with which a Wiley Company has exclusive publishing rights in relation to a particular work (collectively "WILEY"). By clicking "accept" in connection with completing this licensing transaction, you agree that the following terms and conditions apply to this transaction (along with the billing and payment terms and conditions established by the Copyright Clearance Center Inc., ("CCC's Billing and Payment terms and conditions"), at the time that you opened your Rightslink account (these are available at any

Permission Number 3806210031731

ELSEVIER LICENSE TERMS AND CONDITIONS

Feb 21, 2016

This is an Agreement between Mohammad N Haque ("You") and Elsevier ("Elsevier"). It consists of your order details, the terms and conditions provided by Elsevier, and the payment terms and conditions.

All payments must be made in full to CCC. For payment instructions, please see information listed at the bottom of this form.

Supplier	Elsevier Limited The Boulevard,Langford Lane Kidlington,Oxford,OX5 1GB,UK
Registered Company Number	1982084
Customer name	Mohammad N Haque
Customer address	CIBM Callaghan, NSW 2308
License number	3806210031731
License date	Feb 11, 2016
Licensed content publisher	Elsevier
Licensed content publication	Artificial Intelligence
Licensed content title	Wrappers for feature subset selection
Licensed content author	Ron Kohavi,George H. John
Licensed content date	December 1997
Licensed content volume number	97
Licensed content issue number	1-2
Number of pages	52
Start Page	273
End Page	324
Type of Use	reuse in a thesis/dissertation
Portion	figures/tables/illustrations
Number of figures/tables /illustrations	1
Format	both print and electronic
Are you the author of this Elsevier article?	No
Will you be translating?	No
Original figure numbers	Fig 1.
Title of your thesis/dissertation	Genetic Algorithm-Based Ensemble Method for Large-Scale Biological Data Classification
Expected completion date	Feb 2016
Estimated size (number of pages)	200
Elsevier VAT number	GB 494 6272 12
Price	0.00 AUD
VAT/Local Sales Tax	0.00 AUD / 0.00 GBP

Permission Number 3813910409023

ELSEVIER LICENSE TERMS AND CONDITIONS

Feb 21, 2016

This is an Agreement between Mohammad N Haque ("You") and Elsevier ("Elsevier"). It consists of your order details, the terms and conditions provided by Elsevier, and the payment terms and conditions.

All payments must be made in full to CCC. For payment instructions, please see information listed at the bottom of this form.

Supplier	Elsevier Limited The Boulevard,Langford Lane Kidlington,Oxford,OX5 1GB,UK
Registered Company Number	1982084
Customer name	Mohammad N Haque
Customer address	CIBM Callaghan, NSW 2308
License number	3813910409023
License date	Feb 11, 2016
Licensed content publisher	Elsevier
Licensed content publication	Elsevier Books
Licensed content title	Machine Learning Proceedings 1994
Licensed content author	George H. John,Ron Kohavi,Karl Pfleger
Licensed content date	1994
Number of pages	9
Start Page	121
End Page	129
Type of Use	reuse in a thesis/dissertation
Intended publisher of new work	other
Portion	figures/tables/illustrations
Number of figures/tables /illustrations	1
Format	both print and electronic
Are you the author of this Elsevier chapter?	No
Will you be translating?	No
Original figure numbers	Figure 4
Title of your thesis/dissertation	Genetic Algorithm-Based Ensemble Method for Large-Scale Biological Data Classification
Expected completion date	Feb 2016
Estimated size (number of pages)	
Elsevier VAT number	GB 494 6272 12
Price	0.00 AUD
VAT/Local Sales Tax	0.00 AUD / 0.00 GBP
Total	**0.00 AUD**

Written Permission for Using the Figure 1.1

Re: Regarding Permission to use Figure in Thesis

Subject: Re: Regarding Permission to use Figure in Thesis
From: "M. Madan Babu | LMB" <madanm@mrc-lmb.cam.ac.uk>
Date: 24/09/15 01:35
To: Mohammad Nazmul Haque <c3172331@uon.edu.au>

Dear Mohammad
I am happy for you to use the figure in your thesis.
All the best,
Madan

On 21/09/2015 08:51, Mohammad Nazmul Haque wrote:

> Dear Prof. M. Madan Babu,
>
> I am a PhD Student at the University of Newcastle, Australia. I am working on biological data classification technique. I need to use the figure-1 appeared in the chapter ***Babu, M. M. (2004). An Introduction to Microarray Data Analysis. In R. P. Grant (Ed.), Computational Genomics: Theory and Application (pp. 225–249). Horizon Press, U.K***.
>
> According to the publisher's copyright policy I have to follow their guideline as:
> "Caister Academic Press also permits anyone to reproduce figures and tables from our books and journals provided (1) Caister Academic Press owns the copyright (2) permission is also obtained from the original authors and (3) full reference is made to the original source."
>
> Could you please provide me the permission to adapt that figures for inclusion in my thesis chapter?
>
> Thank you very much sir.
>
> With Regards,
> **Mohammad Nazmul Haque**
> *PhD. Candidate (Computer Science)*
> Centre for Bioinformatics, Biomarker Discovery & Information-Based Medicine (CIBM)
> School of Electrical Engineering and Computer Science
> **The University of Newcastle**
> Hunter Medical Research Institute (HMRI)
> Kookaburra Circuit 1, New Lambton Heights, NSW--2305, AUSTRALIA
> M: +61 405 118 986, W: +61 2 4042 0490

--

M. Madan Babu, PhD

Programme Leader, MRC Laboratory of Molecular Biology, Cambridge, UK
Director of Studies, Trinity College, Cambridge, UK

Executive Editor, Nucleic Acids Research, UK
Associate Editor, Molecular BioSystems, UK

Group page: http://mbgroup.mrc-lmb.cam.ac.uk/
Email: madanm@mrc-lmb.cam.ac.uk

Permission Number 03832290290836

SPRINGER LICENSE
TERMS AND CONDITIONS

Mar 19, 2016

This is a License Agreement between Mohammad N Haque ("You") and Springer ("Springer") provided by Copyright Clearance Center ("CCC"). The license consists of your order details, the terms and conditions provided by Springer, and the payment terms and conditions.

All payments must be made in full to CCC. For payment instructions, please see information listed at the bottom of this form.

License Number	3832290290836
License date	Mar 19, 2016
Licensed content publisher	Springer
Licensed content publication	Springer eBook
Licensed content title	Multi-objective Optimization
Licensed content author	None
Licensed content date	Jan 1, 2008
Type of Use	Thesis/Dissertation
Portion	Figures/tables/illustrations
Number of figures/tables/illustrations	1
Author of this Springer article	No
Order reference number	None
Original figure numbers	Fig. 2.1 Example of a Pareto curve
Title of your thesis / dissertation	Genetic Algorithm-Based Ensemble Method for Large-Scale Biological Data Classification
Expected completion date	Feb 2016
Estimated size(pages)	200
Total	0.00 USD
Terms and Conditions	

Introduction
The publisher for this copyrighted material is Springer. By clicking "accept" in connection with completing this licensing transaction, you agree that the following terms and conditions apply to this transaction (along with the Billing and Payment terms and conditions established by Copyright Clearance Center, Inc. ("CCC"), at the time that you opened your Rightslink account and that are available at any time at http://myaccount.copyright.com).
Limited License
With reference to your request to reuse material on which Springer controls the copyright, permission is granted for the use indicated in your enquiry under the following conditions:

Permission Number 3832321387782

Permission Number 3954000980935

JOHN WILEY AND SONS LICENSE TERMS AND CONDITIONS

Sep 22, 2016

This Agreement between Mohammad N Haque ("You") and John Wiley and Sons ("John Wiley and Sons") consists of your license details and the terms and conditions provided by John Wiley and Sons and Copyright Clearance Center.

License Number	3954000980935
License date	Sep 22, 2016
Licensed Content Publisher	John Wiley and Sons
Licensed Content Publication	Wiley Books
Licensed Content Title	Combining Pattern Classifiers: Methods and Algorithms, 2nd Edition
Licensed Content Author	Ludmila I. Kuncheva
Licensed Content Date	Aug 1, 2014
Licensed Content Pages	384
Type of use	Dissertation/Thesis
Requestor type	University/Academic
Format	Print and electronic
Portion	Figure/table
Number of figures/tables	1
Original Wiley figure/table number(s)	FIGURE 1.12 Bias and Variance
Will you be translating?	No
Title of your thesis / dissertation	Genetic Algorithm-based Ensemble Methods for Large-Scale Biological Data Classification
Expected completion date	Oct 2016
Expected size (number of pages)	250
Requestor Location	Mohammad N Haque CIBM The University of Newcastle University Drive Callaghan, NSW 2308 Australia Attn: Mohammad N Haque
Publisher Tax ID	EU826007151
Billing Type	Invoice
Billing Address	Mohammad N Haque CIBM The University of Newcastle University Drive Callaghan, Australia 2308 Attn: Mohammad N Haque

Permission Number 3955870396941

D

List of selected Probes from METABRIC dataset

List of selected 444 probes by the MO-EoC-WFS (using CM_2 feature ranking method) for METABRIC discovery dataset is given below:

ILMN_1738401	ILMN_1769849	ILMN_1788166	ILMN_2143566
ILMN_1689146	ILMN_1743620	ILMN_1693014	ILMN_1766650
ILMN_1807423	ILMN_1693218	ILMN_1653750	ILMN_1775235
ILMN_1668766	ILMN_1713813	ILMN_2189675	ILMN_2382942
ILMN_1692938	ILMN_1731184	ILMN_1684217	ILMN_1779416
ILMN_1813270	ILMN_1718731	ILMN_2326273	ILMN_1703891
ILMN_1721354	ILMN_1805737	ILMN_1720373	ILMN_1651329
ILMN_2344120	ILMN_1747016	ILMN_2336781	ILMN_1803073
ILMN_1737184	ILMN_1685403	ILMN_1679809	ILMN_1888359
ILMN_1719753	ILMN_1750748	ILMN_2353161	ILMN_1720998
ILMN_2381257	ILMN_1666845	ILMN_1659895	ILMN_2406656
ILMN_1786720	ILMN_1725276	ILMN_1665538	ILMN_1811014
ILMN_1663119	ILMN_2192072	ILMN_1652631	ILMN_2066088
ILMN_1781758	ILMN_1772521	ILMN_1704537	ILMN_1745501
ILMN_1752899	ILMN_2170814	ILMN_1663390	ILMN_2310814
ILMN_1800091	ILMN_2255133	ILMN_1678535	ILMN_1826165
ILMN_1736760	ILMN_1680390	ILMN_1688071	ILMN_1685174
ILMN_2353054	ILMN_1746846	ILMN_1835913	ILMN_2311020
ILMN_1664855	ILMN_1704294	ILMN_1722489	ILMN_1795342
ILMN_1660654	ILMN_2149164	ILMN_1716925	ILMN_1755741
ILMN_1780255	ILMN_1671554	ILMN_1753766	ILMN_2334359
ILMN_1654319	ILMN_1729216	ILMN_1728787	ILMN_2269256
ILMN_2212909	ILMN_2228162	ILMN_1695397	ILMN_1796059

ILMN_1725678	ILMN_2163723	ILMN_1661443	ILMN_1699665
ILMN_2050246	ILMN_1869390	ILMN_2231299	ILMN_2081883
ILMN_1813607	ILMN_1740762	ILMN_1785570	ILMN_1809433
ILMN_1811387	ILMN_1692731	ILMN_1809639	ILMN_2092077
ILMN_1739233	ILMN_2368188	ILMN_1746359	ILMN_1806456
ILMN_1814151	ILMN_1750974	ILMN_1844029	ILMN_2353862
ILMN_1901921	ILMN_2317923	ILMN_1849013	ILMN_1772686
ILMN_1750394	ILMN_1746517	ILMN_1730612	ILMN_2183692
ILMN_2217601	ILMN_1737514	ILMN_2120555	ILMN_2113490
ILMN_2145396	ILMN_1751753	ILMN_1790350	ILMN_1670099
ILMN_1743055	ILMN_1653358	ILMN_2246956	ILMN_1753139
ILMN_1668619	ILMN_1758404	ILMN_1801119	ILMN_1686562
ILMN_2161330	ILMN_2347949	ILMN_1709132	ILMN_1701918
ILMN_1768772	ILMN_1693270	ILMN_1694535	ILMN_2088437
ILMN_2215639	ILMN_1781360	ILMN_1907649	ILMN_1679632
ILMN_1717052	ILMN_1748751	ILMN_1654385	ILMN_2365465
ILMN_1898518	ILMN_1745005	ILMN_2387952	ILMN_1676990
ILMN_2329569	ILMN_1697559	ILMN_1788874	ILMN_1730977
ILMN_1767129	ILMN_1710427	ILMN_2283597	ILMN_1652826
ILMN_2191192	ILMN_2289844	ILMN_1811363	ILMN_1719250
ILMN_2125763	ILMN_1775285	ILMN_1731237	ILMN_2087692
ILMN_1661078	ILMN_1727078	ILMN_1711208	ILMN_1697460
ILMN_1720604	ILMN_1700268	ILMN_1708341	ILMN_2216637
ILMN_1794595	ILMN_1657409	ILMN_1691884	ILMN_2344971
ILMN_1655610	ILMN_1805636	ILMN_1811330	ILMN_2413898
ILMN_1683576	ILMN_2060413	ILMN_1673941	ILMN_1786125
ILMN_1723709	ILMN_1763540	ILMN_1742881	ILMN_1737728
ILMN_2281786	ILMN_1763127	ILMN_1784783	ILMN_1673721
ILMN_1772588	ILMN_2101526	ILMN_1801313	ILMN_1683450
ILMN_1708402	ILMN_1802780	ILMN_1659312	ILMN_1714730
ILMN_1810978	ILMN_2395451	ILMN_1661895	ILMN_1720526
ILMN_2161820	ILMN_1782389	ILMN_2363621	ILMN_1801257
ILMN_1729801	ILMN_1667018	ILMN_1759910	ILMN_2374425
ILMN_2352131	ILMN_1796337	ILMN_1718198	ILMN_2136495
ILMN_1803236	ILMN_1651776	ILMN_1744023	ILMN_1685916
ILMN_1773459	ILMN_1730917	ILMN_1791095	ILMN_1670238
ILMN_1674533	ILMN_1753340	ILMN_1718046	ILMN_2202948
ILMN_1713952	ILMN_1730355	ILMN_1769219	ILMN_2301083
ILMN_2410713	ILMN_1657095	ILMN_1742073	ILMN_1796949
ILMN_1749118	ILMN_1697338	ILMN_1770085	ILMN_1801939
ILMN_2405254	ILMN_1787815	ILMN_1805104	ILMN_1751444
ILMN_1728898	ILMN_1737255	ILMN_1708983	ILMN_1796589
ILMN_1770678	ILMN_1775587	ILMN_1698885	ILMN_1777233
ILMN_1661466	ILMN_2128770	ILMN_1711894	ILMN_1716400

ILMN_1651237	ILMN_2357438	ILMN_1767556	ILMN_1728197
ILMN_1703906	ILMN_1673673	ILMN_1654072	ILMN_1778523
ILMN_1781943	ILMN_1712803	ILMN_1720158	ILMN_1680424
ILMN_1753196	ILMN_1654268	ILMN_1801043	ILMN_1691860
ILMN_1815184	ILMN_1725260	ILMN_1658356	ILMN_1678215
ILMN_2049021	ILMN_1711470	ILMN_1660086	ILMN_1732296
ILMN_1749829	ILMN_2368721	ILMN_1752932	ILMN_2116299
ILMN_1666305	ILMN_1799667	ILMN_1726204	ILMN_1657145
ILMN_2042771	ILMN_2412384	ILMN_2304512	ILMN_2120695
ILMN_1751776	ILMN_1782045	ILMN_1799098	ILMN_1775170
ILMN_1731070	ILMN_2051373	ILMN_2193325	ILMN_1672503
ILMN_1739645	ILMN_1680955	ILMN_1737988	ILMN_1772910
ILMN_1747911	ILMN_1737195	ILMN_1809099	ILMN_1780170
ILMN_1795852	ILMN_1806040	ILMN_1709486	ILMN_2131861
ILMN_1763907	ILMN_1709634	ILMN_1740609	ILMN_2358760
ILMN_2222008	ILMN_1654151	ILMN_2076600	ILMN_1664516
ILMN_1670353	ILMN_1780667	ILMN_1670305	ILMN_1687235
ILMN_2392472	ILMN_1668814	ILMN_1664464	ILMN_1726108
ILMN_1802819	ILMN_1801632	ILMN_2388800	ILMN_1751016
ILMN_1695658	ILMN_1665035	ILMN_1694840	ILMN_2400500
ILMN_2384785	ILMN_2197128	ILMN_1780799	ILMN_1711124
ILMN_1782403	ILMN_1743445	ILMN_1785071	ILMN_2409220
ILMN_1728934	ILMN_1778087	ILMN_1791447	ILMN_1655915
ILMN_1664630	ILMN_1766914	ILMN_2373791	ILMN_1728496
ILMN_1728972	ILMN_1676822	ILMN_1689111	ILMN_1779711
ILMN_2413650	ILMN_1651282	ILMN_1789733	ILMN_1735822
ILMN_1700337	ILMN_2063168	ILMN_1704753	ILMN_2072296
ILMN_2362549	ILMN_1655611	ILMN_1677092	ILMN_1755721
ILMN_2285996	ILMN_1664176	ILMN_1654398	ILMN_1688322
ILMN_2077550	ILMN_2072568	ILMN_2384241	ILMN_2330861
ILMN_1786065	ILMN_1657766	ILMN_2184184	ILMN_2066756
ILMN_1726720	ILMN_1732158	ILMN_1767448	ILMN_2214355
ILMN_1809590	ILMN_1723684	ILMN_1726245	ILMN_1765770
ILMN_1811472	ILMN_1765363	ILMN_1671928	ILMN_1789507
ILMN_1777564	ILMN_1655468	ILMN_2329914	ILMN_1681503
ILMN_1735093	ILMN_1741356	ILMN_1651610	ILMN_2096322
ILMN_1798108	ILMN_2089752	ILMN_1741021	ILMN_1880983
ILMN_2368718	ILMN_2223941	ILMN_1773389	ILMN_1756326
ILMN_1766658	ILMN_1795325	ILMN_1751607	ILMN_1738093
ILMN_2409298	ILMN_1723481	ILMN_2408683	ILMN_1770053
ILMN_2143155	ILMN_1754103	ILMN_2360415	ILMN_1656452
ILMN_2048700	ILMN_1753101	ILMN_1809291	ILMN_1735762
ILMN_1794539	ILMN_1653934	ILMN_1764309	ILMN_1688022
ILMN_1686097	ILMN_1749868	ILMN_2105441	ILMN_1729533

E
Additional Results

This chapter includes additional experimental results.

Table S1: Classification performances of base classifiers for the WBC dataset.

Classifier	**MCC**	**Accuracy (%)**	**FMeasure**	**Precision**
BayesNet	**0.941**	**97.28**	**0.979**	**0.993**
DecisionStump	0.840	92.42	0.940	0.972
DecisionTable	0.871	94.13	0.955	0.960
IBk	0.895	95.28	0.964	0.957
J48	0.893	95.14	0.963	0.969
JRip	0.893	95.14	0.963	0.967
LibSVM	0.910	95.72	0.966	0.993
LMT	0.911	95.99	0.970	0.965
Logistic	0.924	96.57	0.974	0.974
NaiveBayes	0.914	95.99	0.969	0.986
NaiveBayesUpdateable	0.914	95.99	0.969	0.986
OneR	0.837	92.70	0.946	0.923
PART	0.870	94.13	0.955	0.952
RandomForest	0.912	95.99	0.969	0.971
RandomTree	0.860	93.71	0.952	0.950
REPTree	0.865	93.85	0.953	0.958
SGD	0.927	96.71	0.975	0.978
SimpleLogistic	0.911	95.99	0.970	0.965
VotedPerceptron	0.815	90.99	0.928	0.974
ZeroR	0.000	65.52	0.792	0.655

TABLE S2: Classification performances of base classifiers for the BUPA dataset.

Classifier	**MCC**	**Accuracy (%)**	**FMeasure**	**Precision**
BayesNet	0.043	56.81	0.273	0.467
DecisionStump	0.201	61.74	0.511	0.552
DecisionTable	0.143	59.71	0.437	0.529
IBk	0.241	63.19	0.554	0.564
J48	0.328	67.83	0.581	0.642
JRip	0.325	67.83	0.565	0.655
LibSVM	0.127	59.42	0.079	**0.857**
LMT	**0.407**	**71.59**	0.626	0.701
Logistic	0.352	68.99	0.596	0.658
NaiveBayes	0.149	53.91	0.583	0.470
NaiveBayesUpdateable	0.149	53.91	0.583	0.470
OneR	0.087	55.94	0.457	0.474
PART	0.261	64.06	0.569	0.573
RandomForest	0.355	68.12	**0.638**	0.610
RandomTree	0.243	63.48	0.547	0.571
REPTree	0.277	65.51	0.548	0.610
SGD	0.304	66.96	0.533	0.657
SimpleLogistic	0.356	69.28	0.579	0.682
VotedPerceptron	0.333	67.54	0.446	0.789
ZeroR	0.000	57.97	0.000	0.000

TABLE S3: Classification performances of base classifiers for the PIMA dataset.

Classifier	**MCC**	**Accuracy (%)**	**FMeasure**	**Precision**
BayesNet	0.429	74.35	0.623	0.639
DecisionStump	0.375	71.88	0.588	0.602
DecisionTable	0.377	72.40	0.579	0.619
IBk	0.331	70.18	0.554	0.580
J48	0.417	73.83	0.614	0.632
JRip	0.434	74.61	0.626	0.644
LibSVM	0.000	65.10	0.000	0.000
LMT	0.485	77.47	0.634	0.732
Logistic	0.480	77.21	0.636	0.718
NaiveBayes	0.468	76.30	**0.643**	0.678
NaiveBayesUpdateable	0.468	76.30	**0.643**	0.678
OneR	0.329	70.83	0.531	0.605
PART	0.435	74.48	0.629	0.638
RandomForest	0.434	74.22	0.632	0.630
RandomTree	0.318	69.14	0.554	0.559
REPTree	0.444	75.39	0.623	0.670
SGD	**0.497**	**77.99**	0.641	**0.744**
SimpleLogistic	0.485	77.47	0.634	0.732
VotedPerceptron	0.135	65.36	0.289	0.509
ZeroR	0.000	65.10	0.000	0.000

TABLE S4: Classification performances of base classifiers for the AD dataset using the 5-protein biomarker.

Classifier	**MCC**	**Accuracy (%)**	**FMeasure**	**Precision**
BayesNet	**0.914**	**95.65**	**0.953**	0.932
DecisionStump	0.803	90.22	0.889	0.923
DecisionTable	0.803	90.22	0.889	0.923
IBk	0.786	89.13	0.886	0.848
J48	0.803	90.22	0.894	0.884
JRip	0.848	92.39	0.914	0.949
LibSVM	0.870	93.48	0.930	0.909
LMT	0.893	94.56	0.943	0.911
Logistic	0.893	94.56	0.943	0.911
NaiveBayes	**0.914**	**95.65**	**0.953**	0.932
NaiveBayesUpdateable	**0.914**	**95.65**	**0.953**	0.932
OneR	0.803	90.22	0.889	0.923
PART	0.825	91.30	0.902	0.925
RandomForest	0.890	94.56	0.940	**0.951**
RandomTree	0.690	83.70	0.839	0.765
REPTree	0.803	90.22	0.889	0.923
SGD	0.893	94.56	0.943	0.911
SimpleLogistic	0.893	94.56	0.943	0.911
VotedPerceptron	0.827	91.30	0.900	0.947
ZeroR	0.000	45.65	0.627	0.457

TABLE S5: Classification performances of base classifiers for the MCI dataset using the 5-protein biomarker.

Classifier	**MCC**	**Accuracy (%)**	**FMeasure**	**Precision**
BayesNet	0.282	63.83	0.638	0.600
DecisionStump	0.145	57.45	0.545	0.545
DecisionTable	0.100	55.32	0.512	0.524
IBk	0.157	57.45	0.583	0.538
J48	0.272	63.83	0.605	0.619
JRip	0.100	55.32	0.512	0.524
LibSVM	0.367	68.08	0.681	0.640
LMT	**0.529**	**74.47**	**0.769**	0.667
Logistic	0.512	**74.47**	0.760	**0.679**
NaiveBayes	0.351	65.96	0.692	0.600
NaiveBayesUpdateable	0.351	65.96	0.692	0.600
OneR	0.145	57.45	0.545	0.545
PART	0.183	59.57	0.537	0.579
RandomForest	0.191	59.57	0.578	0.565
RandomTree	0.065	53.19	0.522	0.500
REPTree	0.145	57.45	0.545	0.545
SGD	0.476	72.34	0.745	0.655
SimpleLogistic	**0.529**	**74.47**	**0.769**	0.667
VotedPerceptron	0.226	61.70	0.550	0.611
ZeroR	0.000	46.81	0.638	0.468

TABLE S6: Classification performances of base classifiers for the AD dataset using the 18-protein biomarker.

Classifier	**MCC**	**Accuracy (%)**	**FMeasure**	**Precision**
BayesNet	0.848	89.13	0.886	0.786
DecisionStump	0.923	90.22	0.889	0.803
DecisionTable	0.923	90.22	0.889	0.803
IBk	0.930	**94.56**	**0.941**	**0.891**
J48	**0.951**	**94.56**	0.940	0.890
JRip	0.745	79.35	0.787	0.591
LibSVM	0.889	92.39	0.920	0.849
LMT	0.860	88.04	0.871	0.760
Logistic	0.837	85.87	0.847	0.716
NaiveBayes	0.891	93.48	0.932	0.873
NaiveBayesUpdateable	0.891	93.48	0.932	0.873
OneR	0.923	90.22	0.889	0.803
PART	0.946	90.22	0.886	0.806
RandomForest	0.864	89.13	0.884	0.783
RandomTree	0.791	81.52	0.800	0.628
REPTree	0.923	90.22	0.889	0.803
SGD	0.851	90.22	0.899	0.809
SimpleLogistic	0.860	88.04	0.871	0.760
VotedPerceptron	0.889	92.39	0.920	0.849
ZeroR	0.457	45.65	0.627	0.000

TABLE S7: Classification performances of base classifiers for the MCI dataset using the 18-protein biomarker.

Classifier	**MCC**	**Accuracy (%)**	**FMeasure**	**Precision**
BayesNet	0.581	63.83	0.679	0.314
DecisionStump	0.545	57.45	0.545	0.145
DecisionTable	0.524	55.32	0.512	0.100
IBk	0.594	65.96	0.704	0.368
J48	0.571	59.57	0.558	0.186
JRip	0.594	65.96	0.704	0.368
LibSVM	0.613	68.08	0.717	0.404
LMT	**0.654**	**70.21**	0.708	0.414
Logistic	**0.654**	**70.21**	0.708	0.414
NaiveBayes	0.576	63.83	0.691	0.331
NaiveBayesUpdateable	0.576	63.83	0.691	0.331
OneR	0.545	57.45	0.545	0.145
PART	0.625	65.96	0.652	0.321
RandomForest	0.552	59.57	0.627	0.213
RandomTree	0.500	53.19	0.560	0.078
REPTree	0.545	57.45	0.545	0.145
SGD	0.633	**70.21**	**0.731**	**0.440**
SimpleLogistic	**0.654**	**70.21**	0.708	0.414
VotedPerceptron	0.619	63.83	0.605	0.272
ZeroR	0.468	46.81	0.638	0.000

TABLE S8: Classification performances of base classifiers for the UAB datasets.

Classifier	**MCC**	**Accuracy (%)**	**FMeasure**	**Precision**
BayesNet	0.470	78.00	0.499	0.373
DecisionStump	0.321	80.80	0.273	0.212
DecisionTable	0.425	76.40	0.377	0.243
IBk	0.680	86.80	0.653	0.592
J48	0.420	76.80	0.435	0.293
JRip	0.547	81.20	0.554	0.443
LibSVM	**0.870**	86.80	0.541	0.530
LMT	0.863	**89.20**	**0.686**	0.646
Logistic	0.621	83.60	0.576	0.494
NaiveBayes	0.452	76.80	0.522	0.393
NaiveBayesUpdateable	0.452	76.80	0.522	0.393
OneR	0.344	76.80	0.294	0.166
PART	0.384	76.40	0.358	0.217
RandomForest	0.631	82.80	0.396	0.348
RandomTree	0.393	75.60	0.382	0.234
REPTree	0.468	77.20	0.464	0.328
SGD	0.760	88.00	0.646	0.591
SimpleLogistic	0.863	**89.20**	**0.686**	**0.646**
VotedPerceptron	0.603	84.00	0.585	0.491
ZeroR	0.000	80.00	0.000	0.000

TABLE S9: Classification performances of base classifiers for the IAB datasets.

Classifier	**MCC**	**Accuracy (%)**	**FMeasure**	**Precision**
BayesNet	0.459	77.60	0.493	0.365
DecisionStump	0.321	80.80	0.273	0.212
DecisionTable	0.356	75.20	0.335	0.186
IBk	0.694	87.20	0.661	0.601
J48	0.454	78.40	0.452	0.322
JRip	0.392	74.40	0.437	0.280
LibSVM	**0.870**	86.80	0.541	0.530
LMT	0.821	**88.80**	**0.673**	0.626
Logistic	0.753	86.40	0.577	0.525
NaiveBayes	0.443	76.40	0.517	0.386
NaiveBayesUpdateable	0.443	76.40	0.517	0.386
OneR	0.344	76.80	0.294	0.166
PART	0.360	75.60	0.325	0.181
RandomForest	0.548	82.40	0.406	0.351
RandomTree	0.342	72.80	0.336	0.169
REPTree	0.468	77.20	0.464	0.328
SGD	0.824	**88.80**	0.666	0.624
SimpleLogistic	0.821	**88.80**	**0.673**	**0.626**
VotedPerceptron	0.589	83.60	0.601	0.501
ZeroR	0.000	80.00	0.000	0.000

TABLE S10: Classification performances of base classifiers for the UEAB datasets.

Classifier	MCC	Accuracy (%)	FMeasure	Precision
BayesNet	0.533	81.20	0.569	0.456
DecisionStump	0.321	80.80	0.273	0.212
DecisionTable	0.416	77.60	0.337	0.223
IBk	0.678	86.40	0.649	0.572
J48	0.419	76.40	0.437	0.290
JRip	0.437	76.80	0.456	0.311
LibSVM	**0.894**	**88.80**	0.651	**0.629**
LMT	0.736	87.20	0.657	0.587
Logistic	0.610	80.00	0.472	0.384
NaiveBayes	0.458	76.40	0.536	0.412
NaiveBayesUpdateable	0.458	76.40	0.536	0.412
OneR	0.344	76.80	0.294	0.166
PART	0.464	78.00	0.476	0.342
RandomForest	0.696	84.40	0.528	0.464
RandomTree	0.388	75.20	0.356	0.209
REPTree	0.424	80.00	0.373	0.282
SGD	0.771	87.60	**0.667**	0.612
SimpleLogistic	0.736	87.20	0.657	0.587
VotedPerceptron	0.595	84.00	0.639	0.544
ZeroR	0.000	80.00	0.000	0.000

TABLE S11: Classification performances of other ensemble of classifiers used in DE-HEoC for the appendicitis datasets.

	Ensemble	MCC	Acc	Prec	F-Meas	AUC	SEN	SPEC
1	AdaBoostM1	0.58	90.54	0.90	0.90	0.83	0.91	0.65
2	Bagging	0.32	86.49	0.84	0.85	0.77	0.86	0.39
3	RandomCommittee	0.46	87.84	0.87	0.88	0.87	0.88	0.56
4	RandomSubSpace	-0.07	83.78	0.74	0.79	0.72	0.84	0.13
5	RandomForest	0.50	89.19	0.88	0.89	0.83	0.89	0.56
6	RandomTree	0.44	82.43	0.87	0.84	0.77	0.82	0.72
7	Stacking	0.00	86.49	0.75	0.80	0.48	0.86	0.14

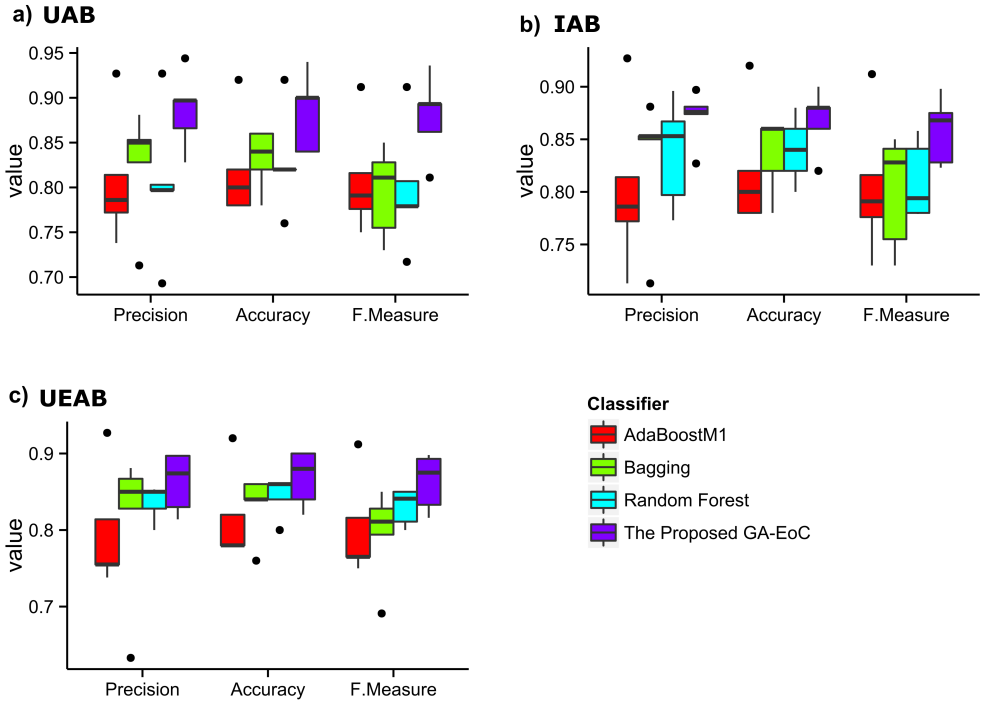
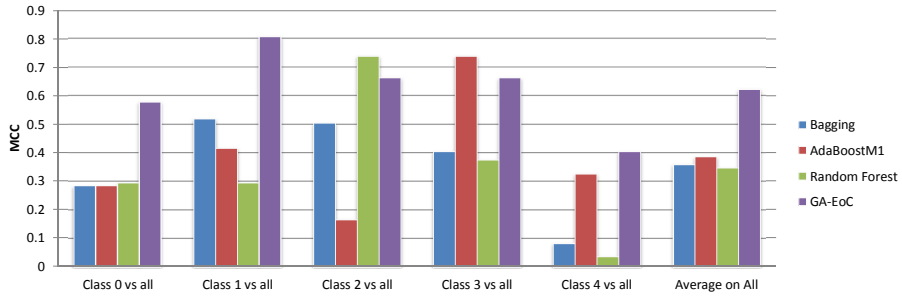


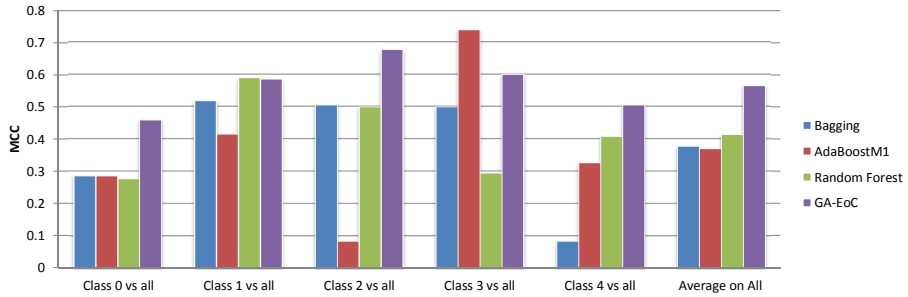
FIGURE S1: The classification performances of `AdaBoostM1`, `Bagging`, `Random Forest` and GA-EoC are compared in terms of Precision, Accuracy and F-Measure scores for (a) UAB, (b) IAB and (c) UEAB datasets.

TABLE S12: Classification performances of other ensemble of classifiers used in DE-HEoC for the australian datasets.

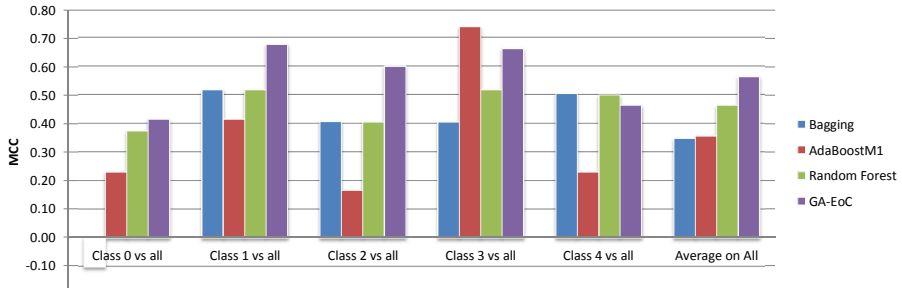
	Ensemble	**MCC**	**Acc**	**Prec**	**F-Meas**	**AUC**	**SEN**	**SPEC**
1	AdaBoostM1	0.66	85.51	0.85	0.85	0.92	0.86	0.79
2	Bagging	0.70	86.96	0.87	0.87	0.92	0.87	0.82
3	RandomCommittee	0.67	85.92	0.86	0.86	0.90	0.86	0.79
4	RandomSubSpace	0.66	85.71	0.86	0.85	0.92	0.86	0.75
5	RandomForest	0.70	87.16	0.87	0.87	0.92	0.87	0.82
6	RandomTree	0.59	81.99	0.82	0.82	0.80	0.82	0.77
7	Stacking	0.00	68.32	0.47	0.55	0.49	0.68	0.32



(a) UAB



(b) IAB



(c) UEAB

FIGURE S2: Classification performances (in terms of MCC) of the genetic algorithm-based ensemble of classifiers vs three ensemble methods for the (a) *UAB*, (b) *IAB* and (c) *UEAB* datasets.

TABLE S13: Classification performances of other ensemble of classifiers used in DE-HEoC for the bupa datasets.

	Ensemble	**MCC**	**Acc**	**Prec**	**F-Meas**	**AUC**	**SEN**	**SPEC**
1	AdaBoostM1	0.14	69.83	0.65	0.64	0.67	0.70	0.39
2	Bagging	0.33	74.38	0.73	0.72	0.73	0.74	0.53
3	RandomCommittee	0.33	71.90	0.72	0.72	0.75	0.72	0.61
4	RandomSubSpace	-0.01	69.01	0.56	0.58	0.65	0.69	0.31
5	RandomForest	0.33	74.38	0.73	0.72	0.75	0.74	0.54
6	RandomTree	0.34	73.14	0.72	0.73	0.66	0.73	0.60
7	Stacking	0.00	69.83	0.49	0.57	0.48	0.70	0.30

TABLE S14: Classification performances of other ensemble of classifiers used in DE-HEoC for the haberman datasets.

	Ensemble	MCC	Acc	Prec	F-Meas	AUC	SEN	SPEC
1	AdaBoostM1	0.27	66.82	0.66	0.66	0.63	0.67	0.60
2	Bagging	0.24	65.42	0.64	0.64	0.65	0.65	0.57
3	RandomCommittee	0.10	57.48	0.58	0.58	0.56	0.57	0.52
4	RandomSubSpace	0.23	65.89	0.64	0.64	0.64	0.66	0.54
5	RandomForest	0.13	59.35	0.59	0.59	0.62	0.59	0.54
6	RandomTree	0.06	55.14	0.56	0.56	0.53	0.55	0.51
7	Stacking	0.00	62.15	0.39	0.48	0.49	0.62	0.38

TABLE S15: Classification performances of other ensemble of classifiers used in DE-HEoC for the monk-2 datasets.

	Ensemble	MCC	Acc	Prec	F-Meas	AUC	SEN	SPEC
1	AdaBoostM1	0.90	95.70	0.96	0.96	0.99	0.96	0.95
2	Bagging	1.00	100.00	1.00	1.00	1.00	1.00	1.00
3	RandomCommittee	1.00	100.00	1.00	1.00	1.00	1.00	1.00
4	RandomSubSpace	0.84	92.72	0.93	0.92	0.99	0.93	0.86
5	RandomForest	1.00	100.00	1.00	1.00	1.00	1.00	1.00
6	RandomTree	0.96	98.01	0.98	0.98	0.98	0.98	0.98
7	Stacking	0.00	66.23	0.44	0.53	0.49	0.66	0.34

TABLE S16: Classification performances of other ensemble of classifiers used in DE-HEoC for the pima datasets.

	Ensemble	MCC	Acc	Prec	F-Meas	AUC	SEN	SPEC
1	AdaBoostM1	0.31	76.21	0.74	0.75	0.77	0.76	0.52
2	Bagging	0.36	78.81	0.77	0.77	0.78	0.79	0.51
3	RandomCommittee	0.22	73.61	0.71	0.72	0.74	0.74	0.46
4	RandomSubSpace	0.18	75.65	0.71	0.70	0.77	0.76	0.34
5	RandomForest	0.39	78.81	0.77	0.78	0.81	0.79	0.56
6	RandomTree	0.28	72.86	0.73	0.73	0.64	0.73	0.56
7	Stacking	0.00	75.09	0.56	0.64	0.49	0.75	0.25

TABLE S17: Classification performances of other ensemble of classifiers used in DE-HEoC for the saheart datasets.

	Ensemble	MCC	Acc	Prec	F-Meas	AUC	SEN	SPEC
1	AdaBoostM1	0.30	75.23	0.74	0.74	0.76	0.75	0.52
2	Bagging	0.26	75.85	0.73	0.73	0.75	0.76	0.45
3	RandomCommittee	0.21	73.07	0.71	0.72	0.70	0.73	0.46
4	RandomSubSpace	0.16	76.16	0.74	0.68	0.75	0.76	0.30
5	RandomForest	0.22	74.92	0.72	0.72	0.77	0.75	0.43
6	RandomTree	0.26	72.45	0.72	0.72	0.63	0.72	0.53
7	Stacking	0.00	75.23	0.57	0.65	0.50	0.75	0.25

TABLE S18: Classification performances of other ensemble of classifiers used in DE-HEoC for the sonar datasets.

	Ensemble	MCC	Acc	Prec	F-Meas	AUC	SEN	SPEC
1	AdaBoostM1	0.52	79.45	0.79	0.79	0.82	0.79	0.69
2	Bagging	0.48	78.08	0.78	0.77	0.82	0.78	0.64
3	RandomCommittee	0.61	82.88	0.83	0.83	0.87	0.83	0.77
4	RandomSubSpace	0.38	74.66	0.74	0.72	0.79	0.75	0.56
5	RandomForest	0.59	82.19	0.84	0.80	0.91	0.82	0.66
6	RandomTree	0.52	78.77	0.79	0.79	0.76	0.79	0.73
7	Stacking	0.00	67.12	0.45	0.54	0.47	0.67	0.33

TABLE S19: Classification performances of other ensemble of classifiers used in DE-HEoC for the titanic datasets.

	Ensemble	MCC	Acc	Prec	F-Meas	AUC	SEN	SPEC
1	AdaBoostM1	0.45	81.77	0.81	0.81	0.73	0.82	0.58
2	Bagging	0.49	83.97	0.85	0.81	0.74	0.84	0.50
3	RandomCommittee	0.51	84.30	0.85	0.82	0.74	0.84	0.51
4	RandomSubSpace	0.47	83.39	0.84	0.80	0.72	0.83	0.47
5	RandomForest	0.51	84.30	0.85	0.82	0.74	0.84	0.51
6	RandomTree	0.51	84.30	0.85	0.82	0.74	0.84	0.51
7	Stacking	0.00	76.90	0.59	0.67	0.50	0.77	0.23

TABLE S20: Classification performances of other ensemble of classifiers used in DE-HEoC for the wdbc datasets.

	Ensemble	MCC	Acc	Prec	F-Meas	AUC	SEN	SPEC
1	AdaBoostM1	0.86	94.72	0.95	0.95	0.98	0.95	0.91
2	Bagging	0.87	94.97	0.95	0.95	0.98	0.95	0.92
3	RandomCommittee	0.90	96.23	0.96	0.96	0.98	0.96	0.93
4	RandomSubSpace	0.89	95.73	0.96	0.96	0.98	0.96	0.92
5	RandomForest	0.88	95.23	0.95	0.95	0.99	0.95	0.90
6	RandomTree	0.83	93.22	0.93	0.93	0.92	0.93	0.90
7	Stacking	0.00	73.37	0.54	0.62	0.49	0.73	0.27

TABLE S21: Classification performances of other ensemble of classifiers used in DE-HEoC for the Churn datasets.

Classifier	MCC	Acc	Prec	F-Meas	AUC	SEN	SPEC
AdaBoostM1	0.366	87.28	0.853	0.851	0.841	0.873	0.382
Bagging	0.000	85.51	0.731	0.788	0.500	0.855	0.145
RandomCommittee	0.268	86.74	0.862	0.821	0.844	0.867	0.236
RandomSubSpace	0.156	85.93	0.87	0.799	0.883	0.859	0.171
RandomForest	0.133	85.81	0.878	0.795	0.887	0.858	0.163
RandomTree	0.265	85.45	0.824	0.830	0.667	0.854	0.332
Stacking	0.000	85.51	0.731	0.788	0.498	0.855	0.145

TABLE S22: lassification performances of base classifiers used in DE-HEoC for the Churn datasets.

Classifier	**MCC**	**Acc**	**Prec**	**F-Meas**	**AUC**	**SEN**	**SPEC**
BayesNet	0.439	85.99	0.861	0.860	0.834	0.860	0.582
DecisionStump	0.317	86.56	0.840	0.841	0.603	0.866	0.348
DecisionTable	0.548	90.16	0.893	0.892	0.815	0.902	0.543
IBk	0.237	83.38	0.812	0.821	0.603	0.834	0.368
J48	0.742	94.03	0.938	0.937	0.876	0.940	0.727
JRip	0.794	95.14	0.950	0.950	0.873	0.951	0.789
KStar	0.187	83.17	0.801	0.812	0.700	0.832	0.316
LibSVM	0.000	85.51	0.731	0.788	0.500	0.855	0.145
Logistic	0.285	85.87	0.830	0.835	0.807	0.859	0.340
LWL	0.369	87.13	0.851	0.852	0.834	0.871	0.399
NaiveBayes	0.485	87.64	0.872	0.874	0.834	0.876	0.592
OneR	0.00	85.51	0.731	0.788	0.500	0.855	0.145
PART	0.598	90.46	0.901	0.902	0.823	0.905	0.664
RandomTree	0.265	85.45	0.824	0.830	0.667	0.854	0.332
REPTree	0.000	85.51	0.731	0.788	0.498	0.855	0.145
SGD	0.248	86.05	0.829	0.826	0.570	0.86	0.280
VFI	0.018	17.22	0.781	0.097	0.686	0.172	0.837
VotedPerceptron	0.060	85.57	0.877	0.790	0.508	0.856	0.148
ZeroR	0.000	85.51	0.731	0.788	0.498	0.855	0.145

F
List of Symbols

I_r	Intensity of the red colour
I_g	Intensity of the green colour
m	the number of rows (genes/features)
n	the number of columns (samples/instances/)
$D_{m,n}$	the dataset containing m rows of features and n columns of samples
r_{ij}	expression ratio for the ith gene in jth sample
P	the probability of an event
ω	a class label
Ω	the set of class labels
$\Re^m$	the m-dimensional feature space
$\mathbb{C}$	the classifier
$\mathbb{C}^*$	the set of classifiers
$\mathbb{T}$	the training dataset
$\mathbb{U}$	the unlabelled testing dataset
$\mathcal{F}$	the set of features
$\mathcal{G}$	feature subset
$\mathbb{E}$	the ensemble of classifiers
k	the number of base classifiers
$\mathbb{R}$	the set of real numbers
$n!$	factorial of n calculated as $1 * 2 * \cdots * n$
$\mathbb{E}_{mv}$	majority-voting (unweighted) ensembles
$\mathbb{I}$	an individual
$\mathbb{P}$	the population
e	the possible number of ensemble combinations $e = 2^k$
`fit`	the fitness function

f	the number of folds
Obj_{mcc}	the objective function to maximise MCC score
R_{χ}	the recombination rate
R_{μ}	the mutation rate
$\mathbb{E}_{wv}$	weighted-voting ensembles
w	weighted od base classifier for voting in ensembles
w^0	voting weight of class labelled $\omega_i = 0$
w^1	voting weight of class labelled $\omega_i = 1$
$\overrightarrow{X}$	individual used in DE
G	the number of generation in DE
NP	the population size in DE
$\overrightarrow{V}$	mutated individual in DE
$\overrightarrow{U}$	recombined individual in DE
$\mathbb{E}_{wv^*}$	best individual returned by DE
(TD)	train-fold data
(VD)	validation-fold data
(TM)	trained weighted-voting ensemble model
(F)	Scaling Factor
$\mathbf{S}$	Set of solutions
x^*	Pareto-optimal solution
λ	hyperparameter
Obj_{acu}	objective function to maximise the accuracy score
Obj_{div}	objective function to maximise the diversity score
Obj_{sze}	objective function to minimise the ensemble size
$l(z_j)$	the number of base classifiers that recognise a sample z_j correctly
$\lvert\mathbb{E}\rvert$	ensemble size
$\mathbb{S}$	selected population in NSGA-II
$\mathbb{Q}$	child population in NSGA-II
Prb_{sz}	size of problem in NSGA-II
Pop_{sz}	size of the population in NSGA-II
$\mathbb{MP}$	merged population of NSGA-II
$\mathbb{F}$	Pareto-frontier