Design of Intelligent Ensembled Classifiers Combination Methods

by

Shayma Alani

A thesis submitted for the degree of

Doctor of Philosophy

Department of Electronic and Computer Engineering

College of Engineering, Design and Physical Sciences

Brunel University London

September 2015

Abstract

Classifier ensembling research has been one of the most active areas of machine learning for a long period of time. The main aim of generating combined classifier ensembles is to improve the prediction accuracy in comparison to using an individual classifier. A combined classifiers ensemble can improve the prediction results by compensating for the individual classifier weaknesses in certain areas and benefiting from better accuracy of the other ensembles in the same area.

In this thesis, different algorithms are proposed for designing classifier ensemble combiners. The existing methods such as averaging, voting, weighted average, and optimised weighted method does not increase the accuracy of the combiner in comparison to the proposed advanced methods such as genetic programming and the coalition method. The different methods are studied in detail and analysed using different databases. The aim is to increase the accuracy of the combiner in comparison to the standard stand-alone classifiers. The proposed methods are based on generating a combiner formula using genetic programming, while the coalition is based on estimating the diversity of the classifiers such that a coalition is generated with better prediction accuracy.

Standard accuracy measures are used, namely accuracy, sensitivity, specificity and area under the curve, in addition to training error accuracies such as the mean square error. The combiner methods are compared empirically with several stand-alone classifiers using neural network algorithms. Different types of neural network topologies are used to generate different models. Experimental results show that the combiner algorithms are superior in creating the most diverse and accurate classifier ensembles. Ensembles of the same models are generated to boost the accuracy of a single classifier type. An ensemble of 10 models of different initial weights is used to improve the accuracy. Experiments show a significant improvement over a single model classifier.

Finally, two combining methods are studied, namely the genetic programming and coalition combination methods. The genetic programming algorithm is used to generate a formula for the classifiers' combinations, while the coalition method is based on a simple algorithm that assigns linear combination weights based on the consensus theory. Experimental results of the same databases demonstrate the effectiveness of the

proposed methods compared to conventional combining methods. The results show that the coalition method is better than genetic programming.

Table of Contents

Acknowledgements

In the name of Allah

I would like to express my deep and sincere gratitude to whom I am greatly indebted to my supervisor Dr. Maysam Abbod for the continuous advices and guidance during this research, I appreciate his patience and support for this research.

My deepest gratitude goes to my family, my parents, especially my mother, I dedicate this thesis to her.

Finally , I would like to thank all of whom have patiently supported and encouraged me.

Declaration

I certify that the effort in this thesis has not previously been submitted for a degree nor has it been submitted as part of requirements for a degree. I also certify that the work in this thesis has been written by me. Any help that I have received in my research work and the preparation of the thesis itself has been duly acknowledged and referenced.

Signature of Student

Shayma Alani

September 2015, London

List of Figures

List of Tables

List of Abbreviations

AI	Artificial Intelligent
AIA	Abstract Intelligent Agents
ANFIS	Adaptive Neuro-fuzzy Inference System
ANN	Artificial Neural Network
AUC	Area Under the Curve
CB	Coalition-based
CC	Coefficient of Correlation
CFGs	Characteristic Function Games
CSG	Coalition Structure Generation
CT	Computed Tomography
CV	Cross Validation
ECD	Ensembled Combiner Design
ECG	Electrocardiography
GCNN	Generalised Classifier Neural Network
GEM	Generalised Ensemble Method
GP	Genetic Programming
HMM	Hidden Markov Model
IA	Intelligent Agent
LCS	Learning Classifier System
LISP	LIST Processor
LR	Layered-Recurrent

MAE	Mean Absolute Error
MAP	Maximum A Priori
MRE	Mean Relative Error
MRI	Magnetic Resonance Imaging
MSE	Mean Square Error
MSE	Mean Squared Error
NARX	Non-Linear Auto-Regression with eXogenous
NFC	Neuro-Fuzzy Classifier
NFG	Normal Form Games
NN	Neural Network
PET	Positron Emission Tomography
RBF	Radial Basis Function
RMS	Root Mean Square
RNNs	Recurrent Neural Networks
ROC	Receiver Operating Characteristic
SRM	Structural Risk Minimisation
SVM	Support Vector Machine
TDNN	Time Delay Neural Networks
UCI	University of California, Irvine

Chapter 1: Introduction

1.1 Introduction

Learning can be defined as obtaining knowledge about a certain situation or an environment in which tasks and situation responses are fulfilled. Humans learn things through studying in educational institutes (e.g. schools, colleges, etc.) or through self- or peer-learning. Machines on the other hand do not usually obtain knowledge by themselves. Handling machines requires special kinds of instructions. For many years, scientists have tried to create machines that can interact with the environment the same way that human beings do. (Bull, 2004)

Machine learning relates to advanced computers and encompasses a developing field of work in solving real-world problems. The intricate or sometimes misconceptions about the nature of many domains e.g. data mining or process control, have created a need for methods that are capable of adapting to the task in hand. Learning classifier systems (LCSs) are machine learning methods that embed reinforcement learning, evolutionary computing and other heuristics to generate adaptive systems (Bull, 2004).

Evolutionary computing methods are search algorithms that operate on the principles of natural selection and genetics. These principles have been employed in various domains such as design, optimisation, control, classification, etc. However, reinforcement learning is a type of learning executed through trial and error upon receiving a numerical reward. The learner tries to design state and action combinations to their benefit, with the objective of maximising future reward. Reward is often obtained at the end of a number of actions that have been executed by the learner. Therefore, reward is generally delayed. This approach is known as secondary reinforcers in animal learning theory. The reinforcers act as stimulation factors and are associated with things such as pain, happiness and food. Reinforcement learning has been applied in a wide range of domains such as gaming, control, stimulation and many others (Bull, 2004).

Classification can be defined as the problem of determining the belonging of a set of categories in a new observation; this determination is based on the training set of data that includes the observations whose category membership is known. A simple example to further understand this definition is by categorising received emails into “spam” and “inbox (non-

spam)" or patient diagnosis based on observed characteristics such as gender, age, medical history, severity of ailment, etc. In machine learning, classification is an example of supervised learning, a learning by which a set of identified observations is presented (Dietterich, 2000).

Normally, the individual observations are further explained as a set of quantifiable properties known as "features". These features may take the form of:

- Category: e.g. blood types (A+, AB-, O-, etc)
- Ordinal: e.g. clothes' sizes (small, Xlarge, XXlarge, etc.)
- Integer value: e.g. the number of cars in a certain city
- Real value: e.g. the measurement of blood sugar, blood pressure, vitamin deficiency, etc.
- Previous observations: e.g. comparing current observations to previous ones based upon similarity or distance

Based on the definition of classification, a classifier is known as the algorithm that solidly implements classification. In statistics, classification is mostly done with logistic recognition and the observations are known as explanatory or independent variables, while the predicted ones are outcomes which are considered as dependent variables. In machine learning, the observations are referred to as "instances" and the explanatory variables are "features" and the predicted categories are "classes". Recently, the concept of combining classifiers was proposed as a novel operation to enhance the performance of individual classifiers. These classifiers can be based on various classification techniques and can also generate a different magnitude of correctly classified individuals. The goal of such an operation is to provide more accurate and precise results (Dietterich, 2000).

Learning from the environment, either for real or artificial subjects, is done via reinforcement learning. A certain subject is provided with special tasks, and upon fulfilment, provided (reinforced) by a certain reward. However, in some cases, a negative reward is given as a sort of punishment or sometimes not given. The task can be a short one equipped with multiple feedback operations or it might be a long one where a bunch of operations are made before receiving feedback such as in board games (chess, checkers, etc.).

The learning classifier system (LCS) was proposed by John H. Holland in a book in 1975 (Bull and Kovacs, 2005). In his model, a typical learning classifier comprises three major components. These components are:

- Performance system
- Apportionment of credit procedure
- Rule discovery system

These operations are shown in Figure 1.1.

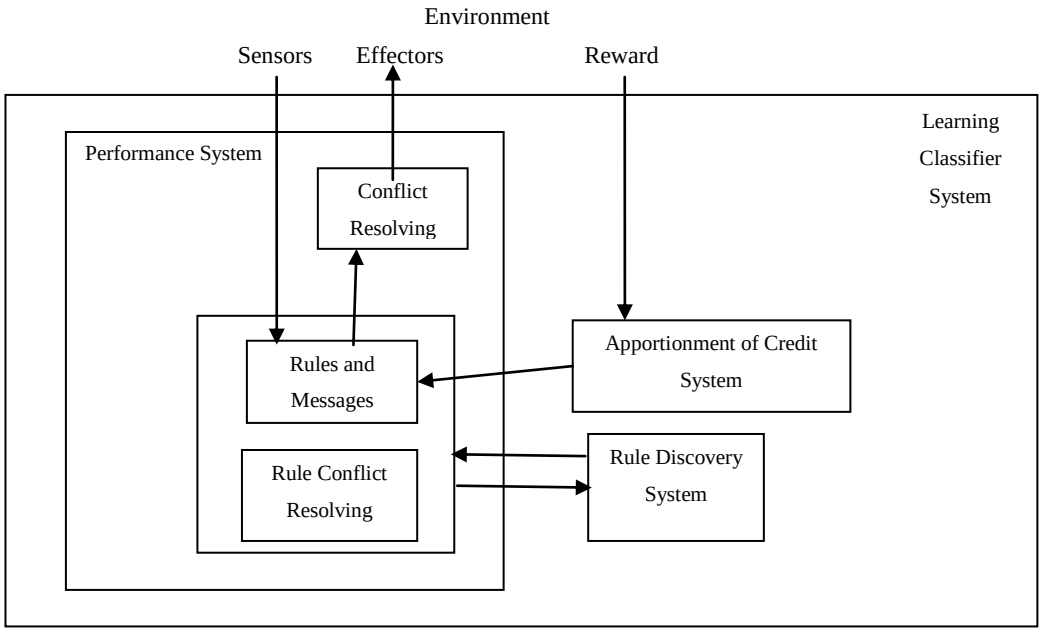


Figure 1.1: Components of a Learning Classifier System

According to Figure 1.1, the performance system comprises a set of rules and a message list. These rules are considered as the system's knowledge base. During processing, the set of rules is considered as the best approximation for the knowledge being analysed by the system. However, all the communication in the system is done through messages. Input is converted into messages via the sensors and then kept in the message list. The rules assign conditions for their execution and generate outputs as effectors. The rules may be considered as "if – then" statements.

First of all, an input message is analysed by matching the first collection of rules in addition to the effecter conditions. In other words, an effecter is shown in the rule collection as other

rules. The main difference is that it generates an output other than another message. In addition, one or more rules might match the message. One message is selected and if it is not selected, an effecter will generate a new message in the list. This will be done a few times before the message placed on the list matches an effecter's condition.

The rule discovery system is quite similar to basic genetic algorithms. In conventional LCSs, the "if – then" rule is portrayed as fixed length strings over small alphabets. Holland has employed three characters which are "0", "1" and "#", with "#" denoting "don't care" in the "if" part of the rule and "copy input" in the "then" portion of the rule.

The fitness implemented in the genetic algorithm is the apportionment of credit procedure instead of a direct function of the rule. The apportionment of credit procedure is known as the "bucket brigade" algorithm due to the method through which credit is passed back via the set of rules. The basic algorithm represented by the "bucket brigade" algorithm is as follows:

0. Assigning initial strength to all classifiers. Clearing the list and appending all input messages to the list. Also, assigning all classifiers to "not active".
1. Activating all classifiers that distinguish messages on the list to "active" and clearing the list
2. Calculating a bid quantity for each classifier appointed "active"
3. Picking randomly, with a probability related to the bid quantity, classifiers to include new messages on the list. Note: each message is marked with the classifier in order for the next to occur.
4. Each classifier that posted a new message pays a payment quantity to the classifier that resulted in activating it.
5. All the classifiers are set to "not active"
6. New environmental messages are included in order to process the list and the operation is repeated again from step 1.

1.2 Aim and Objectives

In general, classifier ensembles are considered a significant development in the field of machine learning and artificial intelligence. They have made it possible for learners to

provide accurate outcomes when it comes to decision making in classification problems and pattern recognition. For that reason, in the process of making this thesis, the main objective is to explore the nature and structure of different ensemble methods and combination methods in order to identify which model provides the best outcome in terms of accuracy and ensemble size for different classification problems.

The next objective is concerned with identifying and studying the nature and structure of an Artificial Neural Network (ANN) in addition to exploring different neural network models, providing the structure of each of the models and determining which of the networks provides the best outcome in machine learning.

Moreover, another objective is to study combination methods in detail by identifying some of the well-known advanced combination methods that are concerned with dealing with huge amounts of data and handling complex classification problems.

The last objective is to implement each model represented in this thesis in real-life problems and to analyse their behaviour under different parameters, comparing them to the actual data in order to distinguish which of the models provides the best predicted outcome when implemented in these particular classification problems.

1.3 Research Methodology

Artificial intelligence is an aspect of computer science that is concerned with making computers more intelligent. The most important requirement for making computers intelligent is the concept of learning. One of the most significant principles in machine learning is ensemble classifiers due to their many functions and benefits in a wide number of applications. This provides the motivation to explore and identify the main aspects of ensemble classifiers, in addition to exploring some of the important combination methods being used in various applications. Neural networks are basically models that mimic the operation of a nervous system such as the brain. These trained networks have many advantages such as real-time operation and adaptive learning. Therefore, this provides motivation for becoming familiar with some of the most popular neural network models. Combination methods are used to combine classifiers of the same type together, thus providing a better outcome and supporting various applications in both machine learning and pattern recognition. This provides the motivation to identify various aspects of combination

methods such as output types, operation levels, combination and algebraic rules, in addition to exploring different methods for classifier combination. Advanced combination methods are currently employed in science and technology and the more complex problems, the more advanced a model should be in order to provide the best prediction and decision making results. The motivation behind this thesis is becoming familiar with some of the most advanced combination methods.

In order to introduce a better algorithm, two main aspects should be taken care of; ensemble accuracy and decrease in error prediction. Ensemble combination methods are important in any ensemble design and each combination method has got its weaknesses and restrictions when it comes to practical implementation. For that reason, a novel combination method is introduced that is based on coalition. Coalition method develops an ensemble method for the datasets.

1.4 Contributions to Knowledge

In this thesis, the Ensembled Combiner Design (ECD) algorithm has been developed in order to improve the accuracy of classifiers. The novelty here is exploring various classifier combination methodologies and techniques, and thus developing a design that provides the most accurate prediction output. This aim was fulfilled by developing a combination method using advanced algorithm.

Throughout this thesis, the introduction of Coalition-based algorithms and Genetic Programming has led to the fact that prediction accuracy can be obtained by developing a simple yet effective approach based on the principle of diversity so as to guide the process of producing the optimal classifier combination, leading to a better accuracy in the predicted outputs. As a result, the employment of a coalition of different classifiers can improve the accuracy of prediction by excluding the weaknesses of some data points, and including the strengths of other data points, in order to obtain an accurate prediction.

The final novelty to this thesis is the most important one, which is the design of new ensemble combination methods that are based on the consensus theory and evolutionary algorithms. To obtain the best characteristics of the proposed combination methods, extensive experimentations and research have been adopted on various databases and comparisons have been made with single conventional combiners such as Voting, Average, Weighted Average

and Optimised Weighted Average combination methods. This comparison was made based on the output performance of each of the combiners mentioned in different arrangements of databases in order to obtain, as previously mentioned, the most optimal characteristics of the proposed combination methods. Thus, The results obtained have successfully demonstrated the best characteristics for the new proposed methods.

1.5 Thesis Outline

The thesis is divided in 7 chapters which are as follows:

- Chapter 1-Introduction: This chapter talks about the research briefly, the objectives of the research, the area which the research is designed for, the addition to knowledge and the research outline.
- Chapter 2-Literature Review: This chapter illustrates some previous studies in the research field including classifier techniques and ensembling techniques.
- Chapter 3-Neural Networks: The theory of neural networks is presented in this chapter, and the different types of networks are described and presented. Each network is tested with a number of data sets. The data sets are presented and described.
- Chapter 4-Ensembles: This chapter presents the ensembling techniques for different classifiers. The different data sets are tested, and the results are presented for different techniques and classifiers.
- Chapter 5-Combination Methods: This chapter shows the standard combination methods, such as average, weighted average, optimised weighted average and finally voting. The results for all the data sets are presented and compared.
- Chapter 6-Intelligent Combination Methods: In this chapter, two intelligent combinations methods are presented, namely genetic programming and the coalition method. The results are presented for all the data sets and compared.
- Chapter 7-Conclusions and Future Work: This is the final chapter which illustrates the stages of the research in brief and presents some ideas that may improve the model in the future.

Chapter 2: Literature Review

2.1 Introduction

Users can compare classifiers across various data sets through the Matlab classification toolbox. Implementation of classifiers such as Neural Networks, Decision Tree, Gaussian Mixture Model, and Gaussian are included in the classification toolbox of (Duin,et al.,2000). A list of functions for unsupervised and supervised classification algorithms is also included in the Matlab classification toolbox. Using convenient graphic user interfaces, significant help in designing categorisation methods for synthetic as well as experimental data is provided by these algorithms. The graphic user interface is designed for two-class and two-dimensional problems (Duin and Tax, 2000). However, most of the algorithms can be used on multi-class data and higher dimensional data. Moreover, by using one of several feature selection algorithms, the higher dimensional data can be reduced to two dimensions. Basic knowledge of Matlab is essential as most of the functions are operated only within the graphic user interface. This chapter reflects upon a critical literature of classifiers such as regression, neural networks, neuro-fuzzy systems, support vector machine (SVM) and Bayesian, and combination methods such as average (mean), weighted average, optimised weighted average and majority voting.

2.2 Classifiers

2.2.1 Regression

Logistic regressions are considered as linear classifiers and are one of the widely used and popular classification techniques. Measured in both first-best and log loss classification accuracy across a number of tasks, logistic regression is one of the best probabilistic classifiers. It is a discriminative probabilistic classification model that operates over real-valued vector. “Features” are those dimensions of the input vectors that are classified and no

limitations are imposed if they are correlated. Multinomial classification is provided when the logistic regression is implemented, i.e., more than two possible output categories are allowed by it (Krishnapuram et al., 2005).

Linear regression is similar to neural network but without the hidden layer. However, one of the major disadvantages of logistic regression is that, in comparison to the other classifiers, it is relatively slow to train. In addition, it requires extensive tuning in the form of feature implementation and selection to achieve state-of-the-art classification performance. In cases where the categories are mutually exclusive and exhaustive, logistic regression models provide multi-category classification (Martin, et al., 2002). Inputs are coded as real-valued vectors of a fixed dimensionality, and are also known as features or predictors. No limitation is imposed on them to be fully or highly linearly correlated, to be with regularisation or to be independent (Cung, et al., 2011).

For multinomial logistic regression, there are some aliases such as Softmax and generalised linear model, regularised regression and shrinkage, the Lasso and Ridge regression, the neural network, maximum entropy classifier and polytomous logistic regression (Kotsiantis, Kanellopoulos and Zaharakis, 2006). Multinomial logistic regression and multi-class, polychotomous logistic regressions are also referred to as multinomial logistic regression. The maximum entropy principle is followed by logistic regression estimation, and therefore, it is also known as “maximum entropy modelling”. The “maximum entropy classifier” is the resulting classifier.

With a logistic activation function trained under log loss, the single-output and one-layer neural network is equivalent to binary logistic regression. It is also known as classification with a single neuron. Ridge regression is referred to as Maximum a priori (MAP) estimation with Gaussian. Parameter shrinkage is also known as MAP estimation with Cauchy, Laplace or Gaussian priors. Regularised regression is equivalent to Laplace and Gaussian priors. With the logit link function, logistic regression is a generalised linear model (Xue and Titterington, 2008). Linear predictors can be converted into probabilities through logistic regression. Several classes are implicated in adapting the stats package logistic regression models for implementations of classifiers. To convert arbitrary objects into mappings to values from string-based features, a feature extractor is used first. Second, these features are converted by a symbol into dimensions. The conversion of arbitrary objects into vectors is supported by a

symbol table and a feature extractor. Finally, to convert the string-based category representations into the integer representation, another symbol table is used.

2.2.2 Neural Networks

A neural network (NN) consists of units (neurons) arranged in layers. An input vector is converted by these neurons within the network into some output. An input is taken by each unit and non-linear function is applied to it and then the output is passed on to the next layer. These networks are also known as feedforward networks, i.e. the output of a unit on the first layer is fed into all units on the next layer (Zhou, et al., 2002). However, there is no feedback to the previous layer. Weightings are applied to the signals that are passing from one unit to another and in order to adapt a neural network to the particular issue at hand, these weightings are tuned in the phase of training. However, for a variety of reasons, the standard feedforward multi-layer NNs often fail to converge completely in classification problems. This failure may occur due to an incorrect architecture with too few hidden neurons and too few layers of weights (Zhang, 2000).

Applications have been found for neural networks in a wide variety of problems. Using Matlab newff function, the neural network classifier is the code that is written for the classification of an image. It is a supervised classification technique and is also known as a clustering tool. Neural networks have proved themselves as proficient classifiers and they are particularly well-suited for addressing non-linear problems. The neural network is certainly one of the best candidates for solving issues given the non-linear nature of factual world phenomena. The use of neural networks is one of several approaches for classifying data (Moradi and Zulkernine, 2004). One way to perceive neural networks is that any number of numeric outputs is produced by them through accepting any number of numeric inputs. They are virtual output-input devices. Patterns are also recognised by neural networks in addition to function fitting. One of the recent techniques in neural networks classification on Matlab is known as GCNN. This technique provides “gradient descent learning on smoothing parameter” and is known to be a toolbox for generalized classifier NN in addition to a recent form of radial basis function RBF-NN (ÇUKUROVA UNIVERSITY, n.d).

2.2.3 Neuro-Fuzzy Systems

Neural networks are known to be suitable for the different applications concerning both pattern classification and recognition. In addition, Matlab contains an contemporary library and a toolbox concerning neural networks. To further explain the development of a fuzzy system, the use of strategies of heuristic learning based on the principle of neural networks is known as “neuro-fuzzy”,Which is the combination of both the fuzzy logic and the artificial neural networks. This kind of systems have shown to mitigate the issues of machine investigation that includes both dimension and space. Furthermore, the hybrid intelligent system in its turn combines two or more intelligent algorithms and methods (Czogała and Łęski, 2000).

In terms of features and basic concepts , the advantage of a neural network learning ability to design a network is taken by a neuro-fuzzy system. To present data, expert knowledge and fuzzy system are used by such a system and then neural network is used in order to adjust output/input membership and develop IF-THEN rules to improve the overall system’s performance. The overall structure of a neural network is similar to that of neuro-fuzzy system. The system is comprised of five layers. In order to present the fuzzy system, it has three middle layers and an output and input layer. The layers exist within the structure of a neuro-fuzzy system. The input layer is the first layer within which the neurons enter into the network. The fuzzification layer is the second layer within which the input neurons are fuzzified according to the membership function. The fuzzy rule layer is the third layer which represents the fuzzy rules of the system. Each neuron is mapped into a fuzzy rule. The output fuzzy set is the fourth layer that represents the output neurons processed by the fuzzy system. The defuzzification layer is the fifth and the last layer within which the output membership functions are used to obtain a value for the input neurons.

The adaptive neuro-fuzzy inference system (ANFIS) depends on the basis of functional equivalence and with some limitations between radial basis function neural networks RBF-NN and Takagi-Sugeno-Kang (TSK) fuzzy systems. The output is a single unit directly calculated through weighting of the inputs based on knowledge base fuzzy rules. In addition they are assigned by a neural network computational algorithm. The Figure 2.1 represents an ANFIS model having two input variables x and y in addition to two rules (Thiago M. Geronimo et al., 2013).

The important things to be considered when it comes to ANFIS performance is the initial number of parameters in the system in addition to the number of inputs and rules. It consists of five layers; the first layer contains the two inputs as well as the rule inputs, while the second and third layers contain the logical operations for the rules. The fourth layer contains the output logic rules and the last layer contains the output which was earlier mentioned the way to obtain it.

Fuzzy system provided explanation ability and knowledge representation (Paiva and Dourado, 2004). However neural network provided learning ability. Adaptability, imprecision tolerance and uncertainty tolerance are provided by both neural network and fuzzy system.

With the nature of neural networking that is based on connections and through combination of human-like techniques of fuzzy systems, neuro-fuzzy is introduced. With the ability to devise IF-THEN rules, the main characteristic of neuro-fuzzy systems its universal-purpose approximation. In addition, both accuracy and interpretability are two characteristics in fuzzy modeling and both play a role in strengthening neuro-fuzzy systems (Abraham, 2001). Generally, both advantages of neural networks and fuzzy systems can be found in neuro-fuzzy systems. Therefore, a fuzzy network that has the characteristics of both surpassing the limitations of fuzzy systems and neural network limitations in addition to containing a fuzzy interface system is known as a "neuro-fuzzy system". In addition, a "neuro-fuzzy classifier (NFC)" is considered as one of the neuro-fuzzy systems.

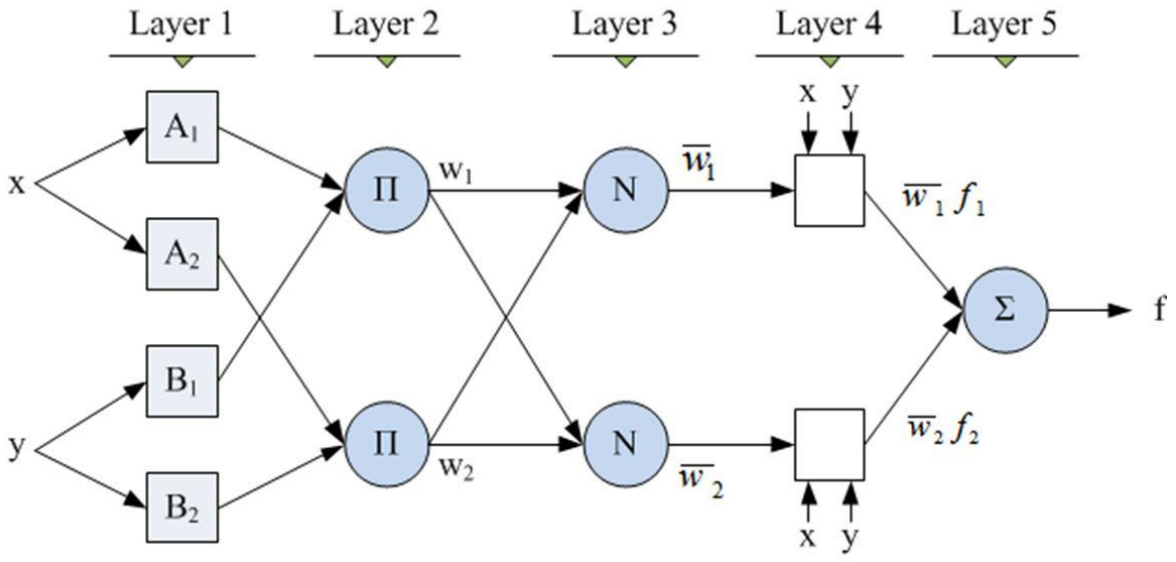


Figure 2.1: ANFIS Model (Thiago M. Geronimo et al., 2013)

2.2.4 Other Classifiers

Support Vector Machine (SVM): One of the new types of pattern classifiers are SVMs. Used for regression analysis and classification, SVMs are supervised learning models in machine learning associated with learning algorithms that can recognise patterns and analyse data. . The main aim of SVMs is to minimise an upper bound of the generalisation error through maximisation of the bargain between the data and the separating hyper plane, unlike traditional networks such as neural networks that minimise the empirical training error. Ranging from image retrieval, recognition information, speaker and speech verification, text categorisation and detection, digit recognition, recognition of handwritten characters, object detection and face recognition, verification and detection, the SVMs have been applied successfully to a number of applications (Tong and Koller, 2002). Under small training sample conditions in high dimensional spaces, SVMs can generalise well enough. Also, in comparison to the traditional empirical risk minimisation principle employed by most neural networks, they have been shown to be superior. Due to numerous real-life data, decent generalisation performance is shown by SVMs. The principle of Structural Risk Minimisation (SRM) in the statistical learning theory lays the foundation of SVMs (Poggio, 2001). SVMs provide better generalisation abilities, i.e., performances on unseen test data. The factor on which the performance of SVMs largely depends is the choice of kernels. A non-linear classification, can be performed efficiently by SVMs in addition to performing linear classification.

Bayesian Classifier The probability of misclassification is minimised by the Bayesian classifier in statistical classification. The Bayesian classifier is based on Bayes' theorem. The idea that the role of an agent is to predict the values of features for members of a class lays the foundation of a Bayesian classifier. However, in the case of the agent being unaware of the class, the Baye's rule can be used to predict some of the features of the given class (Ramoni and Sebastiani, 2001). A probabilistic model of the uses and features is built by the learning agent in a Bayesian classifier and that particular model is used by the learning agent to predict the classification of a new example. Observed variables are probabilistically related to classification which is a latent variable, whereas the Bayesian classifier is a probabilistic model. In the probabilistic model then the classification emerges as an inference. Since common values for the features are shared, examples are grouped in classes. Such classes are referred to as natural kinds. When there is appropriate independence assumption, i.e., when the other features are independent of the given class and when class is a good predictor of the

other features, then Bayesian classifier produces significant results. For natural kinds, this may be appropriate due to classes. In distinguishing the objects that are not distinguished by hands, the evolution of classes take place within natural kinds (Kim and Ghahramani, 2012). Nouns are often associated with natural kinds like the class of chairs or the class of dogs. Bayesian classifiers are also regarded as statistical classifiers, as class membership probabilities are predicted by them.

2.3 Classifier Ensembles

To talk about classifier ensembles, it is first necessary to become familiarised with the concept of supervised learning, particularly how each classifier is generated. A learning algorithm is shown with a training set (instances) $\{(x_1, y_1), (x_1, y_1), \dots, (x_n, y_n)\}$ for an unknown function $y_i = f(x_i)$. The value of $x_i \in X$ is a vector $< x_{i,1}, x_{i,2}, \dots, x_{i,k} >$ and each x_i is known as a feature or an attribute. These features can have an either real or discrete (nominal) value, for instance, age, hair colour, height, weight, etc. Ω is considered as a set of values that correspond to classes or labels. In the case of classification, each $x_i \in X$ corresponds to one value $\omega \in \Omega$ and in the case of regression a continuous value is used. On the other hand, the work presented here is related to classification problems. An example is of a learning algorithm which will result in a classifier based on a hypothesis about a function f with a new object x and predicting the corresponding y value (Kononeko, 2005).

For the purpose of classification the main goal is to come up with a classifier that minimises the prediction error in an independent test data set. Furthermore, the most popular technique used in evaluating the classifier's performance is 0/1 loss function which is as follows:

$$error_{C,f}(x) = \begin{cases} 0 & if\ C(x) = f(x) \\ 1 & otherwise \end{cases} \qquad (2.1)$$

According to Merriam Webster's Collegiate Dictionary, ensemble is defined as "a group producing a single effect". In the field of machine learning, an ensemble is a group of classifiers that are different and combined in a particular way (Polikar, 2009). The goal of the ensemble is to reach a single decision, as shown in Figure 2.2. The objective for any ensemble method is to improve accuracy of the classification prediction of the learning algorithm. Learning algorithms or trained base learners that are on a given training set are known as base classifiers (Adeva, et al., 2005). Ensembles have gained the interest of many

researchers and scientists and have been popular in research since the introduction of machine learning (Zhou, 2015). This area of supervised learning is known by different names such as multiple classifier systems, committees of learners, classifier ensembles etc. (Dietterich, 2014).

Classifier ensembles have a huge benefit due to the fact that combined prediction can possess more accuracy than the base classifiers that make them. Particularly, the generalisation errors of base classifiers when the errors are not correlated are higher than in the case of classifier ensembles. Dietterich (2014) has shown three arguments to justify using classifier ensembles.

Statistical: the main goal of a learning algorithm is to seek a hypothesis closer to the function needed to be predicted. The purpose of ensemble learning is to order patterns or instances into a set of classifications known as labels or classes. 14.Ensemble provides that learning predicts class label or prior unseen records by providing predictions made by multiple classifiers. In other words, each classification depends on the classification models (classifiers) in which they are motivated from a somewhat ideal set of prior classified patterns. In addition, the classification depends on knowledge provided by an expert in the predefined application.

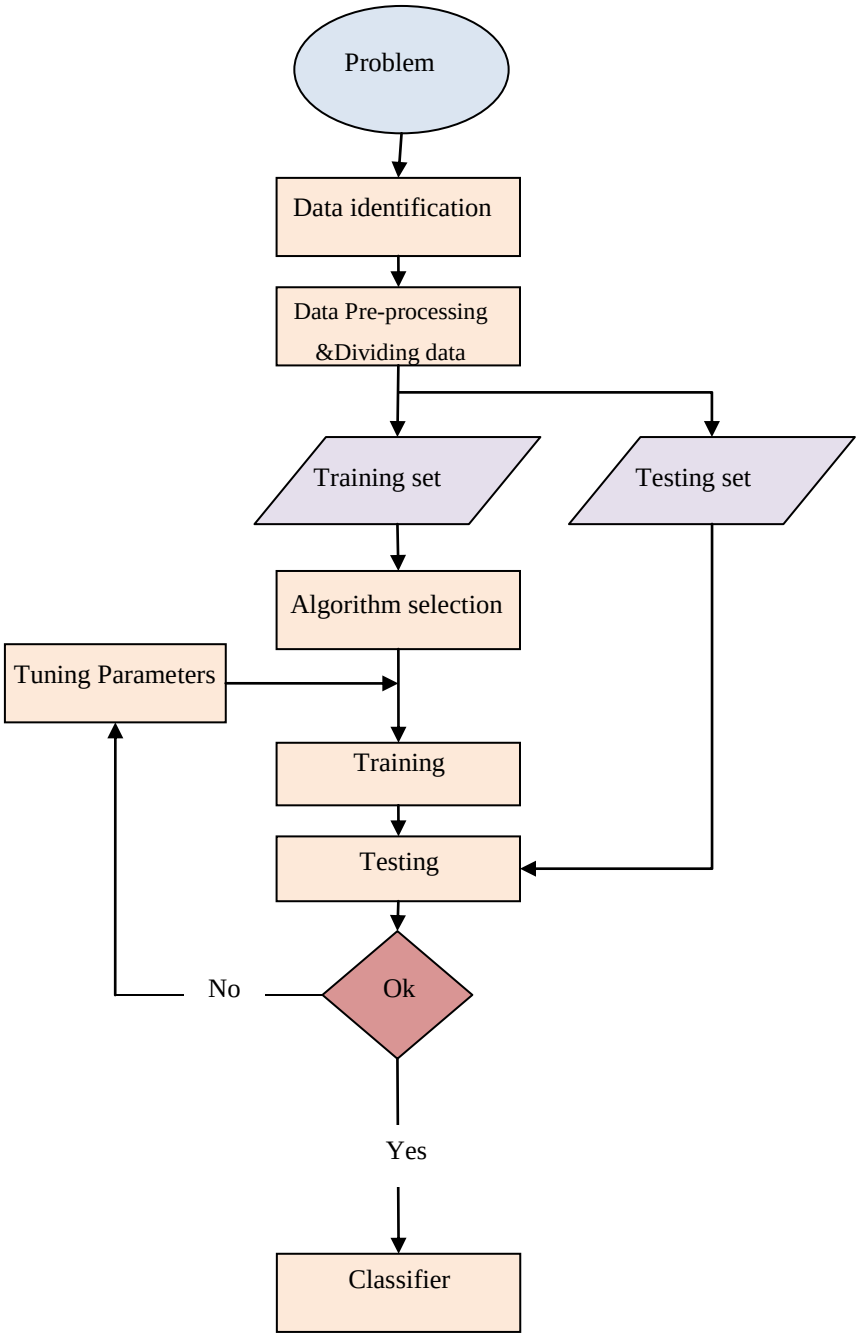


Figure 2.2: Supervised learning process

Moreover, in an ideal supervised learning process, a set of instances or a training set is provided and each set is labelled and the purpose is to build a model for the sake of labelling the new instances. The algorithm that builds the model is known as an "inducer" and an instance of an inducer for a specific set is known as a "classifier". The main idea of ensemble learning is to combine a set of similar models doing the same task for the sake of obtaining a proper global model having more reliable and accurate decisions obtained than from a single

model. Bhulmann and Yu (2003) mentioned that ensemble learning methods began in the late 1970s with a simple Turkeys Twining, a two linear regression ensemble model.

In the past decade, research done in the machine learning community has led to a conclusion that combining the output of multiple classifiers has the effect of reducing the generalisation error (Cruz and Wishart, 2006). Moreover, ensemble methods are found to be very efficient due to the fact that each type of classifier has different "inductive bias" and that they can employ this diversity to reduce the variance error without increasing the bias error. With vast research and interest in the subject of ensemble methods, a huge number of classifiers are now available to researchers. In addition, there are many factors to differentiate between different ensemble methods and these factors are:

1. Inter-classifiers relationship: how each classifier affects the other classifiers and how they are related. Ensemble methods are principally divided into two main types, sequential and concurrent.

2. Combining method: the method of combining classifiers produced by an induction algorithm. The combiner determines the resultant output simply from the outputs of the individual inducers.

3. Diversity generator: for the purpose of developing a more efficient ensemble, there should be some diversity among classifiers. This diversity can be obtained via different presentations of input data or by providing plenty of input data to the output data for the sake of diversity.

4. Ensemble size: number of classifiers in the ensemble.

The ideal ensemble method classifier contains the following blocks:

1. Training set: the instances in a training set are represented as vectors having attribute values. A has been used for the sake of representing the input attributes set containing n attributes. As a result, A and Y represent the goal attribute or the class variable.

2. Base inducer: is defined as an inducer that gets a training set and produces a classifier that yields the input/target attributes. Let's take "I" to denote an inducer, M = I (S) is used to represent A classifier M being stimulated by "I" on "S".

3. Diversity generator plays the role in diversity generation among classifiers.

4. Combiner: for different classifiers, the combiner combines proper classifications.

Specifically, it is important to distinguish between independent and dependent frameworks when it comes to ensemble generation. In dependent frameworks, the output is employed in constructing the subsequent classifier. Therefore, it is likely to use the previous iteration's knowledge to assist the learning of the subsequent iterations. On the other hand, each of the classifiers is constructed in an independent way and thus the outputs are combined. The block diagram in Figure 2.3 represents the framework of a general ensemble classifier:

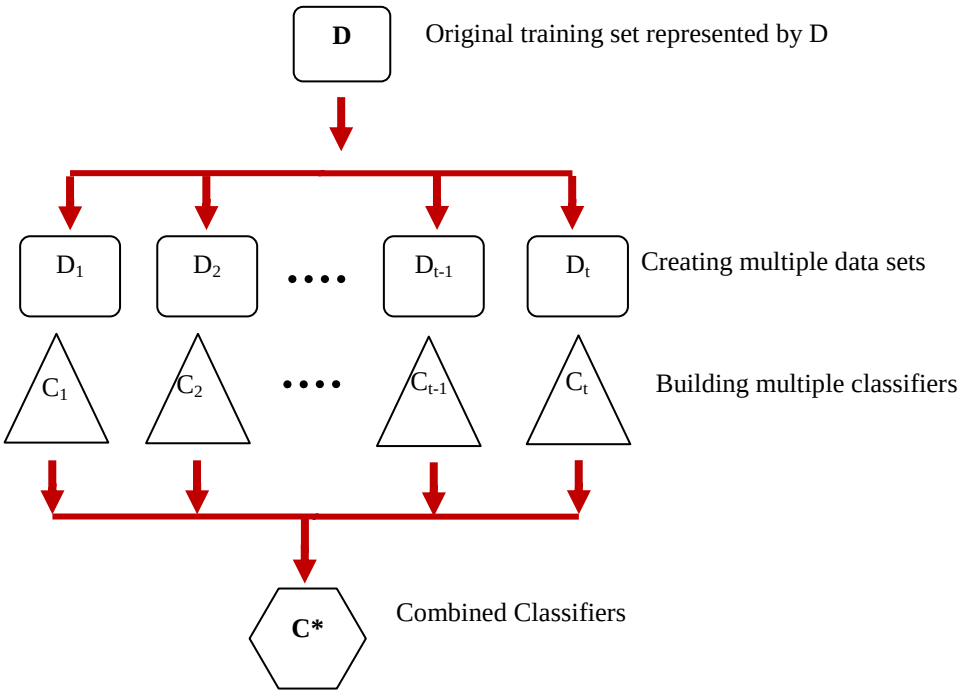


Figure 2.3: Framework of general ensemble classifier

2.4 Combination Methods

Ensemble combination methods have a huge impact on the ensemble design outcome in addition to the generalisation ability of the design itself. As mentioned in section 2.2, Dietterich has mentioned three arguments to substantiate using classifier ensembles. However, for these arguments to prevail and for the use of ensemble to present optimal benefits and results, a combination method must be employed so that the ensemble will not depend on the best single classifier for its output (Melville, 2003).

Essentially, the failure of conventional learning methods is connected with the three issues mentioned by Dietterich (2014). A learning algorithm is known to have a high

"computational variance" however if it experiences a representational issue then the learning algorithm is known to have a high bias (Rokach, 2005).

2.4.1 Average

In machine learning, more specifically in the process of creating artificial neural networks, ensemble averaging is the method based on creating multiple models and producing a desired output by combining the models, in contrast to creating a single model. In brief, an ensemble of multiple models has been shown to perform better than any single model, due to the fact that the multiple errors of the models are averaged out (Naftaly,et al., 1996).

Ensemble averaging is considered as one of the simplest types of committee machines. It is one of the two main types of static committee machines, along with boosting. Ensemble averaging is based on keeping the less satisfactory networks in place, but with less weight. Ensemble averaging theory depends on merely two properties of artificial neural networks (ANNs):

1. The bias can be reduced at the trade-off of increasing variance in any network.
2. The variance can be reduced at no cost to bias when in a group of networks.

Ensemble averaging produces a group of networks each having low bias and high variance and thus combines them to a new network with comparatively low bias and low variance. The main principle of combining networks can be traced back to Pierre-Simon Laplace. In other words, the main idea behind this theory is as follows; creating a set of experts having high variance and low bias, and thus averaging them. So, the set of experts has varying parameters. The steps for the ensemble averaging can be summarised as follows:

1. Generating N experts, each having its own initial values that are mostly assigned randomly from a distribution.
2. Training each expert separately.
3. Combining the experts.
4. Averaging their values.

On the other hand, domain knowledge can be employed for the sake of generating several classes of experts. Experts from each class are trained and then combined. A more

complex approach of ensemble average presents the final result not as an average of all the experts, but as a weighted sum. If each expert is y_i, then the overall result $\tilde{y}$ can be defined as:

$$\tilde{y}(x;\alpha) = \sum_{j=1}^{p} \alpha_j y_j(x) \qquad (2.2)$$

where α represents the set of weights. It is clear that most forms of neural networks are a subset of a linear combination. The standard neural net in which only a single expert is used, is simply a linear combination having all $\alpha_j = 0$ and one $\alpha_k = 1$. A raw average is where all α_j are equal to a constant value, which is 1 over the total number of experts.

The simple mean combiner is considered to be the most popular combiner that can be considered from the "non-trainable" type of combiners, due to its simplicity and effectiveness. In other words, it does not need training of any extra parameters, therefore once the base classifiers are trained, the ensemble is ready to operate. It produces the combined output for a class label:

$$\mu_j(x) = \frac{1}{L}\sum_{i=1}^{L} e_{ij}(x) \qquad (2.3)$$

Which is the equation of the mean of base classifiers e_i in the class of j and an L number of classifiers.

2.4.2 Weighted Average

There are mainly two types of weighted mean combiners mentioned here. The difference between the two is provided from the method of calculating the weights. It is therefore as follows:

Weighted average that requires assigning different weights to the base classifiers, then computing the combined output via averaging the output of the base classifiers:

$$\mu_j(x) = \frac{1}{L}\sum_{i=1}^{L} w_i e_{ij}(x) \qquad (2.4)$$

Where w_i corresponds to the weight given to the tth classifier according to some performance criteria.

In short, the estimated error rate verifies the weight that base classifier e_i recieves.

Weighted average requires calculating the class w_i support from individual support for w_i (class-conscious combiners). Thus the weights are unique for each class and are computed as mentioned before on (2.4).

Since all base classifiers are assumed to have equal weights, the simple average can be considered as a special case of weighted average, despite the fact that it does not necessarily consider weighted average to be superior to simple average. Basically, empirical results here are not capable of confirming that weighted average is always better than simple average. This can be due to the difficulty in the estimation of the weights in the case of noisy and insufficient data or where the ensemble is rather large and learning the weights can result in overfitting. Commonly, simple average is suitable for ensembles where the base classifiers show similar performance, while weighted average is more convenient and provides better results when their performances are different.

2.4.3 Optimised Weighted Average

According to different research and observations concerning the various effects of the weighted averaging method on the variance and bias aspects of the prediction error, a significant amount of reduction in the variance can be obtained by averaging just above neural networks' initial conditions without the necessity of changing the architecture or the training sets. The minimum prediction error for the ensemble is reached later than if it is a single network. In other words, the ensemble prediction error is found to be more flat compared to the error in a single network and as a result, it simplifies the optimal stopping decision. The main goal of this approach comes from observing the behaviour of the bias/variance composition, specifically the fact that averaging ensembles has no effect on the bias aspect of the error. However it reduces the variance when the estimators in the process of averaging are independent.

The procedure here is to estimate a function, $fD(x)$ of the observed data characteristics x to predict class label denoted by y, based on a training set $D = \{(x_1, y_1), \ldots, (x_L, y_L)\}$, and using some level of estimation on D. Moreover, the process of evaluating the estimator is generated from the mean squared error (MSE) distance. This is done through taking the

expected value with respect to P which represents the unknown probability distribution of Y as follows:

$$E\left[\left(y - fD(x)\right)^2 \middle| x, D\right] \qquad (2.5)$$

Decomposing the above yields:

$$E[(y - fD(x))2|x, D] = E[(y - E[y|x])2|x, D] + E[(fD(x) - E[y|x])2] \qquad (2.6)$$

The first term is independent from either the training data D or the estimator fD as it computes the magnitude of noise or variability of y with respect to x. therefore, f can be estimated by:

$$E[(fD(x) - E[y|x])2] \qquad (2.7)$$

The conventional MSE of f is:

$$ED[(fD(x) - E[y|x])2] \qquad (2.8)$$

where ED yields expectations given all possible fixed sized training sets D. To further analyse the performance through MSE, decomposition of the error into variance and bias components is conducted in order to obtain the following:

$$ED[(fD(x) - E[y|x])2] = (ED[fD(x)] - E[y|x])^2 + ED[(fD(x) - ED[fD(x)])^2] \qquad (2.9)$$

The right hand side has two terms, the first one represents the bias while the second one represents the variance. In the case of training on a fixed training set D, reducing the bias with respect to the training set might lead to increasing the estimator's variance yielding low generalization performance. This technique is called "the trade-off between bias and variance". Basically, smoothing is used to reduce variance. On the other hand, this will lead to introducing bias resulting in for instance blurring sharp peaks. Reducing bias is done according to previous knowledge. When previous knowledge is employed for smoothing, it probably results in reducing the overall estimator's MSE.

In neural network training, the variance originates from two terms. The first one comes from "inherent data randomness" while the second one comes from the model's "non-identifiability" or in other words for a particular training data, there exists various local

minima of the error surface. As an example of an identifiable model is the "logistic regression".

The main point is to obtain the optimum training method for lessening the error of the particular ensemble average. Conventional training algorithms tend to decrease the error of the individual neural network for example the equation (2.9) complexity including variance and bias. Basically, the variance increases while the bias decreases with the continuous employment of several training instances. In addition, one will have to stop whenever their sum goes to a minimum. Since the predictor is an ensemble average in this algorithm, it has to seek different minima in order to eliminate some portion of the estimator's variance through ensemble averaging. However, it should be a point which has a smaller bias or longer training time to function as an alternative for ensemble predictor.

2.4.4 Majority Vote

Thanks to significant improvements and research, the use of ensemble classifiers has been widely spread and two main popular ensembles have shown prominent performance which are the boosting and the bagging methods (Kim, et al., 2011). These techniques employ re-weighted or re-sampled training sets obtained from the original data. After that, a learning algorithm is applied repeatedly for each training set.

Boosting was proposed as a technique for enhancing the performance of any weak learning algorithm which needs rather more than a random guess approach (Kim ,et al., 2011). Boosting changes the distribution of the training set adaptively depending on the performance of previous classifiers. To combine these classifiers in ensemble, boosting takes its turn in taking a weighted majority vote of their predictions. Bagging uses bootstrap samples in order to construct the classifiers in ensemble (Kim ,et al., 2011). Each sample is generated by random sampling provided by the same number of instances of the original data. The final output produced by the ensemble is obtained through simple majority voting. Other ensemble methods have also shown that the ensemble can do better than single predictors in most of the cases. Simple majority voting is a method that selects one of many solutions depending on the predicted class with the most votes (Kim ,et al., 2011). It is considered as the most used decision rule in ensemble methods. Weighted majority voting can be generated if each classifier's decision is multiplied by a weight to convey the confidence of each decision. Simple majority voting is considered as a special case of weighted majority voting. In this

method, a weight of 1/k is assigned to each classifier in which k corresponds to the number of classifiers in the ensemble.

Based on the reviewed existing classifier methods, various classification methods showed good results such as logistic regression ,neural network, neural fuzzy and combination methods, however the slow to train and demands systematic modification concerning feature implementation of logistic regression and although NF provides the best results when compared to other classifiers software but it might face a problem when complex and high dimensional data are input, however the ensemble combination methods by combining multiple classifiers together can provides improvements in classification prediction accuracy via minimizing the prediction error. In addition, multiple classifiers provide further accuracy than the original base classifiers that introduced them.

The ensemble combination method can further be enhanced by devising different types of neural networks, different models can be generated. Moreover, multiple initial weights are employed for accuracy enhancement. In order to handle high dimensional input data, different structures of ensemble methods and combination methods are employed.

2.5 Summary

This chapter carried out literature review on classifiers. Few of these classifiers are present within the toolbox of Matlab. Users can compare classifiers across various data sets through the Matlab classification toolbox. A list of functions for unsupervised and supervised classification algorithms is included in the Matlab classification toolbox. Classifiers studied critically are regression, neural networks, neuro-fuzzy systems and SVMs. Units (neurons) arranged in layers are comprised by a neural network. An input vector is converted by these neurons within the network into some output. An input is taken by each unit and a non-linear function is applied to it and then the output is passed on to the next layer. An extensive toolbox and library on neural networks is provided by Matlab. To support the development of a fuzzy system, the employment of strategies of heuristic learning driven from the theory of neural network is referred to as “neuro-fuzzy”. Also, in comparison to traditional empirical risk minimisation principles employed by most of neural networks, they have shown to be superior.

Chapter 3: Neural Networks

3.1 Introduction

An ANN can be defined as an information processing model in which its main motivation comes from how a nervous system, such as the brain, processes and analyses information. The main elements in this network are the architecture and the information-processing system. It consists of a large portion of complex intersected processing elements called neurons. ANNs tend to learn by configuring themselves on applications and feeding themselves with an enormous amount of rules and information that will further assist in pattern recognition or data classification to improve the network efficiency via a learning process. As a result, the programme can then learn how to respond to each input fed into it, and by then takes the appropriate measurements from data processing and pattern recognising. Nowadays, in biological systems, learning involves modifications in the synaptic connections between the neurons; the same process exists in ANNs (Stergiou and Siganos,2015; TechTarget, 2015).

A classical neural network contains from a few dozens, hundreds, even up to millions of artificial neurons known as “units”, classified in a series of layers where each layer is connected to the next one. The first layer is known as “input units” and this layer is designed to receive different types of information from an external source for the sake of performing the learning process. Figure 3.1 shows an example of a neural network (Stergiou and Siganos, 2015).

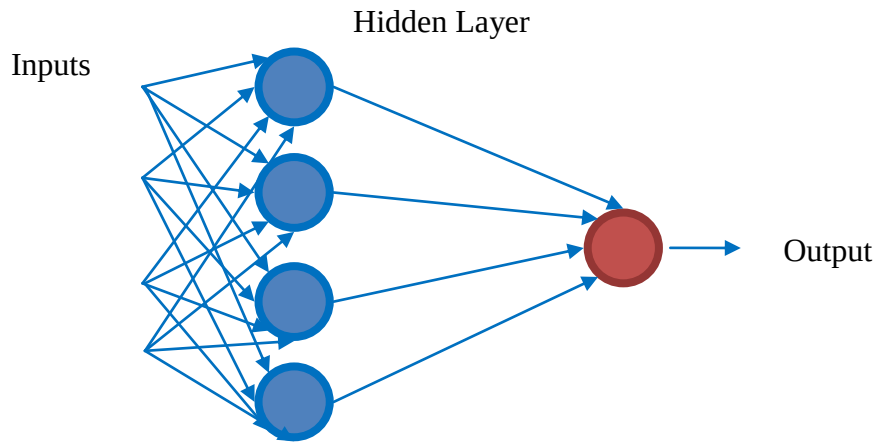


Figure 3.1: Example of a neural network

As for the "output units" layer, it is responsible for handling the learned information, after it passes through the "hidden units" which comprise of one or more layers and these units comprise the majority of an artificial brain. Most of the neural networks tend to be "fully connected", or, in other words, each hidden unit is connected to every output unit. This connection between one unit and another is known as a "value of weight" and is represented by a digit in which it can be either "positive" or "negative". The higher the weight value the more influence of projects unit on another (Stergiou and Siganos, 2015).

3.2 Overview of Neural Networks

Neural networks with their prominent capability of extracting meanings and patterns from complex inaccurate data, can be quite useful in pattern recognition and trend detection of data that are too difficult for humans and conventional computers to process.

A trained ANN is known as an "expert" in the aspect of data analysis. These "experts" have various advantages such as:

1. Adaptive learning: learning based on previous data training.
2. Providing projections to new areas of interest in addition to providing answers to "what if" questions.

3. Self-organisation: organising learned information on its own.
4. Real-time Operation: processing of data is done in parallel, in addition to hardware being designed which utilises this practical process.
5. Error Tolerance via Redundant Information Coding: it is possible to recover destruction of data despite the magnitude of network damage.

Moreover, nowadays in software processing of ANNs, the approach of simulation inspired by biology has been somewhat abandoned for a more systematic approach that is inspired by the basis of signal processing, probability and statistics. In addition, some of these systems build components in larger systems to combine both adaptive and non-adaptive elements. These new systems do not have much relevance to the traditional connectionist models of an artificial intelligent system, despite their real-world problem-solving applications. However, principles that are in common include non-linear, parallel, local and distributed adaptation and processing (Stergiou and Siganos, 2015).

3.3 Neural Network Models

Neural network models in AI are famously known as "ANNs" and go under the simple mathematical function of $f : X \rightarrow Y$ or a distribution over X or both $X \; and \; Y$. An ANN model is generally defined by three parameters which are: (Russel, 2015)

1. The pattern of interconnections between different neuron layers.
2. The process of learning through updating the weights of the interconnections.
3. The activation function f that processes the weighted input X into its output Y.

The models discussed in this chapter are as follows: Cascade-forward back propagation network, feedforward input time-delay back propagation network, fitting network, feedforward back propagation network, radial basis function network and layered-recurrent network (Shiffman, 2012).

3.4 Experimental Data

In this chapter, and specifically chapters 4, 5 and 6, the performances of the algorithms are analysed through experimentations including 7 representative datasets obtained from the University of California, Irvine (UCI) repository. These data sets were employed in similar studies as well. While assigning these datasets, binary and multi-class datasets should be

considerably included. Moreover, multiple number of attributes and data items should be considered as well.

There are up to 9 representative datasets which have been analyzed and 7 datasets applied in the algorithms. On the other hand, there were other datasets that were not included such as “Horse colic” which was missing some values and “Haber man” which was not practical when the previous methods were applied.

In this section, the 7 data sets are summarised. In addition, different classes of data sets are included in each data set such as binary, categorical and numerical classes. Each data set includes the number of instances used for the training. Furthermore, each data set contains a different number of attributes depending on the nature of the data set, therefore dependently setting the standards. There is also statement of loss of data in each data set as experienced and the cost matrix for each data set.

3.4.1. Johns Hopkins University Ionosphere Database

This data set was conducted in 1989. The source of data is Space Physics Group , Johns Hopkins University.

The nature of this dataset is the analysis of radar returns from the ionosphere layer by employing neural networks. In this dataset, the analysis was made based on using back propagation and perceptron training algorithm. 200 instances were used for training, where they were thoroughly divided into 50% positive and 50% negative.

The conclusion was that “linear” perceptron had 90.7% accuracy while a “non-linear” perceptron had 92% accuracy. On the other hand, back propagation had an average of up to 96% accuracy. Those were carried out on the remaining 150 instances, with 123 “good” and 24 “bad” instances. Accuracy on “good” instances was much greater than on “bad” instances. Going back to the back propagation, it was analysed with different numbers of hidden units, in other words, as a correspondence of the performance of the different variants of back propagation after a number of periods.

The radar readings were obtained from a system located in Goose Bay, Labrador. It comprised of 16 high frequency antennas in phased array possessing a transmitted power of up to 6.4 kW. The main goal was to obtain free electrons in the ionosphere layer.

The radar returns that conveyed "good" were the ones presenting evidence of some sort of structure in the layer. On the other hand, "bad" returns were the ones that did not, meaning their signals will protrude through the ionosphere.

Moreover, the received signals were handled by using an autocorrelation function in which its variables are the pulse time and pulse number. From Goose Bay systems, there were 17 pulse numbers. In this database, the instances are expressed using 2 attributes per pulse number, with each one yielding complex values obtained from the function through the complex electromagnetic signal.

The number of instances was 351. While the number of attributes was 34 in addition to the class attribute (all of the 34 attributes were continuous while the 35^{th} attribute was either "good" or "bad" depending on the definition mentioned above). Moreover, this is a binary kind of classification. In addition, the missing values were none.

3.4.2. Contraceptive Method Choice

This data set was conducted on June 7, 1997. The sources are:

(a) Origin: This dataset is a subset of the 1987 National Indonesia Contraceptive Prevalence Survey

(b) Creator: Tjen-Sien Lim

(c) Donor: Tjen-Sien Lim

The dataset is part of the 1987 National Indonesia Contraceptive Prevalence Survey. The samples for this data set were married women who were not pregnant or they did not know that they were pregnant at the period of the survey. The nature of this data set was to predict the used contraceptive methods: not using, long-term methods, or short-term methods. This prediction was based on the demographic and the socio-economic situation.

The number of instances was 1473 and the number of attributes was 10 including the class attribute. There were no missing attributes. The 10 attributes had numerical, categorical, and binary types of inputs, and the categorical inputs had a range of ratings depending on the type of attribute. The wife's age input is numerical. The education of the wife is a categorical input ranging from 1 to 3 for 1 being low in education. The same goes for the husband's education with the categorical range from 1 to 3. The number of children born is numerical.

The religion has a binary approach with 0 being a non-Muslim and 1 being a Muslim. The same approach goes for the status of employment for the wife, being 0 as employed and 1 as not employed. The occupation of the husband takes a categorical approach from 1 to 4, the same as the standard of living which is 1 for low and 4 for high standard of living. The exposure of media: 0 means having good exposure and 1 not having good exposure to media. The type of contraceptive methods used is a class attribute with 1 conveying no use, 2 long-term use and 3 short-term use.

3.4.3. Statlog (Heart) Data Set

This data set has 13 attributes extracted from a set of 75 attributes. The attribute information includes age, sex, type of chest pain, blood pressure (resting), serum cholesterol (mg/dl), blood sugar (fasting) (>12- mg/dl), electrocardiographic results (resting) (values 0, 1, 2), maximum achieved heart rate, exercise stimulated angina, old peak = (ST depression stimulated by exercise in relevance to rest), peak exercise slope ST segment, number of major vessels (0 – 3) coloured by fluoroscopy and Thal: 3= normal, 6= fixed defect, 7= reversible defect.

The types of attributes are real (1, 4, 5, 8, 10, 12), ordered (11), binary (2, 6, 9) and nominal (7, 3, 13).

The variables concerned are used to predict either the absence or presence of heart disease. There are no missing values and the number of observations is 270. The cost matrix is as follows (the rows correspond to the real values and the columns represent the predicted values):

	Absence	**Presence**
Absence	**0**	**1**
Presence	**5**	**0**

3.4.4. Statlog (German Credit Data) Data Set

The number of instances is 1000. Two data sets were provided. The number of attributes in "German" was 20, from which 7 are numerical and 13 are categorical. As for "German number", 24 numerical attributes were provided. The numerical values are: Attr. 2: duration (months), Attr. 5: credit amount, Attr. 8: instalment rate in percentage of disposable income, Attr. 11: present residence since (date), Attr. 13: age (years), Attr. 16: number of existing

credit cards in this bank and Attr. 18: number of people being liable to provide maintenance for.

As for the qualitative or categorical attributes, they are as follows: Attr. 1: the status of existing checking account which ranges from A11 to A14, A11 being less than 0 DM, A12 being between 0 and 200 DM, A13 being greater or equal to 200 DM (salary assignment for 1 year at least) and A14 having no checking account. Attr. 3 is the credit history. This attribute ranges into 5 sects from A30 to A34 having A30 corresponding to no credit taken and all were paid; A31 conveys all credits at this bank were paid; A32 says that existing credits were paid back until currently. As for A33, it mentioned that there was a delay in paying in the past and the last one A34 conveys the possibility of a critical account or other accounts existing.

Attribute 4 discusses the purpose. It ranges into 10 sub-attributes from A40 to A410. A40 is a new car, A41 is a used car, A42 is furniture or equipment, A43 is radio/TV, A44 is domestic appliances, A45 is repairs, A46 is education, A47 is vacation (does not exist), A48 is retaining, A49 is business and A410 is others.

Attribute 6 talks about savings account/bonds. It ranges from A61 to A65, A61 being less than 100 DM, A62 from 100 to 500 DM, A63 from 500 to 1000 DM, A64 more than 1000 DM and A65 having unknown or no savings account.

Attribute 9 discusses personal status and sex. From A91 to A95, A91 being a divorced/separated male, A91 is a divorced/separated/married female, A93 is a single male, A94 is a married/widowed male, and A95 being a single female.

Attribute 7 talks about present employment. A71 is unemployed, A72 is less than a year, A73 ranges from 1 to 4 years, A74 from 4 to 7 years, and A75 is more than 7 years. Attribute 10 talks about other debtors/ guarantors being A101 none, A102 co-applicant and A103 a guarantor.

Attribute 12 is property. A121 is real estate. A122, if not A121, is building society savings agreement or life insurance. A123, if not A121 or A122, car or other, not in Attr. 6. A124 is unknown or no property. Attribute 14 is other instalments plans which A141 is bank, A142 is stores and A143 is none.

Attribute 15 is housing: A151, A152 and A153 being rented, owned, and for free respectively. Attribute 17 discusses job: 171 being unemployed or unskilled with no residency. A172 is unskilled but resident, A173 is a skilled official employee and A174 is a management/self-employed/highly-qualified employee/officer.

Attribute 19 is having a telephone whether registered under the customer's name (A192) or not (A191). Attribute 20 being a foreign worker (A201) or not (A202).

The cost matrix is as follows:

	1= Good	**2= Bad**
1= Good	**0**	**1**
2= Bad	**5**	**0**

The rows correspond to the actual classification while the columns represent the predicted classification. It is worse to assign a customer as good when they are bad (5) and vice versa (1).

3.4.5. Australian Credit Approval Data Set

This data set is related to credit card applications. All attributes and values have been altered to vague symbols for the sake of protecting the secrecy of the information.

There is a variety of attributes: continuous, nominal with a small number of values, and nominal with a large number of values. In addition, there are also a few missing values. The number of instances was 690 and the number of attributes was 14 in addition to the class attributes.

From those 14, there are 6 numerical attributes and 8 categorical attributes. The attributes correspond to numerical and categorical. The categorical ones are A1: 0,1 (a, b), A4: 1,2,3 (p, g, gg), A5: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14 (ff, d, I, k, j, aa, m, c, w, e, q, r, cc, x), A6: 1, 2, 3, 4, 5, 6, 7, 8, 9 (ff, dd, j, bb, v, n, o, h, z), A8: 1, 0 (t, f), A9: 1, 0 (t, f), A11: 1, 0 (t, f) and A12: 1, 2, 3 (s, g, p). As for the continuous attributes, they are A2, A3, A7, A10, A13 and A14.

A15 is a class attribute of either 1 or 2 (+ or -).

About 37 cases (5%) had one or more missing values where both A1 and A2 had 12 missing ones. A4, A5 had 5 missing values. A6 and A6 had 9 missing values and A14 had 13 missing values.

These missing values were compensated by the mode of the "categorical" and the mean of the "continuous".

For the class distribution, class 2 is "+" with 307 instances (corresponding to 44.5%) while the "– " is with 383 (about 55.5%). There is no cost matrix in this data set.

3.4.6. Breast Cancer Wisconsin (Original)

This data set was conducted on 8th January 1991. Attributes from 2 to 10 are concerned with representing instances, each instance having one of the two class possibilities: benign or malignant.

The number of instances as of July 1992 had reached 699. The attributes are 10 in addition to the class attribute. The attribute information is: Sample Code number (ID number), clump thickness (1-10), uniformity of cell size (1-10), uniformity of cell shape (1-10), marginal adhesion (1-10), single epithelial cell size (1-10), bare nuclei (1-10), bland chromatin (1-10), normal nucleoli (1-10), mitoses (1-10) and class (2 = benign and 4 = malignant).

The number of missing attribute values is 16, denoted by "?".

As for the class distribution, benign attribute has 458 which is 65.5% and 241 for malignant which is 34.5%.

3.4.7. Bladder Cancer Data Set

The number of inputs is 4 which are tumour type, patient details, protein expression and DNA mutation. Tumour type is divided into TCC grade from 1 to 3, where 1= best and 3= worse (yielding more aggressive tumours) and TCC stage from 'a' to 4 where a = superficial and non-invasive, 1 = superficial and just invasive, a+1 can be considered together or separately and 2 – 4: invasive tumours.

The patient details include sex, age, smoking (cigarettes/day, cigarettes/year, packs/years), other non-bladder cancers and blood group. Protein expression is divided into three sub-inputs which are P53 (0+1 = normal, 2 = abnormal [mutated]), msh2 (0+1 = abnormal [loss], 2 = normal) and mlh1 (0+1 = abnormal [loss], 2 = normal).

DNA mutations are divided into 5 sub inputs which are bat 26 (5 loci have been analysed, bat 26, bat 25 etc.). Normally they are stable (S), bat 25 (in some types of cancer they are unstable [I]), mfd 15 (any tumour with only 1 stable marker [I] – the whole tumour can), apc (considered as unstable [I]) and d2s123.

The number of outputs is 2 which are actual tumour and behaviour. Actual tumour is basically how long it took to come back after the surgery in a month. As for the behaviour, there are a few criteria. These criteria are: no. of recurrent (for superficial TCC [a + 1] basically the number of times it came back), time of progression (when did it become more advanced), time to TCC specific death (when it did kill the patient [2-4 tumours], for superficial tumours [a +1] deaths rarely occur), time to non-TCC death (from other causes), status at 5 years (AV= alive and no tumour, AT= alive with tumour, DN = died not from TCC, DT = died from TCC) and Study ID number.

3.5 Data Modelling

There were multiple types of ANN models that has been employed according to their architecture or function. Each model is compatible for a certain method or application, therefore, it is not suitable for these methods. In addition, there were 12 models of ANN put to test, and they were Cascade-forward Back Propagation Network, Feedforward Input Time-delay Backdrop Network, Fitting Network, Feed-forward Back Propagation Network, Radial Basis Function Network, Layered-Recurrent Network, distributed time delay network, Elman back propagation network, Generalized regression network, feedforward back propagation network with feedback from O/P, Exact radial basis network and pattern recognition network present. As a result, 6 of these models Cascade-forward Back Propagation Network, Feedforward Input Time-delay Backdrop Network, Fitting Network, Feed-forward Back Propagation Network, Radial Basis Function Network and Layered-Recurrent Network present the most efficient results and therefore suitable for the methods, while others some were eliminated such as “Elman”, due to the fact that it takes a long time for training. Moreover, “Hopefield” contained no training syntax and while others that gave out multiple outputs, some are suitable for only one input and one output.

Each of the algorithms was analyzed through number of experimentations regarding its performance. There are up to 9 representative datasets which have been analyzed and 7

datasets applied in the algorithms. On the other hand, there were other datasets that were not included such as “Horse colic” which was missing some values and “Haber man” which was not practical when the previous methods were applied.

Sensitivity,specificity,accuracy, area under the curve (AUC) and root mean square (RMS) are the analysis parameters of the performance results of the models. Sensitivity and specificity are two statistical parameters that are concerned with performance and classification functions. Sensitivity in its turn measures the proportion of “positives” that are rightly recognized as such. On the other hand, specificity measures the amount of “negatives” that are rightly recognized as such (Altman and Bland, 1994). In statistics and engineering, accuracy is a measurement of the extent of “closeness” of a certain value to the true value (Berrizbeitia, 2016).

Receiver operating characteristic (ROC). ROC is a graphical representation that shows the performance of a classifier system and the discrimination threshold is varied. The curve is done through the plot of the true positive rate (sensitivity) and the false positive rate (specificity) (Berrizbeitia, 2016). Area under the curve(AUC) is defined informally as the signed area of the region in the xy-plane that is bounded by the graph of f, the x-axis and the vertical lines $x = a$ and $x = b$. The area above the x-axis adds to the total and that below the x-axis subtracts from the total. When imagining an xy plane, the area under the curve AUC is signed area of the xy region that is limited by the graph of the function, the x-axis and the vertical lines a and b. in addition, the area above the x-axis adds to the total while the area below the x-axis is subtracted from the total area . The root mean square (RMS) is the quadratic mean and is the square root of the arithmetic mean of squares of a set of values. In statistics, the RMS of an estimator measures the imperfection of the estimator’s fitness to the data(Wolfram MathWorld, 2016) .

The performance of these models is based on 80% network training and 20% network testing. Furthermore, for training and testing, different percentages for each proportion was made. For example, the ratio of 90% of training to 10% testing was made in addition to 70% training to 30% testing. However, the best performance was obtained from having 80% for training and 20% for testing, therefore we have utilized the 80 to 20 percentage as a default method throughout the research.

3.5.1 Cascade-forward Back Propagation Network

The cascade feedforward back propagation network is quite similar to the feedforward network. However, it includes a weight connection carried out from the input to each layer (output and hidden layers). This is shown in Figure 3.2. The neurons in the first layer, the input layer, have a single function of providing a buffer for spreading the input signals to the neurons in the hidden layer. Each neuron in the hidden layer sums its input signal, weights them, then computes its outputs. Training a network comprises of modifying its weights accordingly via learning algorithms (Godara and Gupta, 2012).

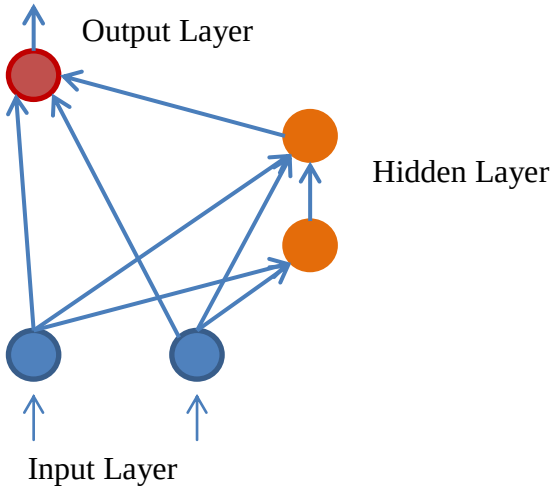


Figure 3.2: Cascade-forward back propagation network (Godara and Gupta, 2012)

Though a two-layer feedforward network has the ability to virtually learn any input-output relationship, multi-layer feedforward networks possess the capability of learning even the most complex relationships quickly (Dheeraj, et al., 2014).

To reach the optimised status in this algorithm, the Tangent-sigmoid transfer function is used. The performance (prediction accuracy) of each algorithm, the cascade forward back propagation and the feedforward back propagation can both be analysed via using Mean Square Error "MSE", Mean Absolute Error "MAE", Mean Relative Error "MRE" and Coefficient of Correlation "CC" (Dheeraj, et al., 2014).

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(observed - predicted)^2 \qquad (3.1)$$

$$MAE = \frac{1}{n}\sum_{i=1}^{n} |observed - predicted| \quad (3.2)$$

$$CC = \frac{\sum(x-x')(y-y')}{\sqrt{\sum(x-x')^z(y-y')^z}} \quad (3.3)$$

$$MRE = \frac{1}{n}\sum_{i=1}^{n} \frac{(observed-predicted)}{observed} \quad (3.4)$$

At which n corresponds to the number of data patterns for the independent data set, while x and y correspond to the sample means average (observed) and (predicted) while *x'* and *y'* are the averages of observed and predicted values (Dheeraj, et al., 2014). There are two training algorithms that can be used to update the weights of the network which are Levenberg-Marquardt and Bayesian regulation (Amiri and Esna, 2012).

Back propagation algorithms which are gradient-based are mostly used by researchers. The thing about them is that they are inefficient due to the fact that the gradient tends to vanish when reaching the solution. However, Hessian-based algorithms permit the network to learn trends of complex mapping which are more appropriate. As the solution is reached, the training process converges quickly due to the fact that the Hessian does not tend to vanish on reaching the solution. Moreover, the Levenberg-Marquardt (LM) algorithm is employed to benefit from the advantages of the Hessian dependent training (Amiri and Esna, 2012).

Bayesian Regulation algorithm is another training method for backpropagation, in which it begins by random distribution of initial weights and biases. After integrating the input patterns to the networks, updating the initial weight starts to reach final distribution. These methods are rather resilient to high noise level and possess good approximation with arbitrary accuracy of training and have the ability to enhance generalisation performance. Moreover, structural learning with forgetting is considered the main process used for regularisation. It possesses good approximation with arbitrary accuracy of training in addition to improving generalisation performance (Amiri and Esna, 2012).

This model was proved to provide excellent results and has been applied in different algorithms employed in this chapter. There are 6 network models in this thesis and each one will be applied on the 7 data sets. Based on that, the model(s) that give the best results will be concluded.

Tables 3.1 and 3.2 show the performance results of cascade-forward back propagation networks. The tables represents a summary of the data sets. Some criteria were taken in consideration when choosing the datasets. Binary, non-binary and multi-class datasets are included in addition to a variety in the number of attributes and data items.

In each table represented below, the row represents the 7 data sets which are: Breast Cancer, Bladder Cancer, Statlog (Heart), Contraceptive, German Credit and Australian Credit. In addition, the column represents the analysis parameters which are: Sensitivity, Specificity, Accuracy, AUC and RMS. Each cell contains a number which refers to a percentage (for the first three columns) while the other two columns represent measurements of area under the curve and root mean square. The performance of this model is based on 80% network training and 20% network testing. This division has been chosen as it delivers the best results.

Table 3.1: Cascade-forward back propagation Network (CFBP) training performance

	Sensitivity	**Specificity**	**Accuracy**	**AUC**	**RMS**
Breast Cancer	92.4528	99.6845	97.2689	0.99197	0.1984
Bladder Cancer	66.6667	98.7654	85.9259	0.9337	0.3240
Statlog (Heart)	78.3133	99.0385	89.8396	0.94158	0.3326
Contraceptive	95.7627	27.4376	66.5373	0.69419	0.5857
German Credit	76.8844	92.6148	88.1429	0.91382	0.3288
Australian Credit	92.1296	91.0798	91.6084	0.96488	0.2727
Ionosphere	92.8105	86.6667	90.535	0.92654	0.3390

Table 3.2: CFBP Network testing performance

	Sensitivity	**Specificity**	**Accuracy**	**AUC**	**RMS**
Breast Cancer	72.6027	98.4127	88.9447	0.97894	0.3056
Bladder Cancer	41.6667	67.6471	56.8966	0.59926	0.6282
Statlog (Heart)	55.8824	90.6977	75.3247	0.82064	0.4948
Contraceptive	94.0945	22.4599	63.7188	0.65916	0.6198
German Credit	57	80.303	72.4832	0.75123	0.4640
Australian Credit	87.9121	82.7957	85.3261	0.9063	0.3445
Ionosphere	95.7143	68	88.4211	0.86495	0.5533

Figures 3.3 to 3.9 are scatter plots representing the results that are shown in Tables 3.1 and 3.2. Each of the 7 data sets mentioned above have both trained and tested figures. These figures contain three plots: the actual predicted output, the rounded predicted output and Specificity vs. Sensitivity plots. According to these plots, the overall performance of the model will be compared to that of the other models to determine which one of them presents a better outcome. Relative to the results, the best prediction would yield a point in the upper left corner or coordinate (0,1) of the ROC space, representing both 100% sensitivity (no false negatives) and 100% specificity (no false positives). In addition, the (0,1) point is considered a perfect classification. In the ROC plot, when the curve is nearly touching the left top corner coming closer to 1, that means that this ROC enjoys high accuracy and when it goes down that corresponds to low accuracy ROC. Therefore, the more higher and closer to the corner the curve is, the better results are obtained in terms of accuracy.

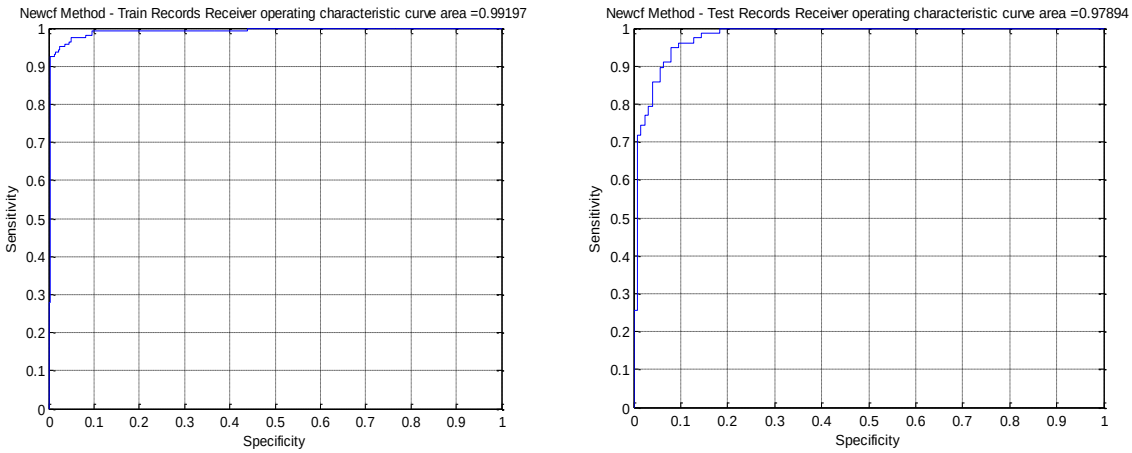


Figure 3.3: ROC training/testing of Breast cancer record

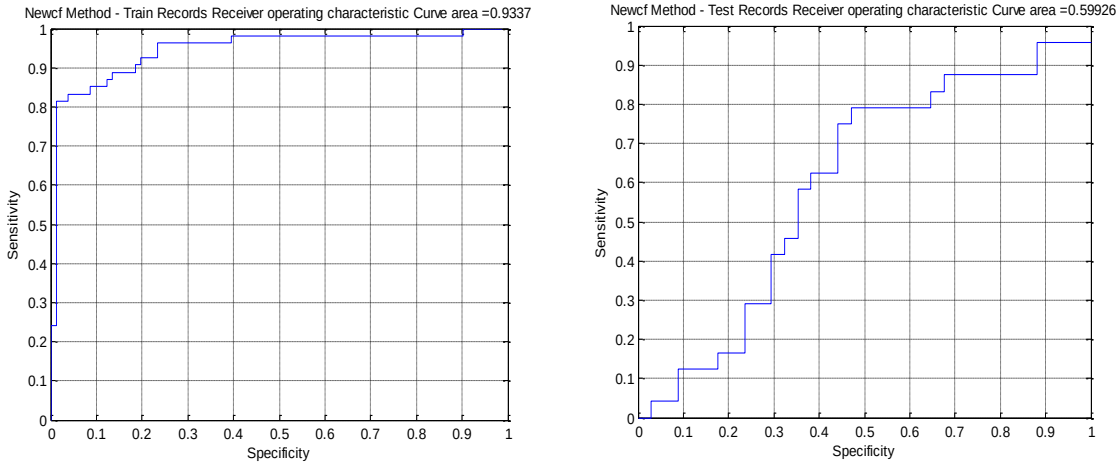


Figure 3.4: ROC training/testing of Bladder cancer record

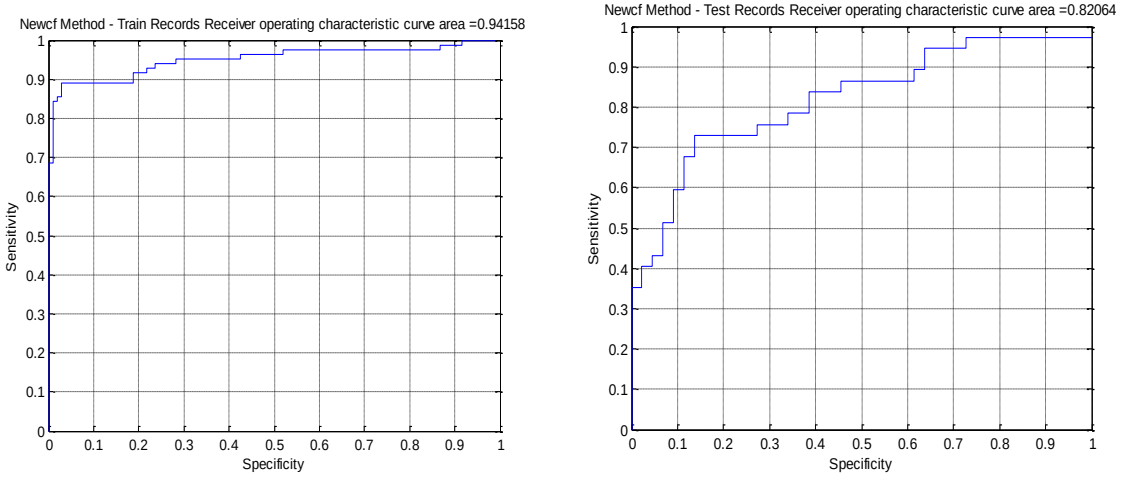


Figure 3.5: ROC training/testing of Statlog(Heart) record

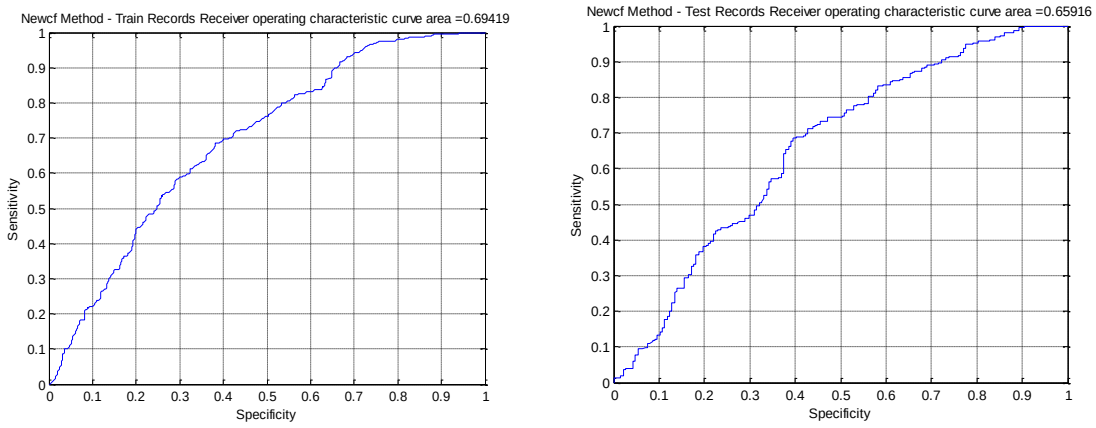


Figure 3.6: ROC training/testing of Contraceptive record

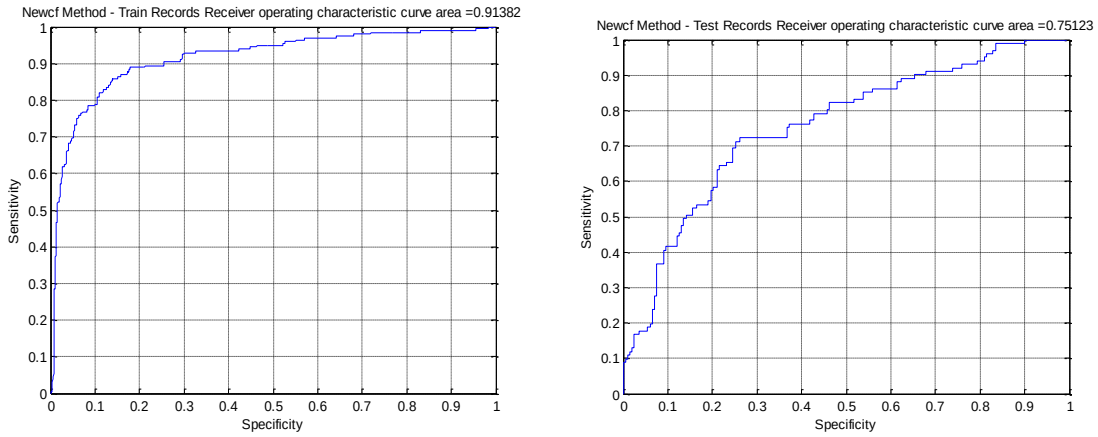


Figure 3.7: ROC training/testing of German record

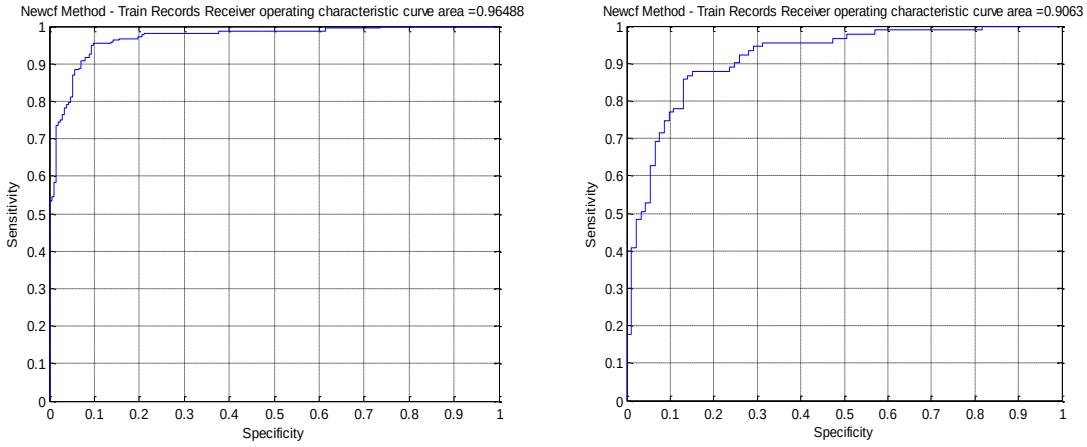


Figure 3.8: ROC training/testing of Australian record

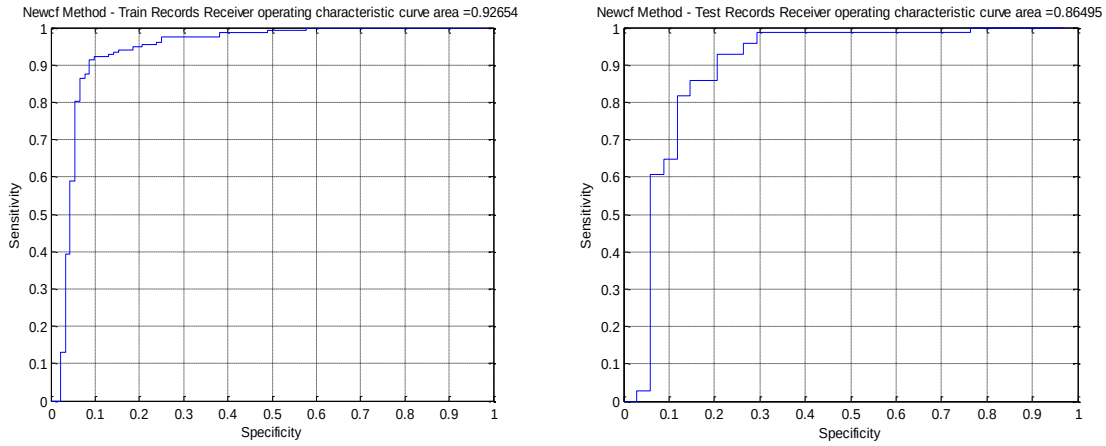


Figure 3.9: ROC training/testing of Ionosphere record

3.5.2 Feedforward Input Time-delay Backdrop Network

Time delay neural networks (TDNNs) can be defined as architectures that have a basic goal of operating on sequential data. The TDNN units have the ability to identify features without reliance on time-shift (sequence position) and they tend to structure part of a larger recognition system. This neural network is a time-shift invariant since it does not possess an internal state. TDNNs are similar to feedforward networks, however the input weight is associated with a tap delay line which allows the network to be equipped with a finite dynamic response to time-series input data. This kind of network comprises many elements. A "feature" is a component of the pattern that requires learning. The unit that is connected to this feature is known as a "feature unit". In the input layer, the number of the feature units is the same as that of the features. For the recognising patterns place or time-invariant, the previous values of activation and connection of the feature units have to be stored. This is done through making a copy of the feature units containing all their outgoing connections in each time step prior to updating the original units. A "delay" is known to be the total number of these time steps saved throughout this method (Mache, 1995).

The feature units and their delays are connected to the original units of the following layer. These units are known as the "receptive field". It is usually, though not necessarily, of the same number as the feature units. Moreover, the feature units might again split up between different receptive fields. Receptive fields might also overlap at the source plane, though it is necessary for them to cover all feature units. The length of the layer is known to be the total delay length. It is equal to the sum of the length of all the delays in the networks layers following the current one minus the number of subsequent layers. Each link in a receptive

field is recopied for every following step of time up to the total delay length. In the learning phase, these links are handled as single and modified depending on the average of those changes they would face when treated separately. The unit activation is performed by including the weighted sum of its input into an activation function (threshold or sigmoid kind of function). For TDNNs, this behaviour is altered via the delays. All the unit inputs are multiplied by the N delays. By structuring a whole network of time delay layers, the TDNN can connect inputs in different points in time or space (Mache, 1995).

Training in this type of network is done through a process similar to back-propagation. It includes particular semantics of coupled links in consideration. To allow this network to provide the preferred behaviour, there has to be a sequence of patterns to be presented to the input layer provided with the feature being shifted within the patterns. Since each feature unit is copied to each frame shift, the complete history of activations is accessible at once. Though the unit copies are purely duplicates seeking the same event, the weights of the corresponding connections between these copies have to be handled as one. First of all, a normal backpropagation method is done, then the error of the output layer is calculated. After that, the error deviations are processed and propagated backward. This results in different correction values for corresponding connections. Furthermore, all correction values for the corresponding links are then averaged and the weights are then updated with that value. This update in the algorithm imposes on the network to train on time and position independent detection of sub-patterns. This significant trait of a TDNN makes them independent from error-facing processing algorithms. The disadvantage however is that it results in a long and complex learning phase (Mache, 1995).

Theoretically, feedforward neural networks can have the chance to learn relationships depending on the mathematical function concerned. However, to model time-series or dynamic systems, memory is needed. One way to supply memory is through recurrent connections. A more practical method is through using a tapped delay line in the input time-series. However the delay line must be long enough to attain suitable performance. By increasing the number of input units, meaning longer delay lines, training times may increase due to the large portion of data needed. In addition, it is also important to choose the delay line length where the memory is provided by internal units (Obst and Riedmiller, 2013).

This network also was beneficial in obtaining great results when applied to the algorithms used in this chapter. Figure 3.10 shows the feedforward input time-delay system.

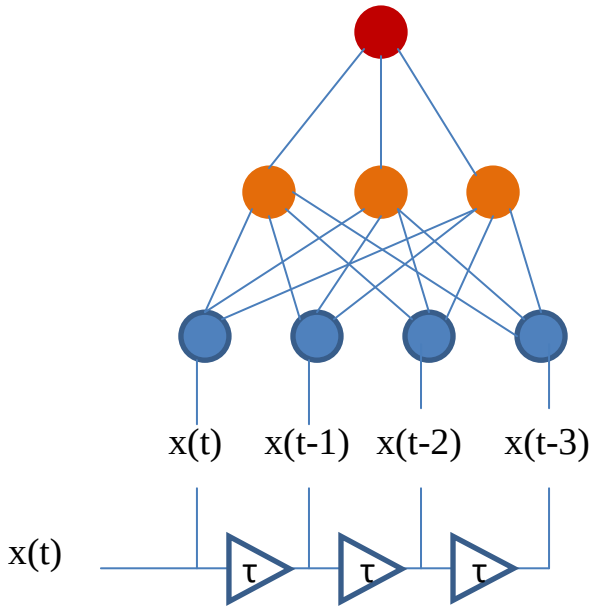


Figure 3.10: Feedforward input time-delay system (Obst and Riedmiller, 2013)

Tables 3.3 and 3.4 show the performance results of feedforward input time-delay backdrop network. The tables represents a summary of the data sets. Some criteria were taken in consideration when choosing the datasets. Binary, non-binary and multi-class datasets are included in addition to a variety in the number of attributes and data items.

In each table, the row represents the 7 data sets. In addition, the column represents the analysis parameters. Each cell contains a number which refers to a percentage (for the first three columns) while the other two columns represent measurements of area under the curve and root mean square. The performance of this model is based on 80% network training and 20% network testing. This division has been chosen as it delivers the best results.

Table 3.3: FFITBP Network training performance

	Sensitivity	Specificity	Accuracy	AUC	RMS
Breast Cancer	90	98.7382	95.8071	0.98364	0.2054
Bladder Cancer	76.9231	43.2099	56.391	0.67353	0.5195
Statlog (Heart)	79.5181	94.3396	87.8307	0.95044	0.3096
Contraceptive	93.2203	42.4036	71.484	0.76149	0.5095
German Credit	30.1508	96.4072	77.5714	0.82838	0.3925
Australian Credit	93.9535	88.7324	91.3551	0.2668	0.96411
Ionosphere	77.1242	60.6742	71.0744	0.78588	0.4533

Table 3.4: FFITBP Network testing performance

	Sensitivity	Specificity	Accuracy	AUC	RMS
Breast Cancer	81.8182	96.8254	91.133	0.96083	0.2718
Bladder Cancer	79.1667	52.9412	63.7931	0.72672	0.5151
Statlog (Heart)	75.6757	88.6364	82.716	0.91032	0.3438
Contraceptive	90.1575	39.0374	68.4807	0.71596	0.5319
German Credit	23.7624	95.9391	71.4765	0.77236	0.4434
Australian Credit	85.5556	76.3441	80.8743	0.86683	0.4022
Ionosphere	71.831	65.625	69.9029	0.71582	0.4975

Figures 3.11 to 3.17 are basically scatter plots representing the results that are shown in Tables 3.3 and 3.4. Each of the 7 data sets have both trained and tested figures. These figures contain three plots, the actual predicted output, the rounded predicted output and Specificity vs. Sensitivity plots. According to these plots, the overall performance of the model will be compared to that of the other models to determine which one of them presents a better outcome. Relative to the results, the best prediction would yield a point in the upper left

corner or coordinate (0,1) of the ROC space, representing both 100% sensitivity (no false negatives) and 100% specificity (no false positives). In addition, the (0,1) point is considered a perfect classification. In the ROC plot, when the curve is nearly touching the left top corner coming closer to 1, that means that this ROC enjoys high accuracy and when it goes down that corresponds to low accuracy ROC. Therefore, the more higher and closer to the corner the curve is, the better results are obtained in terms of accuracy.

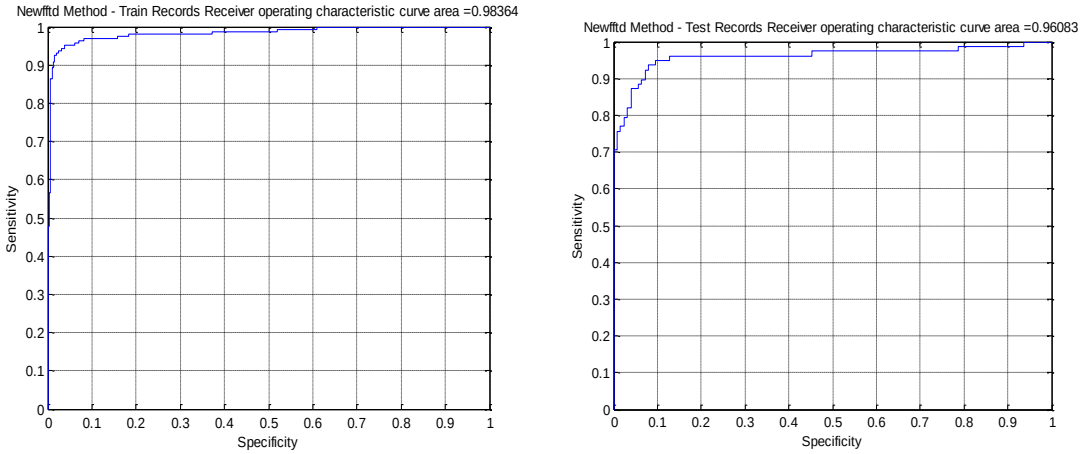


Figure 3.11: ROC training/testing of Breast cancer record

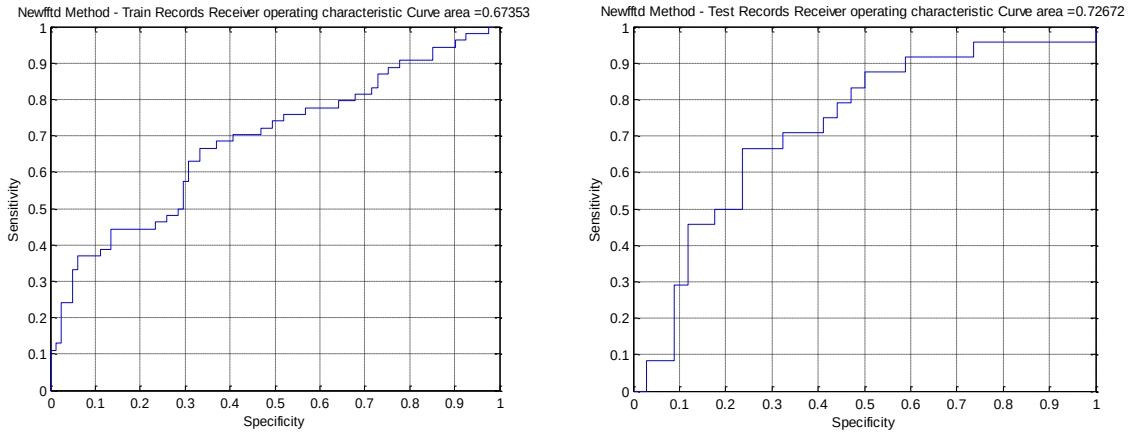


Figure 3.12: ROC training/testing of Bladder cancer record

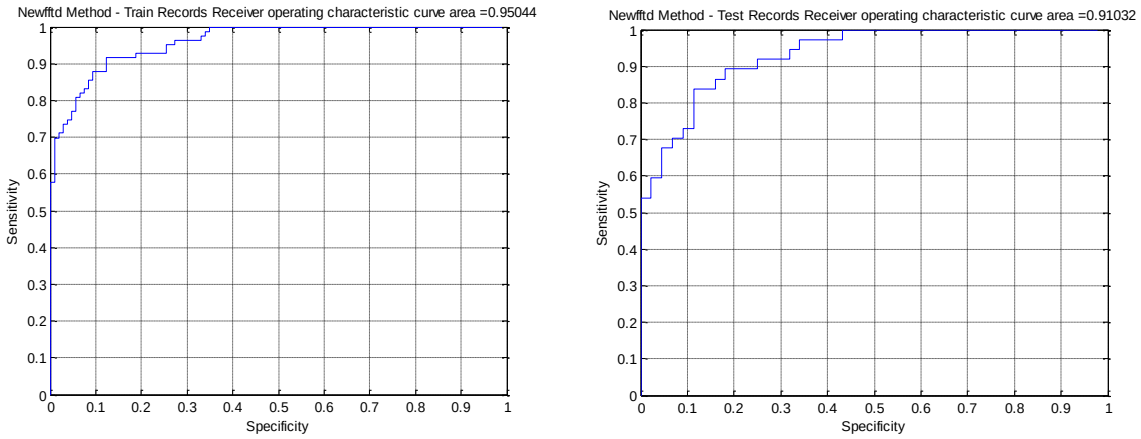


Figure 3.13: ROC training/testing of Statlog(Heart) record

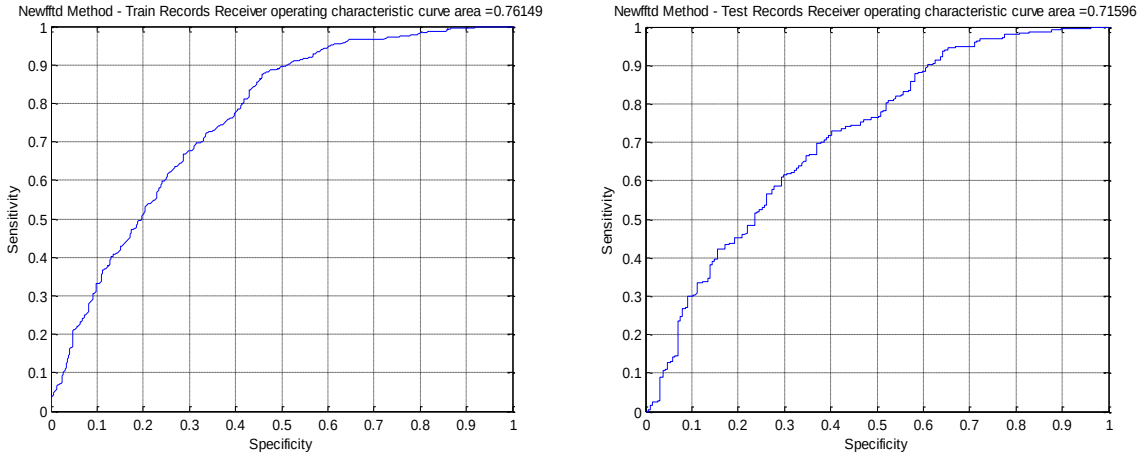


Figure 3.14: ROC training/testing of Contraceptive record

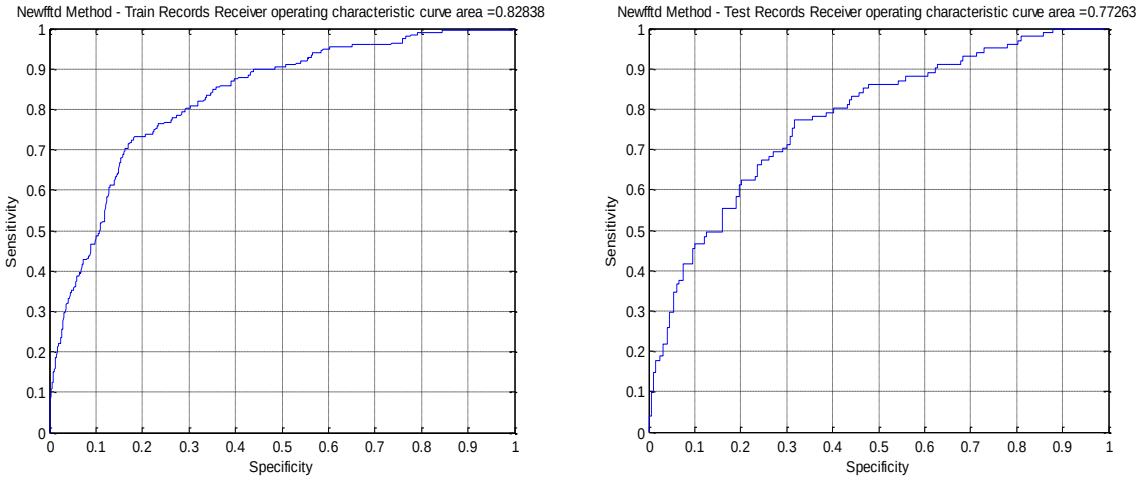


Figure 3.15: ROC training/testing of German record

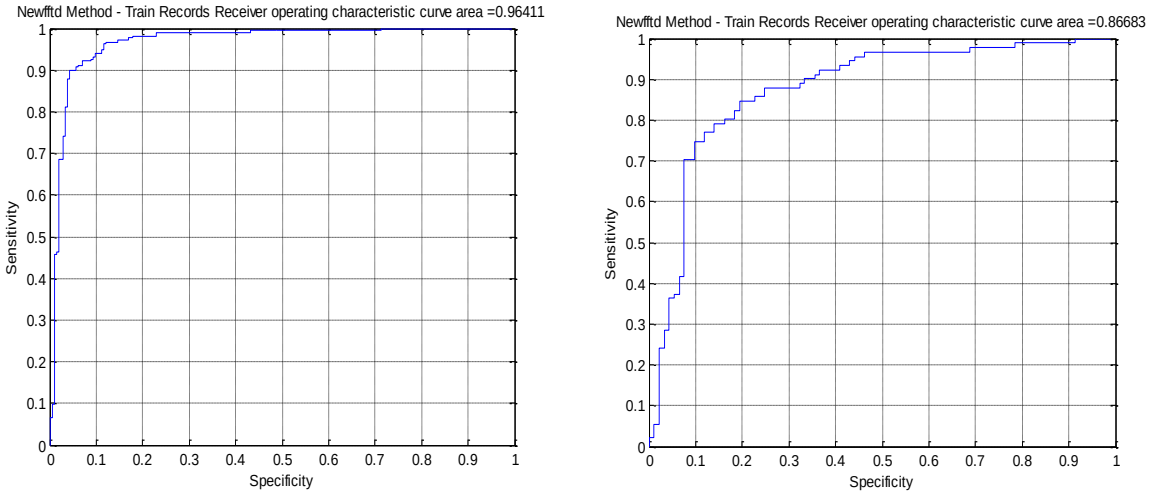


Figure 3.16: ROC training/testing of Australian record

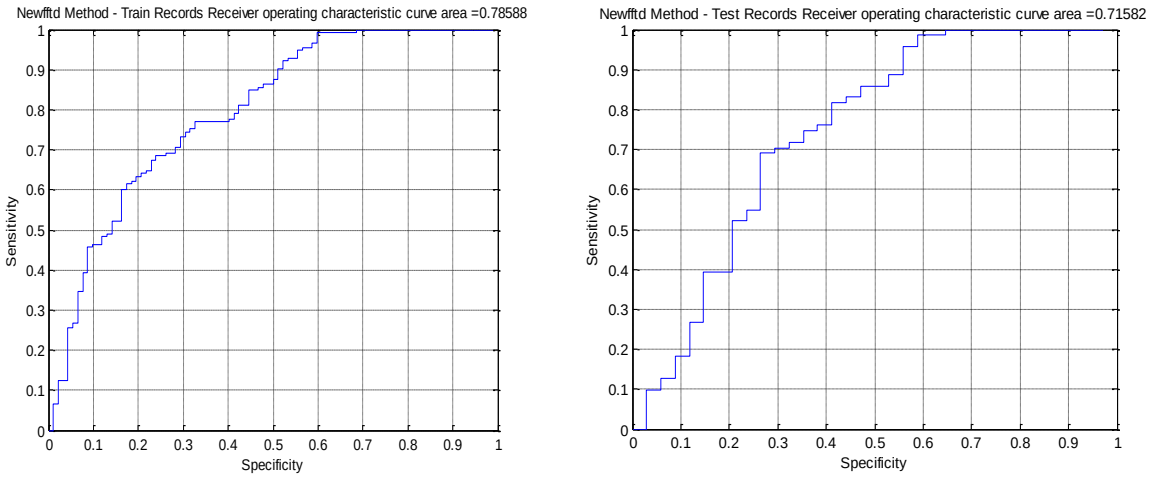


Figure 3.17: ROC training/testing of Ionosphere record

3.5.3 Fitting Network

Fitting networks are feedforward networks that are used to fit an input to output relationship. Neural networks are practical at fitting functions and pattern recognition. One of the main fitting networks being used in artificial intelligent (AI) is curve fitting networks. Curve fitting is known to be "the fitting of parameterised functional forms through sets of experimental data". Curve fitting issues represent a chance for the neural network to provide an arbitrary input to output relationship. Once this has been accurately structured, it can be employed for many tasks (series prediction, function optimisation and approximation). Function approximation is known to be the process of training a neural network by a given set of input-output data through supervised learning for the sake of deducing the relationship between

input and output. After training is done, this network can be used as a black box with an input to output characteristic almost equal to the relation of the training problem (Adrignola, 2010).

Curve fitting works best when using neural networks because it is generally much faster than conventional iterative approaches. In addition to that, more improvements in speed can be attained by employing special purpose hardware for the network, making this method practical for real-time applications. The optimum parameters values can be obtained through minimising an error measure, often chosen to be the sum of the squares of the errors that are between the observed values and the predicted values (by the function). If the functional form is dependent linearly on the parameters (polynomial), this results in an easy-to- solve linear minimisation problem. In many situations, it is important to consider functional forms which rely nonlinearly on the unknown parameters. In such situations, the minimisation process basically includes an iterative algorithm with an initial guess. However, these processes are computationally thorough and thus they are slow, and for complicated problems the necessity for an appropriate initial guess requires human assistance to guarantee convergence to the correct solution (Bishop and Roach, 1992).

For applications requiring large amounts of data and applications that are real-time, there is a significant interest in techniques which can automate the process of curve fitting in addition to operating at high speed. The issue of parameters optimisation of a provided functional form so as to fit experimental data is faced often in data analysis. Mostly, there is a prominent interest in non-linear curve fitting techniques due to their fast and automatic direct mapping with the best fit function. Although the training of these networks requires a lot of hard work, the trained network is capable of rapid data processing. Generally, they are much faster than other iterative methods (Bishop M and Roach, 1992).

The feedforward neural networks (feedforward net) are used to fit an input-output relationship. The fitting network provided great results when applied on the algorithms in this chapter. Figure 3.18 shows a fitting network.

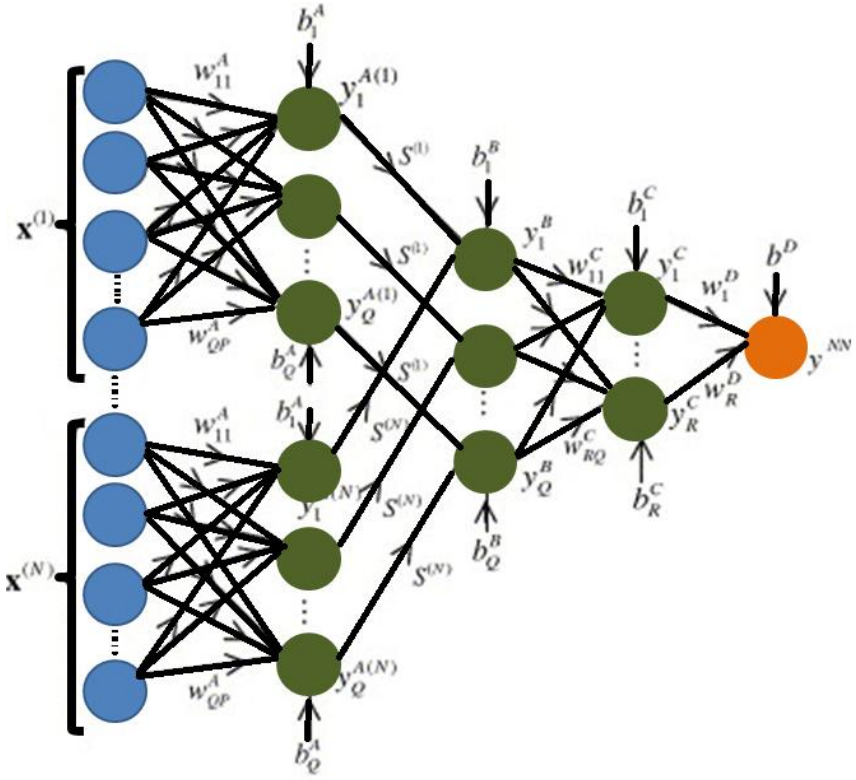


Figure 3.18: Fitting Network (Bishop M and Roach, 1992)

Tables 3.5 and 3.6 show the performance results of a fitting network. The tables represents a summary of the data sets. Some criteria were taken in consideration when choosing the datasets. Binary, non-binary and multi-class datasets are included in addition to a variety in the number of attributes and data items.

In each table, the row represents the 7 data sets. In addition, the column represents the analysis parameters. Each cell contains a number which refers to a percentage (for the first three columns) while the other two columns represent measurements of area under the curve and root mean square. The performance of this model is based on 80% network training and 20% network testing. This division has been chosen as it delivers the best results

.

Table 3.5: FITT Network training performance

	Sensitivity	Specificity	Accuracy	AUC	RMS
Breast Cancer	96.2264	100	98.7395	0.99422	0.1491
Bladder Cancer	42.3077	91.358	72.1805	0.78624	0.4842
Statlog (Heart)	89.0244	87.3786	88.1081	0.3480	0.92942
Contraceptive	96.6102	19.5011	63.6275	0.70272	0.5734
German Credit	26.1307	96.2076	76.2857	0.75409	0.4156
Australian Credit	90.2778	86.385	88.345	0.93762	0.3157
Ionosphere	88.2353	96.7033	91.3934	0.96839	0.3422

Table 3.6: FITT Network testing performance

	Sensitivity	Specificity	Accuracy	AUC	RMS
Breast Cancer	72.0588	97.619	88.6598	0.97029	0.3830
Bladder Cancer	21.7391	72.7273	51.7857	0.56863	0.6530
Statlog (Heart)	72.2222	80.9524	76.9231	0.81388	0.4542
Contraceptive	94.8819	14.4385	60.771	0.66878	0.6084
German Credit	20.7921	93.4673	69	0.6976	0.4620
Australian Credit	91.2088	78.4946	84.7826	0.90571	0.3497
Ionosphere	80.2817	78.125	79.6117	0.84051	0.4241

Figures 3.19 to 3.25 are scatter plots representing the results that are shown in Tables 3.5 and 3.6 above. Each of the 7 data sets mentioned above have both trained and tested figures. These figures contain three plots: the actual predicted output, the rounded predicted output and Specificity vs. Sensitivity plots. According to these plots, the overall performance of the model will be compared to that of the other models to determine which one of them presents a better outcome. Relative to the results, the best prediction would yield a point in the upper

left corner or coordinate (0,1) of the ROC space, representing both 100% sensitivity (no false negatives) and 100% specificity (no false positives). In addition, the (0,1) point is considered a perfect classification. In the ROC plot, when the curve is nearly touching the left top corner coming closer to 1, that means that this ROC enjoys high accuracy and when it goes down that corresponds to low accuracy ROC. Therefore, the more higher and closer to the corner the curve is, the better results are obtained in terms of accuracy.

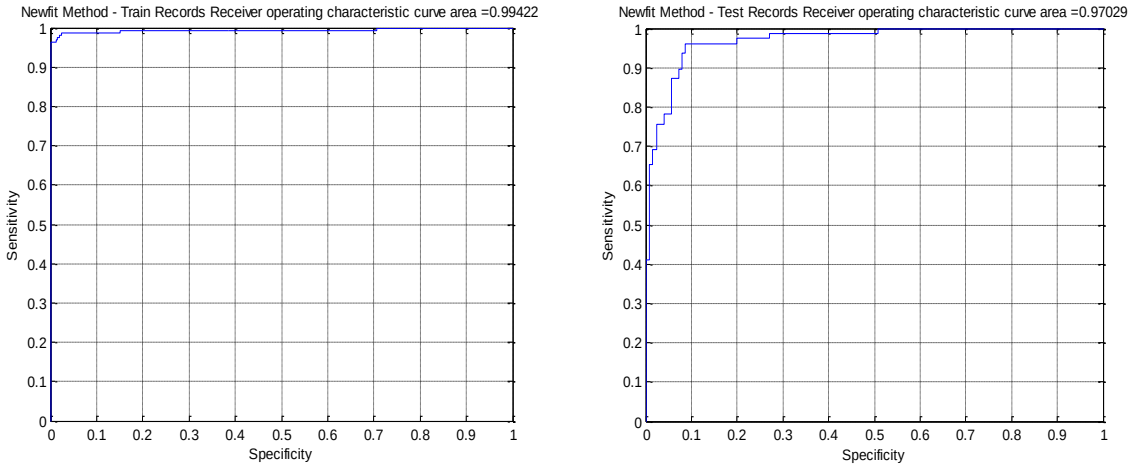


Figure 3.19: ROC training/testing of Breast cancer record

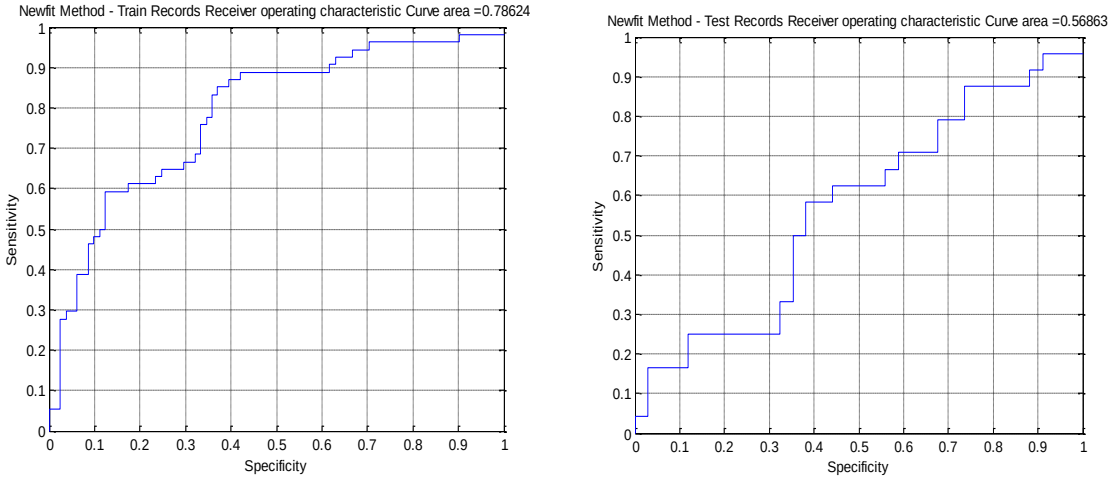


Figure 3.20: ROC training/testing of Bladder cancer record

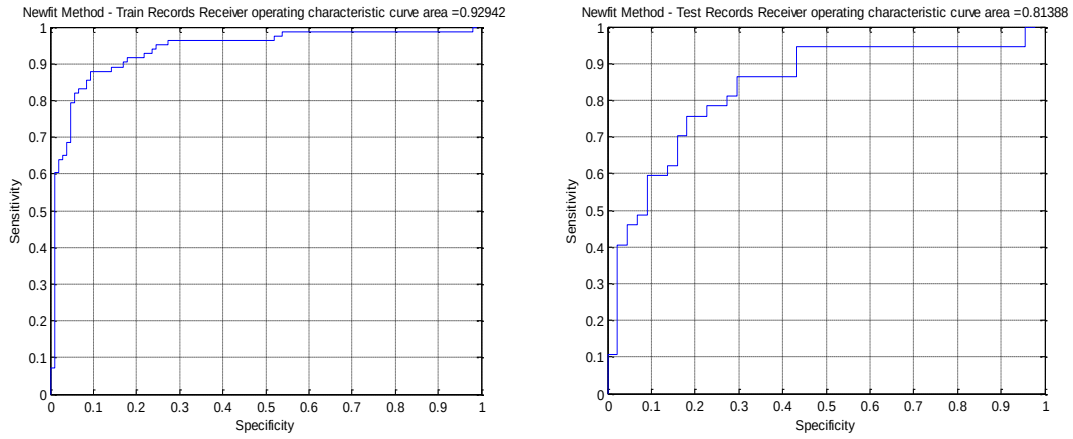


Figure 3.21: ROC training/testing of Statlog(heart) record

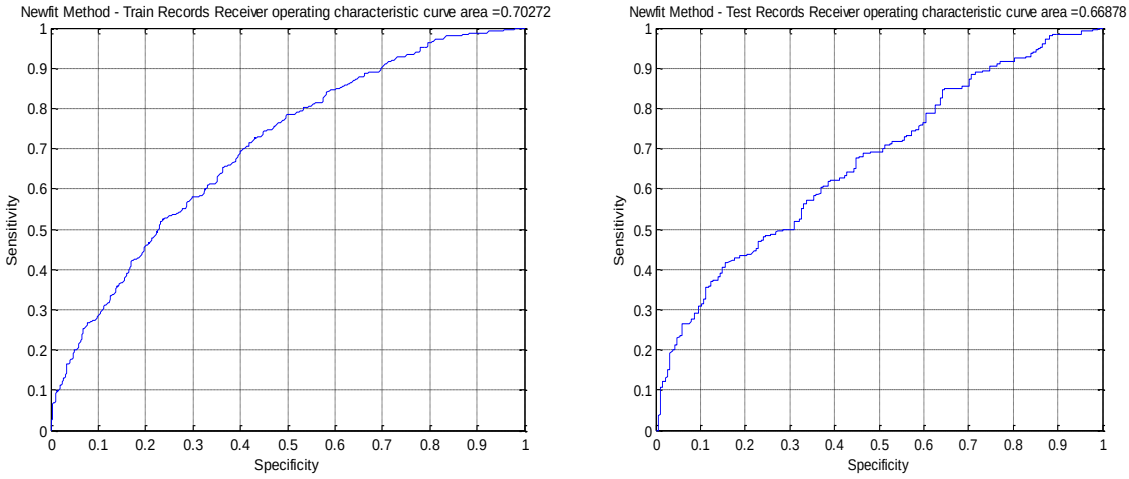


Figure 3.22: ROC training/testing of Contraceptive record

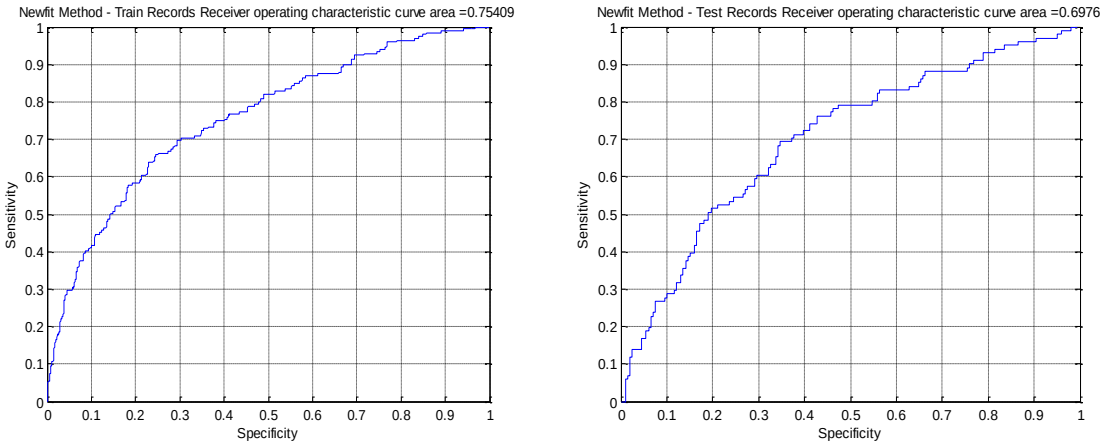


Figure 3.23: ROC training/testing of German record

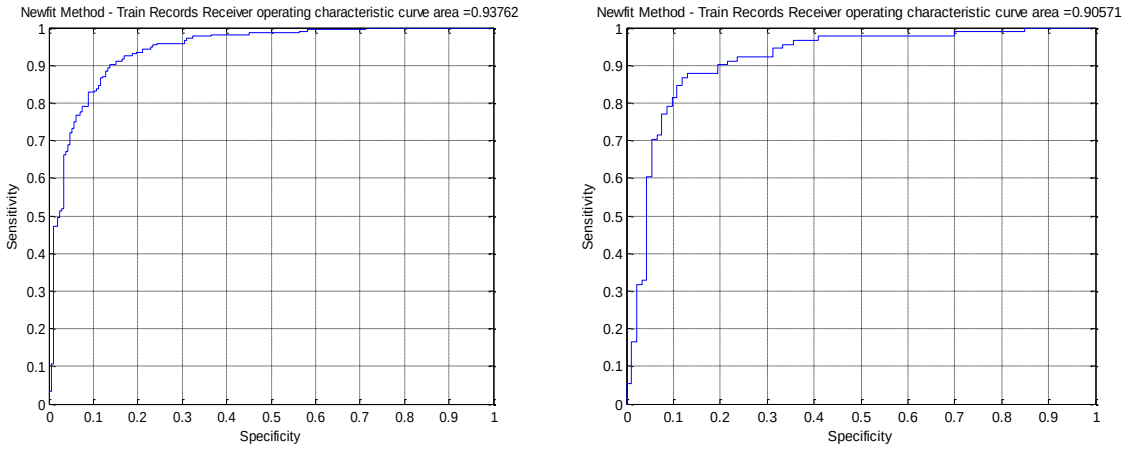


Figure 3.24: ROC training/testing of Australian record

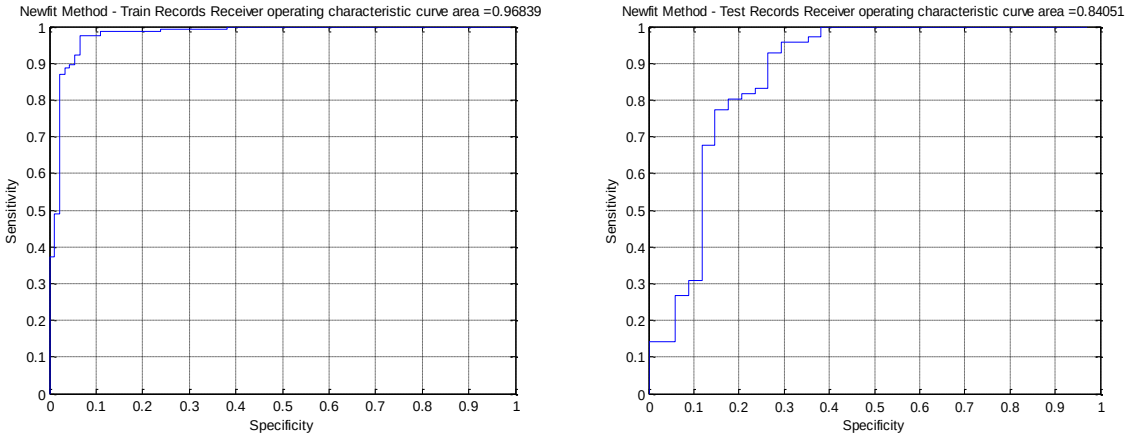


Figure 3.25: ROC training/testing of Ionosphere record

3.5.4 Feed-forward Back Propagation Network

This model is considered as the most common neural network architecture in machine learning. This popularity is due to its applications in many tasks. To be familiarised with this architecture, one must analyse the principle behind its training and pattern processing. This is shown in Figure 3.26. The first term of the model, the feedforward algorithm, illustrates how this network processes and remembers patterns. In addition, neurons are connected forward only. Each layer possesses connections to the next layer e.g. from input layer to the hidden layer, with no back connections (Heaton, 2008).

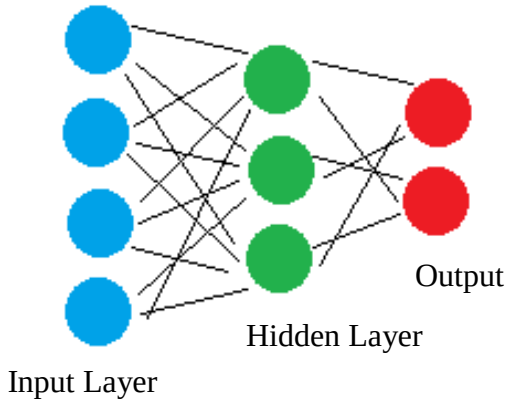


Figure 3.26: Feedforward back propagation network (Heaton, 2008)

The term ‘back propagation algorithm’ illustrates how this neural network is trained. Back propagation is a type of supervised training in which the network must be provided with sample inputs and predicted outputs. The predicted outputs are then compared to the actual output for the sake of the given inputs. Furthermore, the back propagation algorithm calculates the error and modifies the weights of the layers from the output to the input layer (backwards) (Heaton, 2008).

The Feedforward Back Propagation algorithm comprises of one input layer, one output layer and one or more hidden layers as shown in Figure 3.27. For the sake of the learning method, back propagation is required as discussed above. For the back propagation, the output layer weights are updated. There is a predefined value for each neuron in the output layer. The weights updated are from the predefined values of the neurons in addition to the learning rules. This algorithm is practical for sequential problems, however when it comes to other problems, it provides inaccurate results. In addition, through some cases, the learning process was hindered because of local minimum value. This case occurs due to positioning the answers at the smooth part of the threshold function. In the training of such a network, calculations were conducted from the input of the network to the output, and error values then were propagated to previous layers. The output calculations however were carried out layer to layer for the sake of making the output of each layer to be the input of the next one. Basically, these two algorithms work side-by-side together without any mean. In other words, the feedforward algorithm is applied to determine the output and there is no need for back propagation, and the same applies for the back propagation to determine the output without the necessity of the feedforward when structuring a neural network. However this is a special case of that neural network (Amiri and Esna, 2012).

As for the feedforward algorithm, they comprise of one or more hidden layers of sigmoid neurons followed by an output layer of linear neurons. Several layers of neurons with non-linear transfer functions permit the network to learn both linear and non-linear relationships between the input and the output vectors. Then, the output of a network produces values between 0 and 1, and after that the output layer employs a sigmoid transfer function. Figure 3.27 shows a feedforward back propagation network.

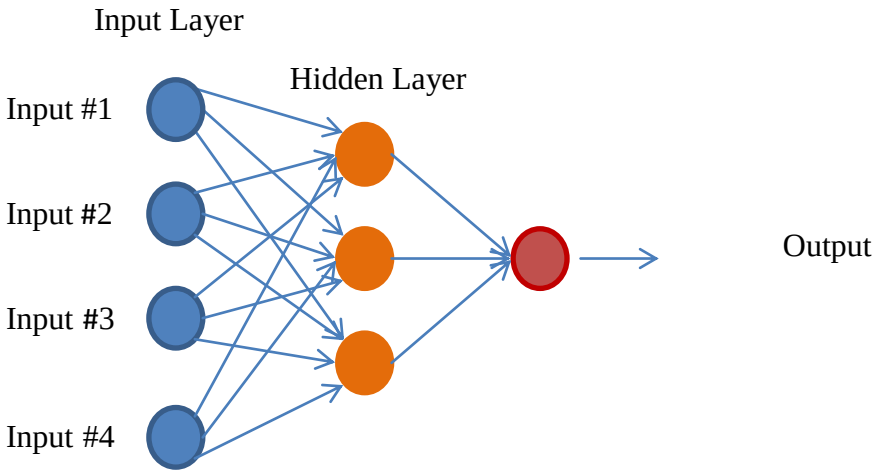


Figure 3.27: An example of a feedforward back propagation network (Amiri and Esna, 2012).

Tables 3.7 and 3.8 show the performance results of feedforward back propagation networks respectively. The tables represents a summary of the data sets. Some criteria were taken in consideration when choosing the datasets. Binary, non-binary and multi-class datasets are included in addition to a variety in the number of attributes and data items.

In each table, the row represents the 7 data sets. In addition, the column represents the analysis parameters. Each cell contains a number which refers to a percentage (for the first three columns) while the other two columns represent measurements of area under the curve and root mean square. The performance of this model is based on 80% network training and 20% network testing.This division has been chosen as it delivers the best results.

Table3.7: FFB Network training performance

	Sensitivity	Specificity	Accuracy	AUC	RMS
Breast Cancer	90.625	99.6845	96.6457	0.98864	0.1796
Bladder Cancer	85.1852	88.8889	87.4074	0.9209	0.3117
Statlog (Heart)	80.7229	92.4528	87.3016	0.92464	0.3320
Contraceptive	93.2203	39.229	70.1261	0.74158	0.5229
German Credit	59.799	93.4132	83.8571	0.88295	0.3515
Australian Credit	87.037	90.6103	88.8112	0.94829	0.2951
Ionosphere	88.2353	83.3333	86.4198	0.89336	0.3633

Table 3.8: FFB Network testing performance

	Sensitivity	Specificity	Accuracy	AUC	RMS
Breast Cancer	85.1351	98.4127	93.5	0.98952	0.2543
Bladder Cancer	47.619	78.7879	66.6667	0.69118	0.6208
Statlog (Heart)	62.1622	85.7143	74.6835	0.78808	0.4608
Contraceptive	90.1575	34.4086	66.5909	0.67097	0.5607
German Credit	41.5842	89.4472	73.3333	0.75541	0.4375
Australian Credit	89.011	86.0215	87.5	0.92887	0.3151
Ionosphere	78.8732	72.7273	76.9231	0.8169	0.4107

Figures 3.28 to 3.34 are scatter plots representing the results that are shown in Tables 3.7 and 3.8. Each of the 7 data sets have both trained and tested figures. These figures contain three plots: the actual predicted output, the rounded predicted output and Specificity vs. Sensitivity plots. According to these plots, the overall performance of the model will be compared to that of the other models to determine which one of them presents a better outcome. Relative to the results, the best prediction would yield a point in the upper left corner or coordinate (0,1) of

the ROC space, representing both 100% sensitivity (no false negatives) and 100% specificity (no false positives). In addition, the (0,1) point is considered a perfect classification. In the ROC plot, when the curve is nearly touching the left top corner coming closer to 1, that means that this ROC enjoys high accuracy and when it goes down that corresponds to low accuracy ROC. Therefore, the more higher and closer to the corner the curve is, the better results are obtained in terms of accuracy.

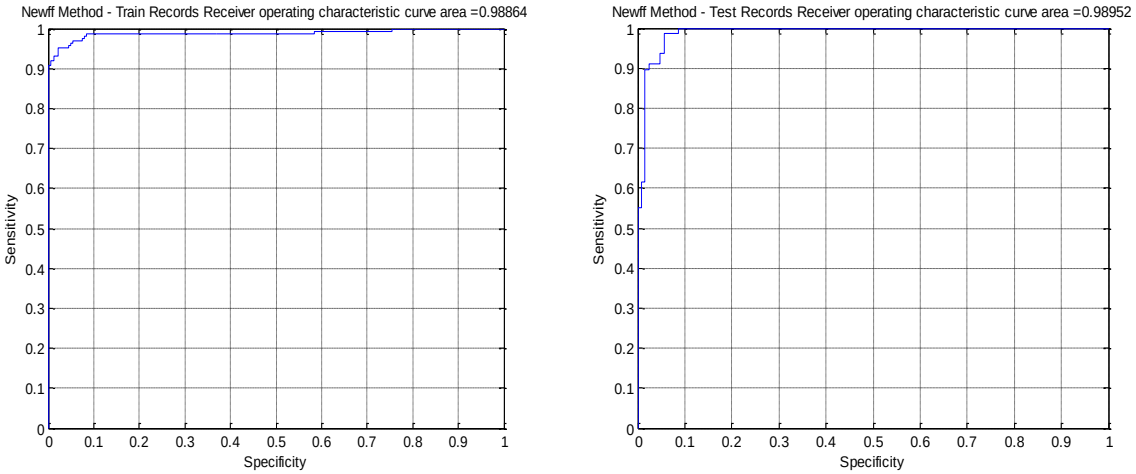


Figure 3.28: ROC training/testing of Breast cancer record

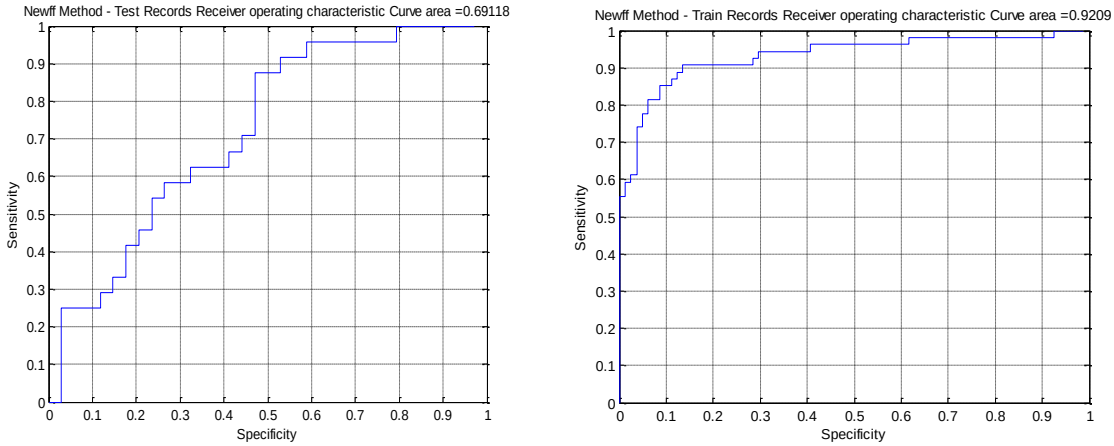


Figure 3.29: ROC training/testing of Bladder cancer record

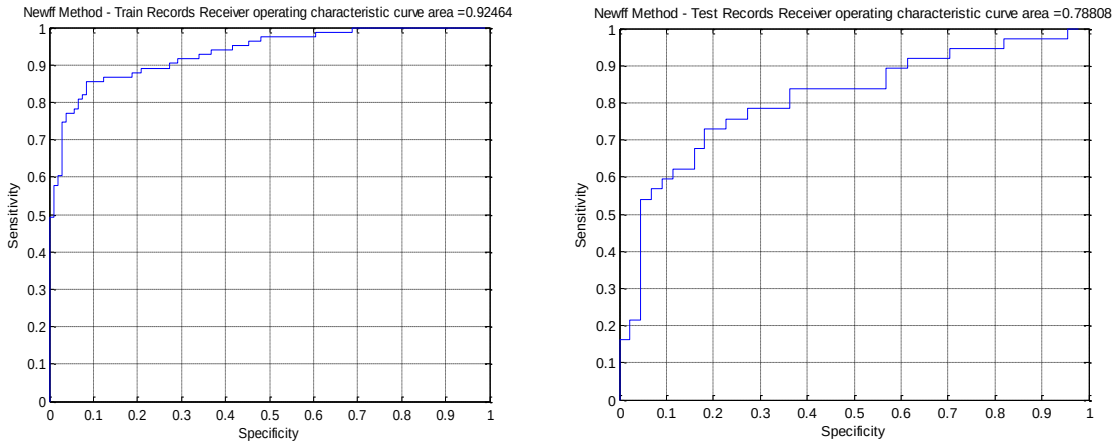


Figure 3.30: ROC training/testing of Statlog(Heart) record

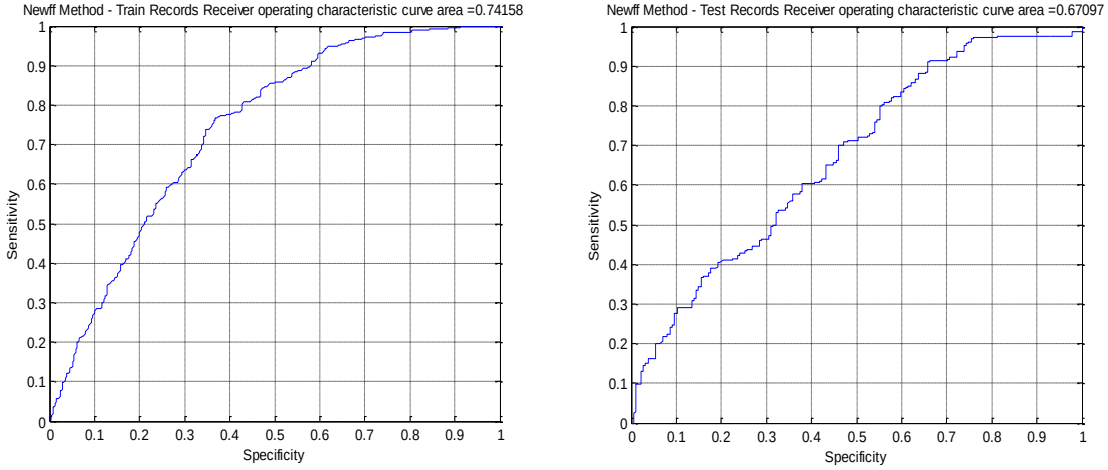


Figure 3.31: ROC training/testing of Contraceptive record

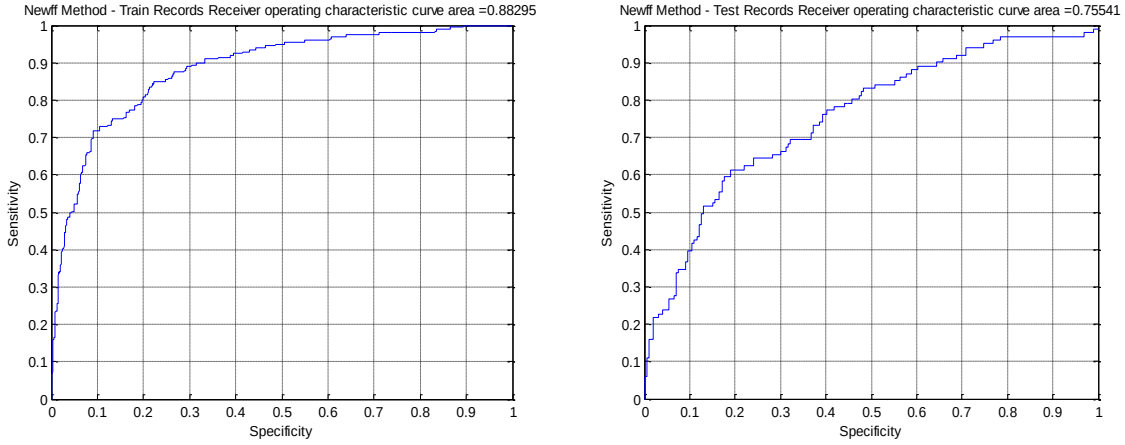


Figure 3.32: ROC training/testing of German record

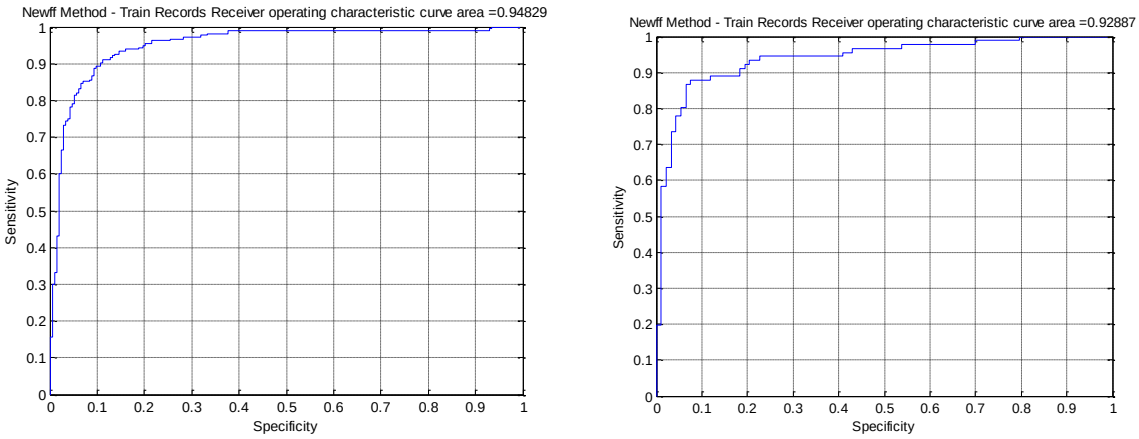


Figure 3.33: ROC training/testing of Australian record

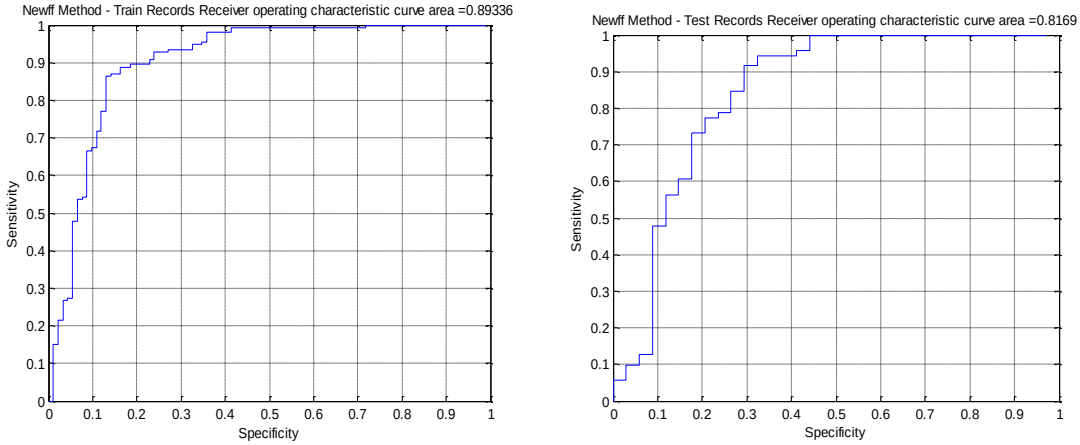


Figure 3.34: ROC training/testing of Ionosphere record

3.5.5 Radial Basis Function Network

A radial basis function (RBF) network is defined as an ANN that applies radial basis functions as activation function. The network output is basically a linear combination of the inputs and neuron parameters in radial basis functions. These networks have many applications such as function approximation, classification, time-series prediction and system control. They were first introduced in a 1988 paper written by Broomhead and Lowe at the Royal Signals and Radar Establishment. Moreover, radial basis functions are a special kind of function. The main feature they possess is that the response of the function decreases or in some cases increases monotonically according to the distance from a certain centre point. The parameters of the model are the centre, the distance and the shape of the radial basis function. These parameters are fixed if the function is linear (Orr, 1996).

Basically, they can be applied in any model either linear or non-linear and any kind of network either single-layer or multi-layer. These networks have been used in association with radial basis functions in single-layer networks. This is illustrated in Figure 3.35.

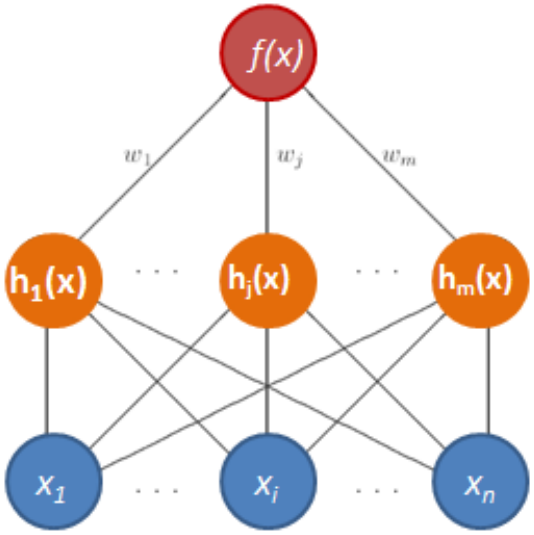


Figure 3.35: Radial Basis Function Network (Orr, 1996)

This figure illustrates the typical radial basis function network. Each *n* component of the input vector *x* feeds forward to basis functions *m* where its outputs are combined linearly into the network output *f(x)* with weights $\{w\}_{j=1}^{m}$ (Orr, 1996).

Figure 3.35 further explains an RBF network. Its inputs are x_1, x_2, ... x_n and the output is $\hat{y}$. The pointers in the figure represent the parameters in the networks. This network comprises of one hidden layer of neurons. At each neuron's input, the distance between the input vector and the neuron centre is calculated. The neuron output is then formed by implementing the basis function to that distance. The output of the network is structured through a weighted sum of the neuron's outputs and the unity bias is represented. This system is often matched with a linear part which is basically additional connections from the inputs to the neuron output (Wolfram, 2015). Single output RBN illustrated in Figure 3.36, this is one of the most used network models after the feedforward network.

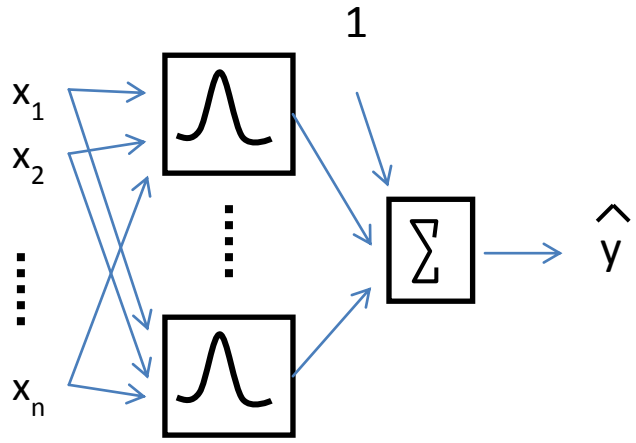


Figure 3.36: Single Output RBN (Wolfram, 2015)

Moreover, RBF networks can have multiple outputs, shown in Figure 3.37.

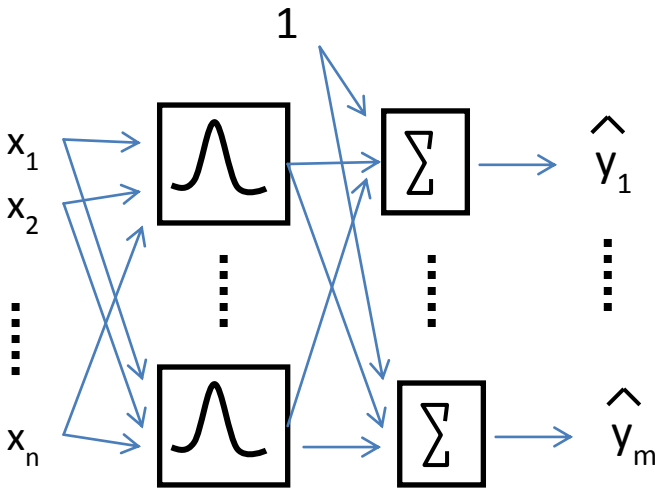


Figure 3.37: Multiple Output Radial Bases Network (Wolfram, 2015)

Tables 3.9 and 3.10 show the performance results of radial basis function network. The tables represents a summary of the data sets. Some criteria were taken in consideration when choosing the datasets. Binary, non-binary and multi-class datasets are included in addition to a variety in the number of attributes and data items.

In each table, the row represents the 7 data sets. In addition, the column represents the analysis parameters. Each cell contains a number which refers to a percentage (for the first three columns) while the other two columns represent measurements of area under the curve and root mean square. The performance of this model is based on 80% network training and 20% network testing. This division has been chosen as it delivers the best results.

Table 3.9: RBF Network training performance

	Sensitivity	Specificity	Accuracy	AUC	RMS
Breast Cancer	100	100	100	0.99054	0.1032
Bladder Cancer	100	100	100	0.98765	0.0000
Statlog (Heart)	100	100	100	0.99057	0.0000
Contraceptive	99.8305	96.5986	98.4481	0.96921	0.1419
German Credit	100	100	100	0.998	0.0000
Australian Credit	100	100	100	0.99531	0.0000
Ionosphere	100	100	100	0.98913	0.0000

Table 3.10: RBF Network testing performance

	Sensitivity	Specificity	Accuracy	AUC	RMS
Breast Cancer	91.2821	100	62.2549	0.49349	0.6144
Bladder Cancer	91.6667	14.7059	46.5517	0.54963	0.6251
Statlog (Heart)	100	100	54.321	0.51351	0.6759
Contraceptive	93.7931	88.2353	62.1739	0.4197	1.5247
German Credit	100	100	66.3333	0.48522	0.5800
Australian Credit	100	100	50.5435	0.024578	0.7065
Ionosphere	97.1831	95.8824	67.619	0.092792	0.6603

Figures 3.38 to 3.44 are basically scatter plots representing the results that are shown in Tables 3.9 and 3.10 above. Each of the 7 data sets have both trained and tested figures. These figures contain three plots: the actual predicted output, the rounded predicted output and Specificity vs. Sensitivity plots. According to these plots, the overall performance of the model will be compared to that of the other models to determine which one of them presents a better outcome. Relative to the results, the best prediction would yield a point in the upper

left corner or coordinate (0,1) of the ROC space, representing both 100% sensitivity (no false negatives) and 100% specificity (no false positives). In addition, the (0,1) point is considered a perfect classification. In the ROC plot, when the curve is nearly touching the left top corner coming closer to 1, that means that this ROC enjoys high accuracy and when it goes down that corresponds to low accuracy ROC. Therefore, the more higher and closer to the corner the curve is, the better results are obtained in terms of accuracy.

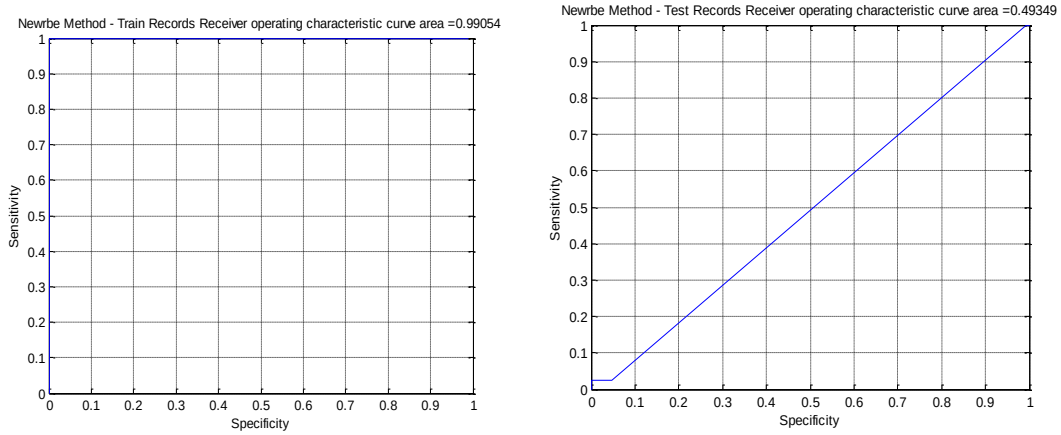


Figure 3.38: ROC training/testing of Breast cancer record

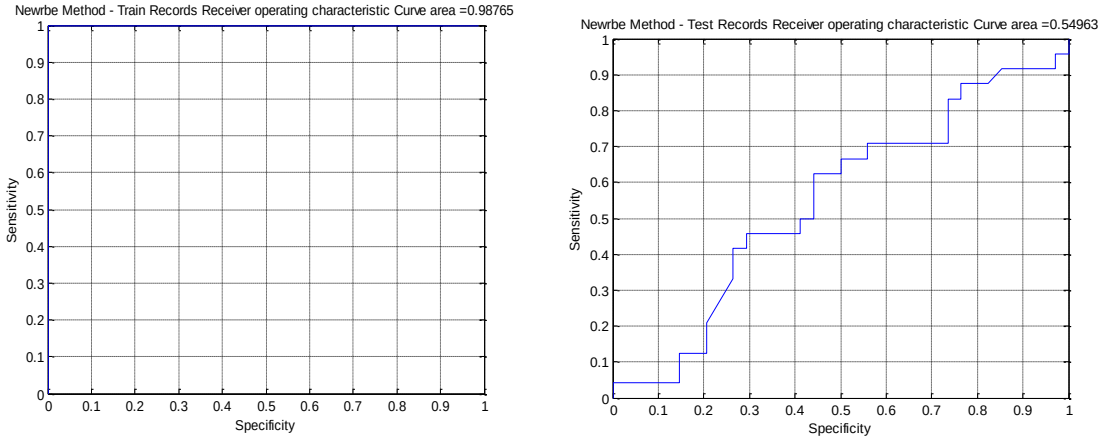


Figure 3.39: ROC training/testing of Bladder cancer record

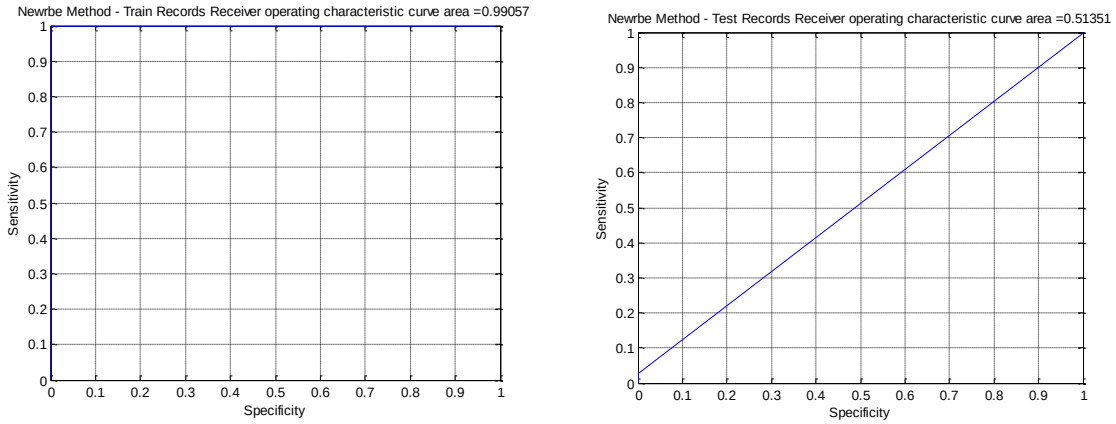


Figure 3.40: ROC of Statlog (Heart) record

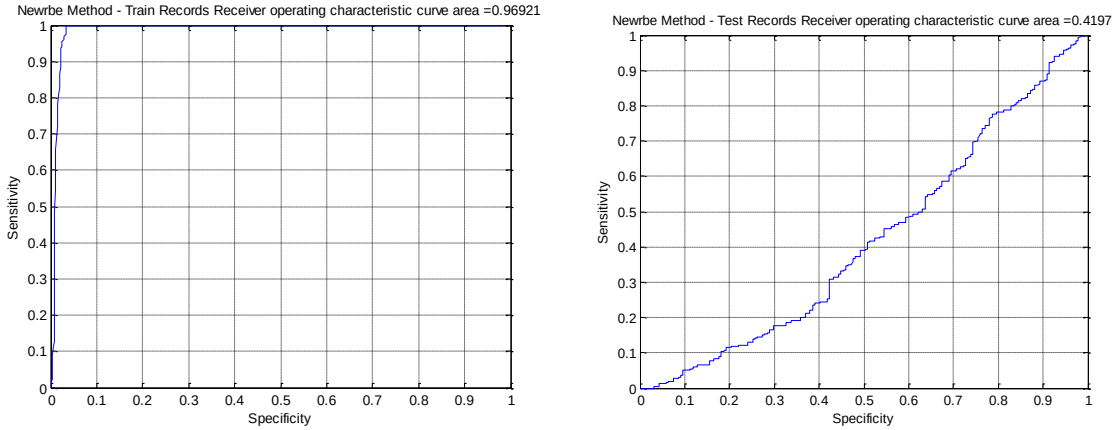


Figure 3.41: ROC training/testing of Contraceptive record

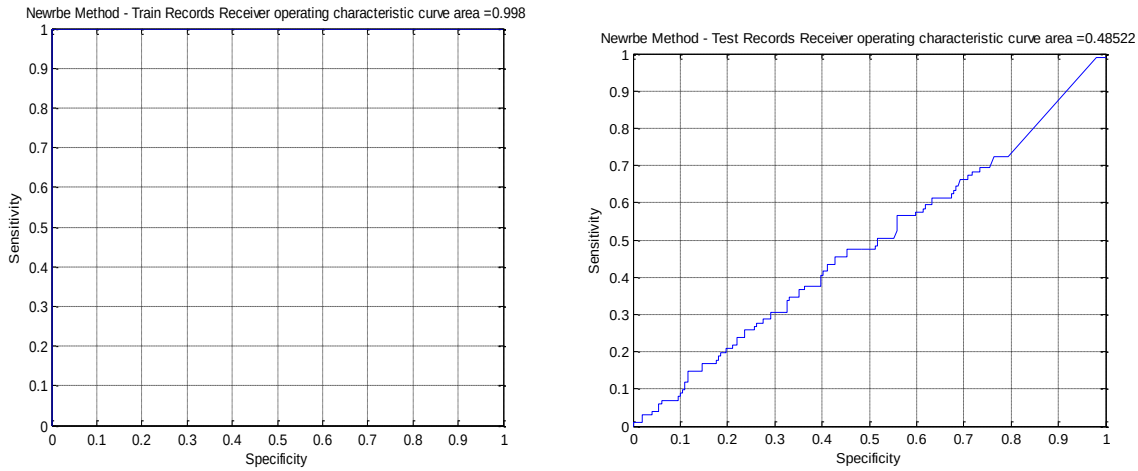


Figure 3.42: ROC training/testing of German record

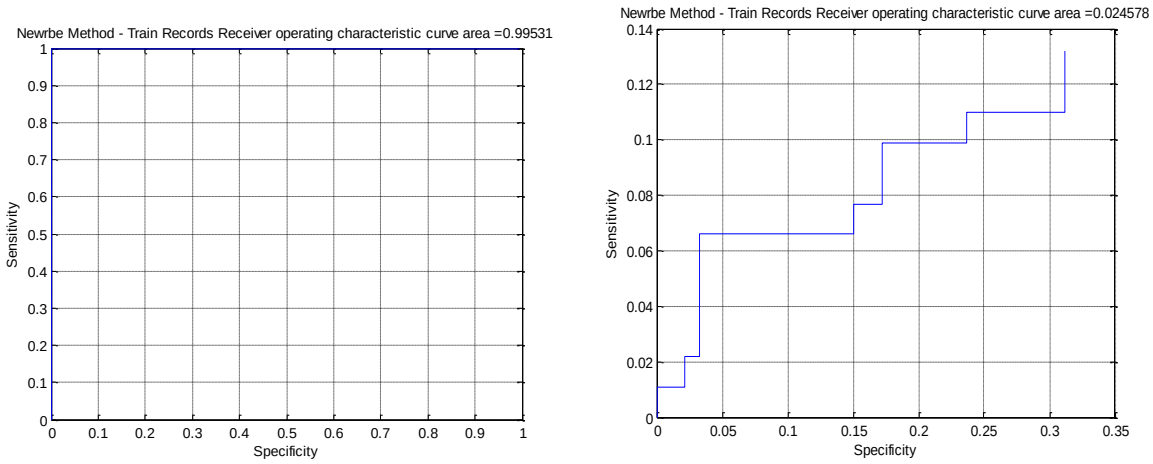


Figure 3.43: ROC training/testing of Australian record

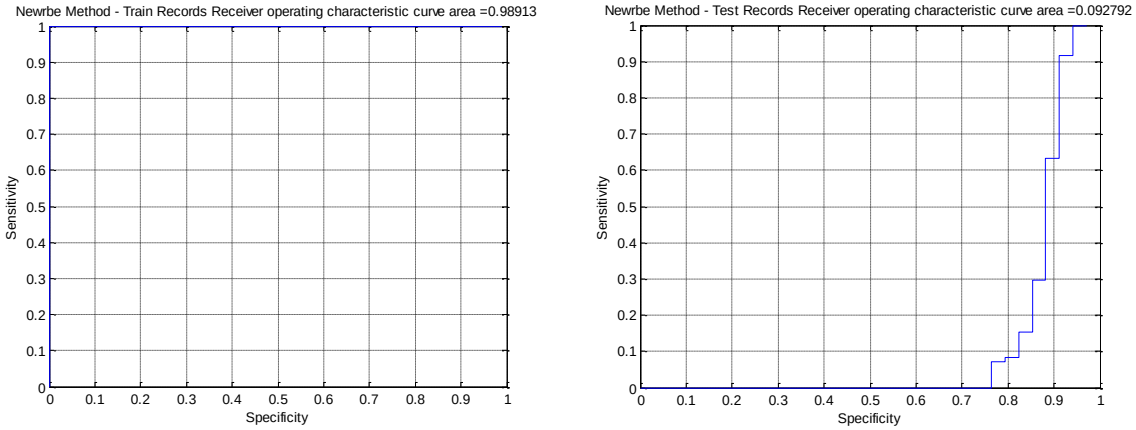


Figure 3.44: ROC training/testing of ionosphere record

3.5.6 Layered-Recurrent Network

Recurrent neural networks (RNNs) can be defined as a type of artificial neural network in which connections between units shape a direct cycle. This results in creating an internal state of the network allowing it to reveal dynamic temporal behaviour. RNNs have the ability to employ their internal memory for the sake of processing arbitrary sequences of inputs, making them practical in tasks such as recognition for connected handwriting achieving the best prediction results (Bullinaria, 2014).

In recurrent networks, the weight matrix for its layers contains input weights coming from all the neurons in the network not just from the prior layer. Unlike feedforward networks, they have feedback elements allowing signals from one layer to be fed back from the previous

one. A typical recurrent network is shown in Figure 3.45. It contains three layers: the input, output and hidden layer. Additional units are provided to the input layer that receives input from the neurons in the hidden layer. In addition, the feedback paths from the hidden layer to the units have fixed unit weight, shown in Figure 3.45 (Bullinaria, 2014).

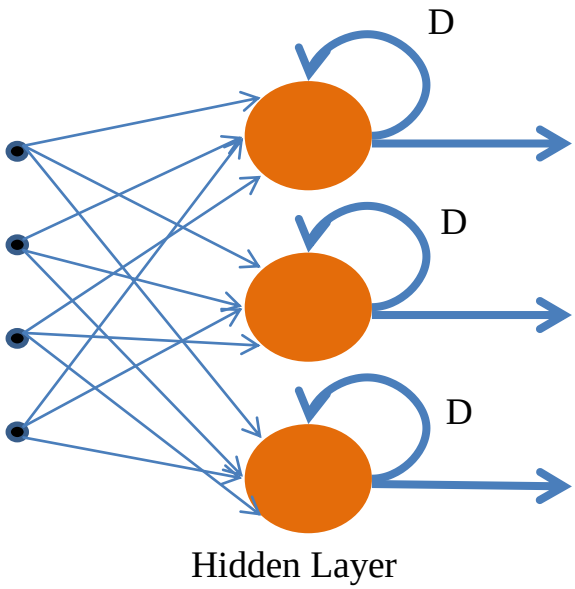


Figure 3.45: Layered recurrent network (Bullinaria, 2014)

The main thing about RNNs is that their networks contain at least one feedback connection, therefore, the activations can loop in the network. This allows the networks to perform temporal processing and sequence learning. RNNs can have many structures. One of the main structures comprises of a multi-layer perceptron or MLP in addition to the added loops. These can benefit the powerful non-linear mapping abilities of the MLP in addition to providing some kind of memory. Others possess a more uniform structure with each neuron connected to all other neurons in the network, or may have stochastic activation functions. Moreover, for conventional structures and deterministic activation functions, learning can be done by similar gradient descent processes to those that lead to the backpropagation algorithm to the feedforward neural networks. When the activations are stochastic, methods such as simulated annealing can be employed (Bullinaria, 2014).

As discussed above, the simplest structure of an RNN is an MLP with its prior hidden unit activations being fed back into the network with the inputs. This is shown in Figure 3.46 (Bullinaria, 2014).

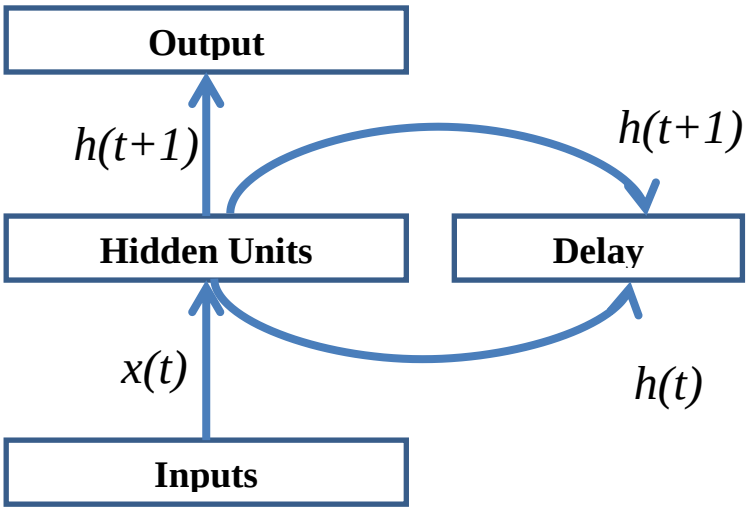


Figure 3.46: Simple structure of a recurrent network (Bullinaria, 2014)

According to the figure, time t must be discredited with the activations being updated at each time-step. The time scale may refer to the operation of the real neurons. Or, in artificial systems, any time step-size can be used for the problem provided. A delay unit requires introduction so as to hold activations until they are processed in the next time-step (Bullinaria, 2014).

By truncating the unfolded network to just a single time-step, this will result in reducing it to a Simple Recurrent Network known as an Elman network as shown in Figure 3.47.

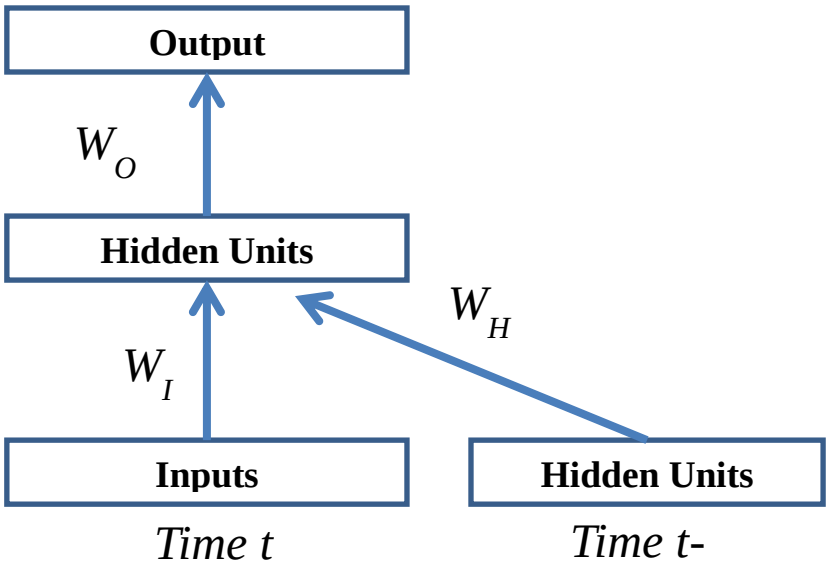


Figure 3.47: Elman network (Bullinaria, 2014)

According to the figure, each set of weights only appears once, therefore it is possible to apply the gradient descent technique using the conventional back-propagation algorithm. As a result, the error signal will not get back propagated very far, and thus it will be difficult for the network to learn how to use information from far time. Practically, this approximation is known to be very useful for many applications. An alternative structure contains a single input and a single output. There is also a delay line at the inputs and the outputs are fed back by a delay line to the input. This model is known as Non-Linear Auto-Regression with eXogenous inputs, or (NARX), shown in the Figure 3.48. It is effective for time-series prediction, where the target y(t+1) is x(t+1) as shown in Figure 3.48 (Bullinaria, 2014).

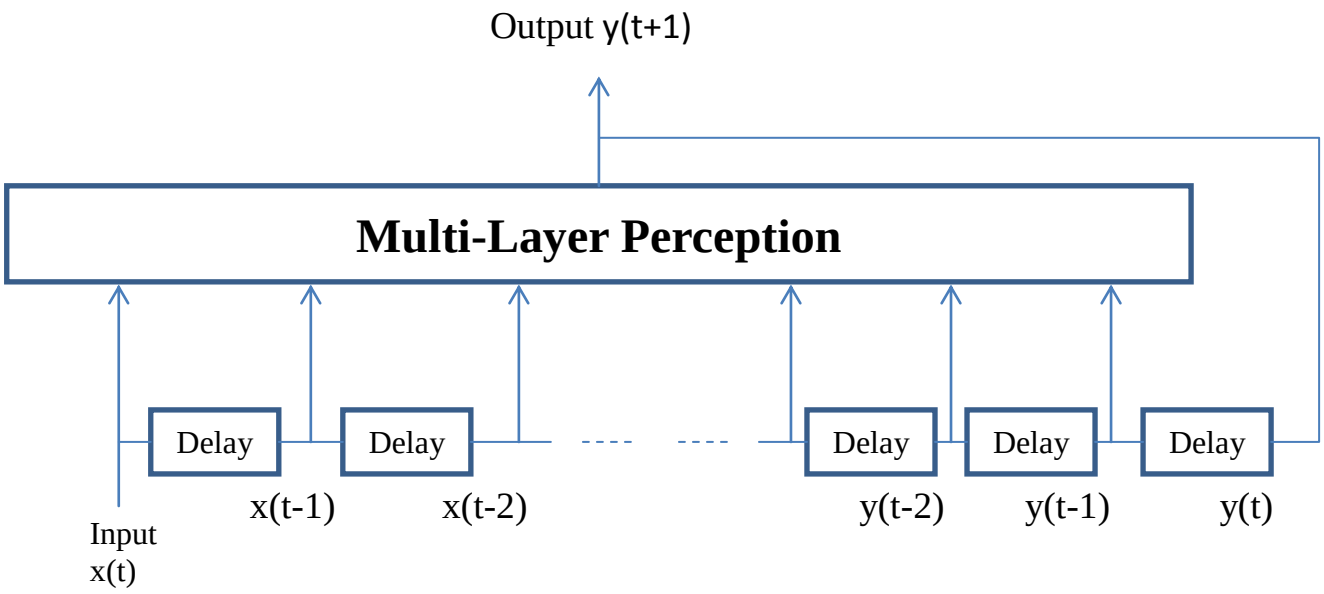


Figure 3.48: NARX Model (Bullinaria, 2014)

Unlike the feedforward networks, in the layered recurrent network each layer has a recurrent connection with a tap delay that it is associated with. This allows the network to have an infinite dynamic response to time series input data. It is similar to the time delay and distributed delay neural networks that are known to have infinite input responses.

Tables 3.11 and 3.12 show the performance results of layered-recurrent (LR) networks respectively. The tables represents a summary of the data sets. Some criteria were taken in consideration when choosing the datasets. Binary, non-binary and multi-class datasets are included in addition to a variety in the number of attributes and data items.

In each table, the row represents the 7 data sets. In addition, the column represents the analysis parameters. Each cell contains a number which refers to a percentage (for the first three columns) while the other two columns represent measurements of area under the curve and root mean square. The performance of this model is based on 80% network training and 20% network testing. This division has been chosen as it delivers the best results.

Table 3.11: LR network training performance

	Sensitivity	Specificity	Accuracy	AUC	RMS
Breast Cancer	99.3789	98.7382	98.954	0.9975	0.0000
Bladder Cancer	88.8889	87.6543	88.1481	0.93599	0.2846
Statlog (Heart)	96.3855	100	98.4127	0.98352	0.1200
Contraceptive	97.7966	39.229	72.7449	0.75893	0.5572
German Credit	68.8442	91.2176	84.8571	0.89757	0.3428
Australian Credit	95.3704	94.3662	94.8718	0.97068	0.2222
Ionosphere	100	92.0455	97.0954	0.98757	0.1711

Table 3.12: LR network testing performance

	Sensitivity	Specificity	Accuracy	AUC	RMS
Breast Cancer	94.8718	96.0317	95.5882	0.97884	0.2025
Bladder Cancer	47.8261	81.25	67.2727	0.69792	0.6201
Statlog (Heart)	52.9412	75.6098	65.3333	0.69963	0.6700
Contraceptive	94.4882	38.7097	70.9091	0.69409	0.5769
German Credit	53.4653	85.9296	75	0.77621	0.4324
Australian Credit	75.8621	82.7586	79.3103	0.84308	1.8391
Ionosphere	97.1831	72.7273	89.4231	0.94366	0.2684

Figures from 3.49 to 3.55 are scatter plots representing the results that are shown in Tables 3.11 and 3.12. Each of the 7 data sets mentioned above have both trained and tested figures. These figures contain three plots, the actual predicted output, the rounded predicted output and Specificity vs. Sensitivity plots. According to these plots, the overall performance of the model will be compared to that of the other models to determine which one of them presents a better outcome. Relative to the results, the best prediction would yield a point in the upper left corner or coordinate (0,1) of the ROC space, representing both 100% sensitivity (no false negatives) and 100% specificity (no false positives). In addition, the (0,1) point is considered a perfect classification. In the ROC plot, when the curve is nearly touching the left top corner

coming closer to 1, that means that this ROC enjoys high accuracy and when it goes down that corresponds to low accuracy ROC. Therefore, the more higher and closer to the corner the curve is, the better results are obtained in terms of accuracy.

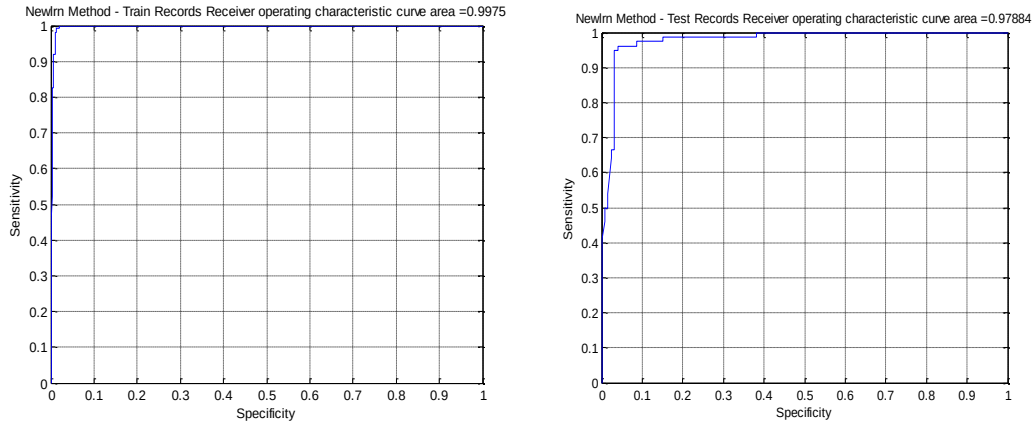


Figure 3.49: ROC training/testing of Breast cancer record

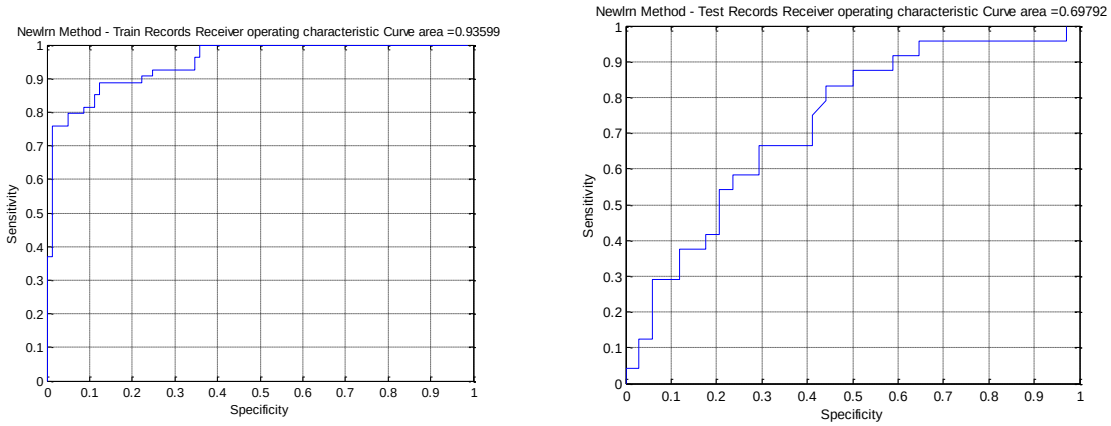


Figure 3.50: ROC training/testing of Bladder cancer record

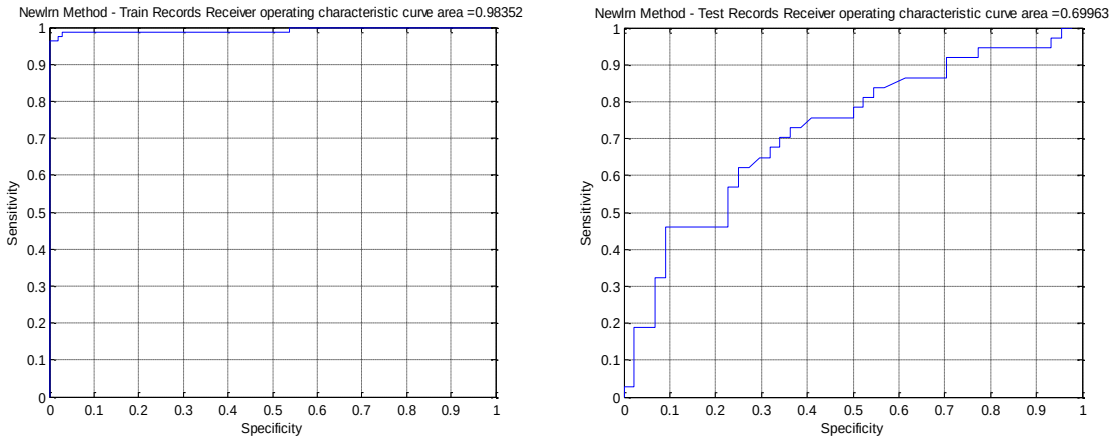
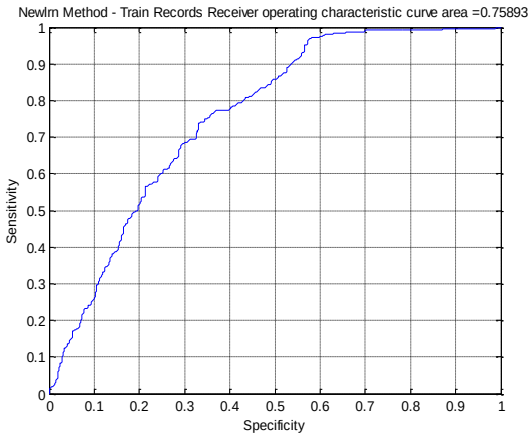


Figure 3.51: ROC training/testing of Statlog (Heart) record



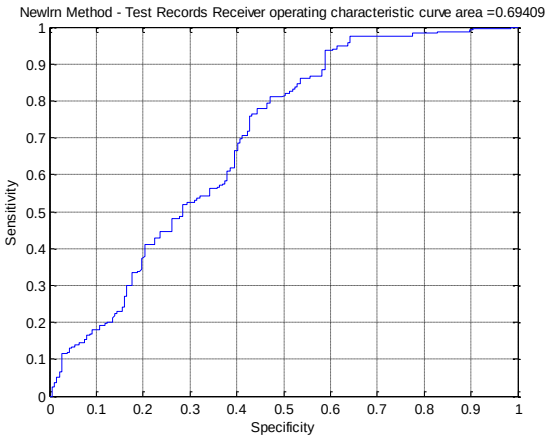
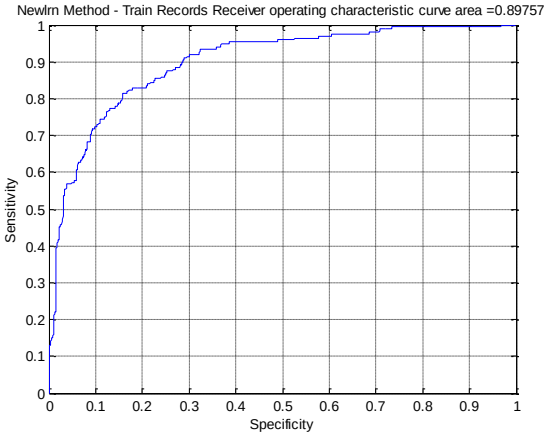


Figure 3:52: ROC training/testing of Contraceptive record



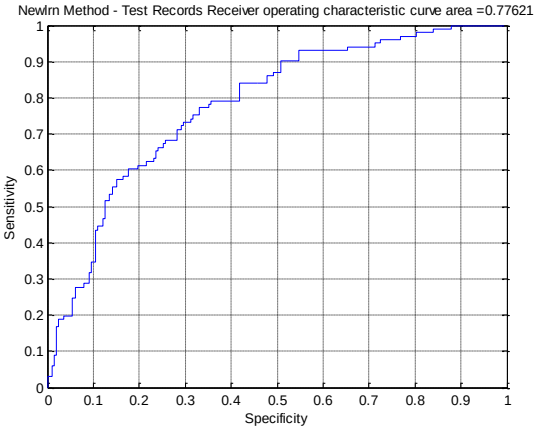
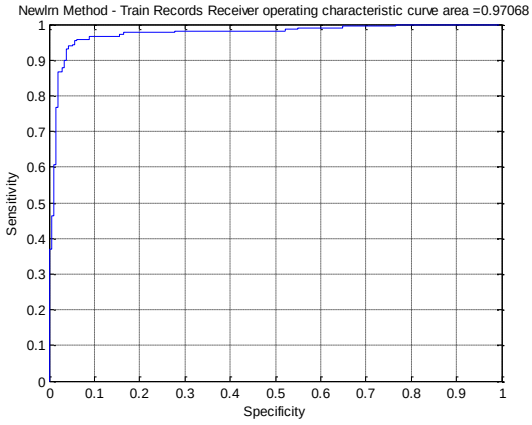


Figure 3.53: training/testing of German record



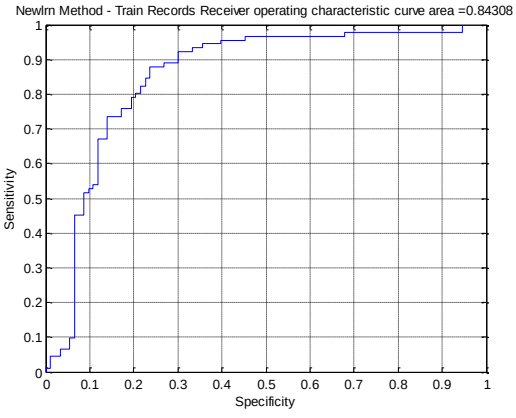


Figure 3.54: ROC training/testing of Australian record

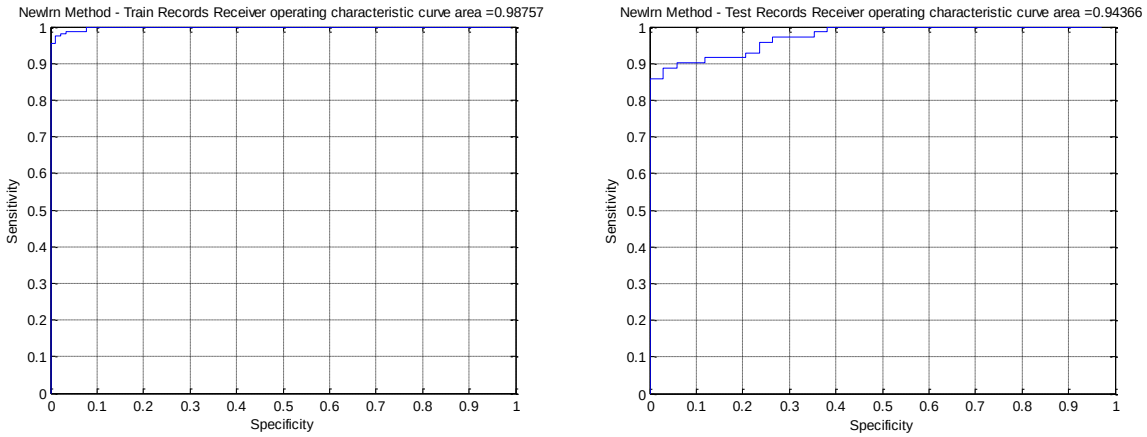


Figure 3.55: ROC training/testing of Ionosphere record

So far, each of the 6 network models was presented with training performance and testing performance. This has been done in order to measure the overall performance of the model by applying 7 data sets which are: Breast cancer, Bladder cancer, Statlog (Heart), Contraceptive, German Credit, Australian Credit and Ionosphere. Each data set was analysed using 5 different analysis parameters such as: Sensitivity, Specificity, Accuracy, AUC and RMS. To take a look at the testing performance values of these parameters and compare them with the training performance ones, they were embedded in two tables to further assist with the next step. The next step is to plot the tables in order to present graphs showing the performance of each data set training-wise and testing-wise.

Based on the results obtained from the tables and figures in this thesis, it can be clearly seen that the six models Cascade-forward back propagation, Feedforward Input time-delay backdrop network ,Fitting network, Feedforward back propagation network, Radial basis network and Layered- recurrent network have been shown greater output for data sets in general.

The results represented in tables 3.13 and 3.14 shows the comparison the results of the 6 different models of artificial neural network based on AUC output results for all datasets using all training dataset sizes. The table clearly shows that each data set leads to different performance .

Table 3.13: AUC Comparison of six models of ANNs training performance

Training	CFBP	FFITBP	FITT	FFB	RBF	LR
Breast Cancer	0.99197	0.98364	0.99422	0.98864	0.99054	0.9975
Bladder Cancer	0.9337	0.67353	0.78624	0.9209	0.98765	0.93599
Statlog (Heart)	0.94158	0.95044	0.3480	0.92464	0.99057	0.98352
Contraceptive	0.69419	0.76149	0.70272	0.74158	0.96921	0.75893
German Credit	0.91382	0.82838	0.75409	0.88295	0.998	0.89757
Australian Credit	0.96488	0.2668	0.93762	0.94829	0.99531	0.97068
Ionosphere	0.92654	0.78588	0.96839	0.89336	0.98913	0.98757

Table 3.14: AUC Comparison of six models of ANNs testing performance

Testing	CFBP	FFITBP	FITT	FFB	RBF	LR
Breast Cancer	0.97894	0.96083	0.97029	0.98952	0.49349	0.97884
Bladder Cancer	0.59926	0.72672	0.56863	0.69118	0.54963	0.69792
Statlog (Heart)	0.82064	0.91032	0.81388	0.78808	0.51351	0.69963
Contraceptive	0.65916	0.71596	0.66878	0.67097	0.4197	0.69409
German Credit	0.75123	0.77236	0.6976	0.75541	0.48522	0.77621
Australian Credit	0.9063	0.86683	0.90571	0.92887	0.024578	0.84308
Ionosphere	0.86495	0.71582	0.84051	0.8169	0.092792	0.94366

3.6 Summary

Neural Networks are generally an artificial simulation to the human brain in processing and analysing information. The more these networks are fed with information, the more they are efficient in methods such as pattern recognition and data classification. They possess many advantages such as learning and recognising patterns from previous data. In addition, they provide real time operation and error recovery systems to better process and handle information. Each neural network model conveys a method by which the input data is classified and analysed. Some of these models depend on previous models with few to major additions to the model for better accuracy and performance.

From the models discussed in this chapter, there is not even a proper definition of what ANNs really are or the possibilities of the applications in different fields. Though, in order for a model to be known as "neural", they must possess certain points such as (Stergiou and Siganos, 2015):

1. They must have a set of adaptive weights which are numerical factors adjusted by a learning algorithm

2. They must be able to approximate non-linear functions of their inputs.

The experimental data mentioned in this chapter was for the purpose of clarifying the real-world applications of machine learning and intelligent systems in harnessing the huge amount of data available for each data set and how it is handled. Neural networks are not meant to make miracles. But if used accurately and purposefully they can provide some amazing advanced results (Stergiou and Siganos, 2015).

Chapter 4: Ensembles

4.1 Introduction

Ensemble learning can be defined as the method in which several models known as classifiers or experts are purposefully combined and produced for analysing and solving problems related to computational intelligence. Ensemble learning is mainly used to improve methods such as prediction, classification, and function approximation in addition to reducing the likelihood of a poor selection. Other applications include selecting the confidence level to the decision made by the model, choosing optimal features, incremental learning, non-stationary ensemble learning, data fusion and error correction (Navone, 2001).

On the other hand, hybridization is basically a system that depends on combining different two or more techniques from various artificial intelligence systems (e.g. neuro-fuzzy systems, fuzzy expert systems, evolutionary neural networks, etc..). Furthermore, the hybridization technique combines two or more algorithms in order to improve the performance for problems concerning optimization. In order to make that happen, the hybridization embeds the most appropriate features of the algorithms combined to construct a new high-level algorithm (Corchado et al., 2009).

An ensemble system is provided by combining different models known as "classifiers". As a result, these systems are recognised as classifier systems or ensemble systems. For the sake of fully understanding the necessity of employing ensemble systems it is important to analyse the psychological approach. Simple examples include questioning different doctors in the same field before going through a certain surgery or treatment, or picking the best laptop by doing some research on product reviews or simply asking different experts in computer shops. Even this research was based on different resources on material concerning ensemble learning. As a result, this will assist in reducing poor and misguided decisions concerning performing a medical procedure, choosing the best electrical device, or producing a comprehensive research (Navone, 2001).

The method of classifying various inputs into many classes is recognised by researchers to be the most popular method of pattern recognition tasks. Classification methods require

constructing statistical models that specify a mapping of the input data, which are different features, into the appropriate outputs. Hence, the goal for these models is to approximate the correct mapping of the inputs to the outputs providing the intention of constructing predictions based on the outputs for either the new or the previously analysed inputs. As a result, this method can have a variety of applications in many fields such as medicine, banking and accounting, sensor technology, weather forecasting and many other domains (Polikar, 2006). Furthermore, this chapter will discuss in detail the types of ensemble learning and the advantages of ensemble learning. Furthermore, some applications concerning the use of ensemble learning in the field of medicine will be discussed briefly.

4.2 Ensemble Methods

Types of ensemble learning, generally, there are two types of ensemble learning: supervised and unsupervised learning. Each one will be discussed in this section. In supervised learning, a set of training examples, with their outputs known, is employed by a learning algorithm to produce a classifier. A practical example is the classification method which is used to predict a legitimate or a fraudulent credit card operation. In this process, each pattern input represents a particular charge and consists of various features such as amount of transaction, time of transfer, credit card balance and location in addition to the output pattern which is the legitimacy of the operation (Polikar, 2009).

A learning algorithm employs the provided examples to further generate a classifier that estimates the mapping of each transaction and its legitimacy. As a result, this classifier can be used to verify the legitimacy of a new transaction (Polikar, 2009). However, unsupervised learning comprises locating a class to a training example without the need to have a target class. Going back to the credit card system, various credit card operations are clustered together based on the basis of similarity. This data can be used to analyse transactions that require reviewing by another algorithm or a bank teller. This technique is recognised as “clustering”;this technique works by looking for similarities in large data groups in which class memberships are unknown for the training system and the observations of similar nature are clustered into one subset. (the data groups which are similar are clustered together (joined) in a technique called “clustering”).

Clustering is the assignment of a set of observations into subsets (called clusters) so that observations in the same cluster are similar in some sense. Clustering is a method of unsupervised learning, and a common technique for statistical data analysis used in many fields.

Several learning algorithms produce a single classifier (it can be a decision-tree or a neural network) which can be utilized in making predictions for the new examples. On the other hand, several decisions such as settings for initial model parameter can have an impact on the performance of the particular classifier. One option is to assign the most suitable classifier. However, this technique does not offer the optimum solution in several cases. In addition, due to the actuality that several classifiers are tested prior to assigning a single classifier, this technique also disregards significant information by ignoring the performance of all the other classifiers(Polikar, 2009). However, this method does not give the optimum solution in many cases. Moreover, due to the fact that several classifiers are tested prior to picking a single classifier, this method also ignores valuable information by ignoring all the other classifiers' performance (Polikar, 2009).

Classifier ensembles are collections of many classifiers in which each of their predictions are combined in particular methods in order to generate the final decision. Ensembles give not only better but more robust solutions in most of their applications due to the fact that they utilize all of the available classifier information. For example, when considering the case where neural networks having different structures of even different weights produced from a supplied training set. Therefore, an ensemble of these classifiers can be constructed by getting each classifier presenting a prediction for a certain pattern and providing feedback to the class that provides the maximum voters. Several learners and scientists have proved that ensembles' performance is higher than that of their base models if the base models provide good performance on noticeable examples and outputs errors on different examples. (Polikar, 2009).

Supervised learning systems are specialised in seeking an appropriate hypothesis from the hypothesis space in order to make an appropriate prediction to a given problem. However, it is difficult to find a good hypothesis despite the fact that there exist appropriate hypotheses in the hypothesis space. In other words, an ensemble is a method of combining weak learners for the sake of producing strong learners (Polikar, 2006).

Evaluating an ensemble prediction basically requires more complexity in operation than single-mode prediction, so it can be considered that ensembles are a technique aiming to compensate for poor learning algorithms by performing many extra computations. Fast algorithms are commonly used with ensembles, although in some applications slower algorithms can be beneficial as well (Polikar, 2006).

An ensemble is intrinsically a supervised learning algorithm because it is capable of being trained and is used to make predictions. The trained ensemble thus represents a single hypothesis, however, it is not compulsory for it to be included within the hypothesis space of the ensemble method. As a result, ensembles enjoy a significant amount of flexibility enabling functions to overfit training data more than single models in theory. However, practically some ensemble methods such as bagging can reduce the problems concerning overfitting the training data (Polikar, 2006).

4.2.1 Why ensemble learning works

There are three main reasons why ensembles can perform better than single classifiers. The reasons in general are statistical, computational and representational. According to Figure 4.1, a learning algorithm can be analysed as seeking a space H of hypotheses in order to identify the best hypothesis.

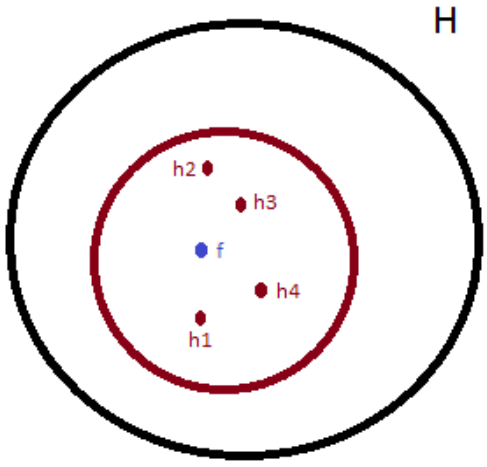


Figure 4.1 Statistical analysis

Providing a finite amount of data, many hypotheses are basically equally good. By performing the average of the accurate classifiers, this can provide a significant approximation of *f* as shown in Figure 4.2.

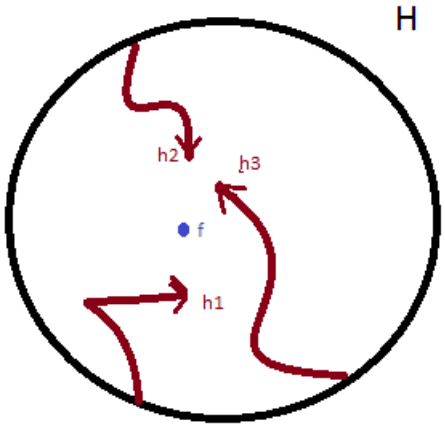


Figure 4.2 Computational Analysis

Since practical hypothesis spaces tend to be huge and infinite, a thorough search is therefore required, thus the learner might have a difficult time in local minimum. To fix the issue of local minima, repeating the research number of times each with random restarts can therefore be successful in constructing an ensemble as shown in Figure 4.3.

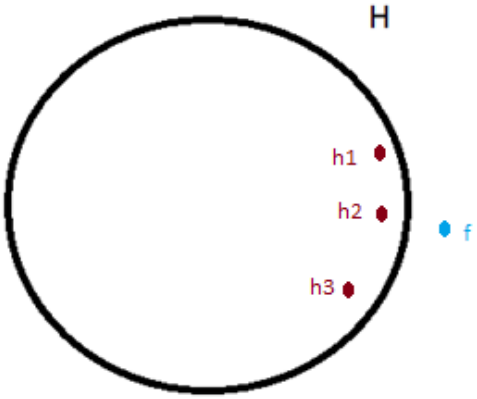


Figure 4.3 Representational Analysis

In addition, the target function might not be attained through individual classifiers, but it can be obtained through ensemble averaging (Dietterich, 2009).

4.2.2 Selection of models

Select of the model can be considered as one reason why ensemble systems are suitable model and practical. There are two parameters for picking the most appropriate classifiers. The first parameter is the type of classifier to be chosen among the many practical classifiers such as MLP, Naïve Bayes classifiers, support vector machines etc. The second parameter is

which form of the classifier to be chosen for the sake of providing different decision boundaries even if all the parameters are constant. However, the best-known technique, which is picking out the classifiers having the smallest error on the training data, is not a very practical method. Performance measured on a training data set even when being measure via a cross-validation technique is proven to be inaccurate according to the classification performance with previously analysed data. After that, all the classifiers may have the same training data or the same classifier performance "pseudo generalisation performance", so the learner is encouraged to choose at random. However, this can lead to choosing a poor model. Using an ensemble of these models can decrease the risk of a poor performance classifier. There is no guarantee that combining multiple classifiers will deliver better results than the individual classifier with the best performance in an ensemble. Alternatively, it can reduce the risk of a poor decision as mentioned earlier (Polikar, 2006).

To ensure the effectiveness of this process, the individual experts should present some level of diversity. Then, the level of diversity in the classifiers (which are obtained by using different training parameters for each classifier), permits individual classifiers to produce different decision boundaries. If suitable diversity is attained, a different error is generated by each classifier, a strategic combination that can result in reducing the total error (Polikar, 2006).

4.2.3 Data Size

Ensemble systems are practical with both large volumes of data and not enough data. In the case of large data making it difficult to train single classifiers, the data can be divided into subsets. Each subset can be assigned to train a separate classifier and then combined using different combination rules which will be discussed later. On the other hand, when there is little data, a technique known as bootstrapping is used where bootstrap samples of the data can be used to train different classifiers and each sample is basically a random sample of the data generated with replacement and analysed as if it was independently generated from the underlying distribution (Polikar, 2006).

4.2.4 Data Fusion

In many applications concerning automated decision making, the information received from different sources do not necessarily give out complementary information. Data fusion is known to be suitable combination of the information. This technique can lead to enhanced accuracy of the classification decision in comparison to a decision based on any other data

source. As an example, when diagnosing a neurological disorder, a neurologist may use the following methods:

- One dimensional time series data such as electroencephalogram
- Two-dimensional spatial data such as positron emission tomography (PET) scan and magnetic resonance imaging (MRI).
- Scalar and/or categorical data such as the amount of certain chemicals in the spinal fluid, age, gender, etc.

These features cannot be all employed together in training a single classifier and even if it did, it would not be successful. In these cases, an ensemble of classifiers can be beneficial in which each classifier is trained independently on the feature sets. The decisions produced by each classifier are combined using the combination rules described later (Polikar, 2009).

4.2.5 Confidence Estimation

Ensemble systems enjoy a nature of assigning a confidence to the decisions made by these systems. For example, taking an ensemble of classifiers trained on a specific problem, if the majority of classifiers agree with their decisions, the system is said to have high confidence in its decision. However, if half of the classifiers make one decision and the other half makes another, this can result in an ensemble of low confidence in the decision. It is important to know that a high confidence ensemble system does not mean that the decision is correct, just as low confidence does not mean that the decision is incorrect (Polikar, 2006).

4.2.6 Diversity

The main reason behind the ensemble's capability to correct the errors of its members is the diversity of the classifiers that constitute the ensemble. Moreover, if all classifiers would provide the same output, the correction of a certain mistake would be impossible. Therefore, individual classifiers need to make different errors at different times within an ensemble system. The goal here is that each classifier produces a different error, and then a proper combination of these classifiers can then reduce the total error, somewhat similar to low pass noise filters.

In general, an ensemble of classifiers is said to be diverse if it has classifiers with decision boundaries which are different from one another. Classifier diversity can be attained through several methods. The most commonly used method is "using different training data sets to train individual classifiers". These data sets can be attained through resembling techniques,

e.g. bagging and bootstrapping. In these methods, training data sets are drawn randomly from the entire training data. To certify that the individual boundaries are different despite the use of similar training data, weaker or unstable classifiers are used as base models due to the fact that they can generate different decision boundaries sufficiently even for small perturbations in their training standards.

Another method to obtain diversity is to "use different training parameters for different classifiers". To further understand how this method works, an example of a series of multi-layer perception neural networks can be trained via different weight initialisations, number of nodes, error goals among other parameters. By adjusting these parameters, this can allow monitoring the instability of the individual classifiers and therefore assist in the diversity. On the other hand, to add more diversity, different classifiers such as MLP, nearest neighbourhood classifiers, decision trees and support vectors can be combined. Or, diversity can be achieved through using different features or subsets of existing features. Generally, producing different classifiers using random feature subsets is known to be the "random subspace method" (Polikar 2009).

4.2.7 Applications

In the field of ensemble learning, pattern recognition is widely used and has generally divided the applications into three subfields. These subfields are remote sensing, identification recognition (fingerprint, iris recognition, etc.), and one versus all classification (fault and intrusion detection and medicine) (Oza, 2008).

Each of these fields projects some difficulties in statistics and amount of data (little or too much) that makes it hard and sometimes impossible for conventional pattern recognition algorithms to be practical in these applications. This section will mention in detail their applications in medicine (Oza, 2008).

The general purpose of medical applications is to maintain the health of human beings and improve the quality of life. Ensemble learning problems in medicine possess several common characteristics. These characteristics are:

- Limited training and testing samples: because of the nature of the problem and privacy issues.

- Imbalanced data sets: few people have a particular disease or an abnormality.

- Too many attributes: sometimes more than the test examples.

A few examples of ensemble methods are described below in different aspects of medicine (Oza, 2008).

4.2.8 Analysis of ECGs

Computer based electrocardiography (ECG) analysers are widely used in clinical practice. The current systems depend on mathematical and statistical algorithms to perform ECG signal analysis. There are several approaches that employ neural networks to enhance the accuracy in diagnosis in order to achieve a more systematic operation despite the complicating factors. Long-term ECG recording evaluation requires automated recognition of events occurring infrequently. Specifically, about 90000 ECG recordings a day which is a tiring and time-consuming operation. However, a proper practical neural network can recognise abnormalities with up to 99.9% accuracy (Papik, 1998).

4.2.9 Oncology

In the subject of cancer treatment, having a predictive model for cancer prognosis is very helpful, especially with the high level of current research on the subject. Cancer treatment is very serious and can be harmful to patients as it includes radiation and chemotherapy. For that reason, the treatment should be aggressive to an extent of necessity to guarantee the best outcome. If the prognosis of the patient is known beforehand, the treatment can be planned so that the patient will have the best survival rate and at the same time face less toxicity from the treatment as possible (Papik, 1998).

Gene expression micro array data has offered a method to produce genetic profiles of malignant tissues through quantifying the expression of thousands parallel gene sequences in a single biological sample. Gathering genetic sample from patients and then calculating the time of recurrence of cancer can provide information about the link between the gene expression and the rate of progress of the disease. This type of information can then be utilised to construct a predictive model that has the ability to provide a prognosis for patients thus assisting physicians in working on the treatment resulting in more effective treatment and reducing the toxic effects of the treatment (Ford, 2012).

There are several diagnostic systems and therapeutic strategies to tackle breast cancer. A neural network successfully judged the probable rate of recurrence of tumours with a success rate of 960 out of 1008 by using information from patients with lymphatic nodes such as

tumour size, status of tumour hormone receptor, number of palpable lymphatic nodules, etc (Ford, 2012).

4.2.10 Radiology

In the field of radiology, ensemble learning is very interesting and most powerful. Images contain so much data that they are too complex to be analysed by conventional systems. By selecting the right training set and learning process, neural network models can be very beneficial in noise filtering and recognising unusual imaging. Conventional imaging analysis processes information row by row. However, new systems use standards from the image by pro-processing. These standards are obtained from the feature of the images. For example, abdominal ultrasound and laboratory analysis do not often give out enough information for diagnosing liver diseases. However, using ultrasonographic and laboratory results, a trained neural network was generated to characterise five classes of liver diseases. Another prominent application employs a back propagation algorithm to detect 7 coronary artery disorders based on myocardial SPECT imaging. In addition, a neural network has successfully been applied to detect the microcalcification in digital mammograms. This algorithm was able to situate regions of concern and also to differentiate pathological changes from false-positive alterations (Papik, 1998). The other important application is multi-modality fusion for example PET/CT computed tomography, projection CT/X-ray and X-ray/ultrasound.

4.2.11 Other Fields of Medicine

- Treatment of speech and hearing impairment
- Differentiating the diagnosis between Alzheimer's diseases and vascular dementia
- Diagnosing shape abnormalities of the cornea
- Examining heart rate regulation and heart failure
- Analysis of heart enzyme levels

4.3 Experimental Results

In order to generate classifier ensembles, data need to be divided into two sub data; training and testing. The testing data remains the same while the training data is chosen via a sliding window. The training data is then partitioned into 10 subsets, each of these sets is trained 10 times, then the best result is selected. Thus, 10 models out of 100 are trained and the final

prediction will be the average taken for these models. As for the testing data, the best 10 models are taken and fed to the training data to observe the prediction. The ensemble is devised to model 7 datasets.

Tables 4.1 to 4.7 show the performance results of the ensemble method. The tables represents a summary of the data sets. Some criteria were taken in consideration when choosing the datasets. Binary and multi-class datasets are included in addition to a variety in the number of attributes and data items. In each table, the row represents the performance results of the data sets. In addition, the column represents the analysis parameters which are: Sensitivity, Specificity, Accuracy, AUC and RMS. Each cell contains a number which refers to a percentage (for the first three columns) while the other two columns represent measurements of the area under the curve and the root mean square. The performance of this method is based on 80% on network training and 20% on network testing.

Furthermore, for training and testing, different percentages for each proportion was made. For example, the ratio of 90% of training to 10% testing was made in addition to 70% training to 30% testing. However, the best performance was obtained from having 80% for training and 20% for testing, therefore we have utilized the 80 to 20 percentage as a default method throughout the research.

Figures 4.4 to 4.10 are scatter plots representing the results that are shown in Tables 4.1 to 4.7. Each of the 7 data sets mentioned above have both trained and tested figures respectively. These figures contain, Specificity versus Sensitivity plots. According to these plots, the overall performance of the model will be compared to the predicted results. Relative to the results, the best prediction would yield a point in the upper left corner or coordinate (0,1) of the ROC space, representing both 100% sensitivity (no false negatives) and 100% specificity (no false positives). In addition, the (0,1) point is considered a perfect classification. In the ROC plot, when the curve is nearly touching the left top corner coming closer to 1, that means that this ROC enjoys high accuracy and when it goes down that corresponds to low accuracy ROC. Therefore, the more higher and closer to the corner the curve is, the better results are obtained in terms of accuracy.

4.3.1 Johns Hopkins University Ionosphere

Table 4.1 shows the performance results of the Ionosphere data sets. Figure 4.4 shows the receiver operating characteristic (ROC) of the Ionosphere data sets.

Table 4.1: Ionosphere training/testing performance

	Sensitivity	Specificity	Accuracy	AUC	RMS
Training	100	90.7216	96.3115	0.96488	0.2132
Testing	100	89.2857	97.1429	0.95918	0.1834

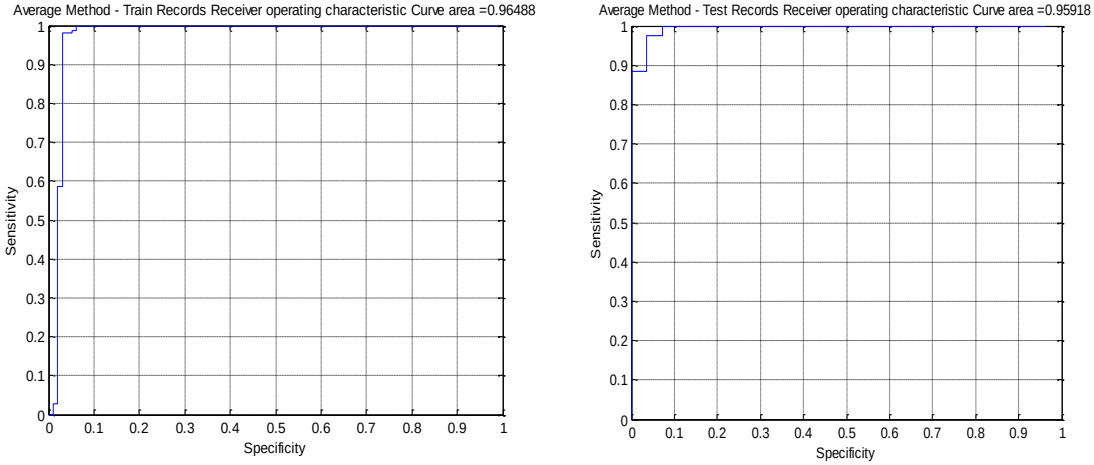


Figure 4.4: ROC training/testing of Ionosphere records

4.3.2 Contraceptive Method Choice

Table 4.2 shows the performance results of the contraceptive data sets. Figure 4.5 shows the ROC of the contraceptive data sets.

Table 4.2: Contraceptive training/testing performance

	Sensitivity	Specificity	Accuracy	AUC	RMS
Training	97.8078	57.7626	80.7953	0.83893	0.4579
Testing	97.2	58.6387	80.4989	0.83802	0.4596

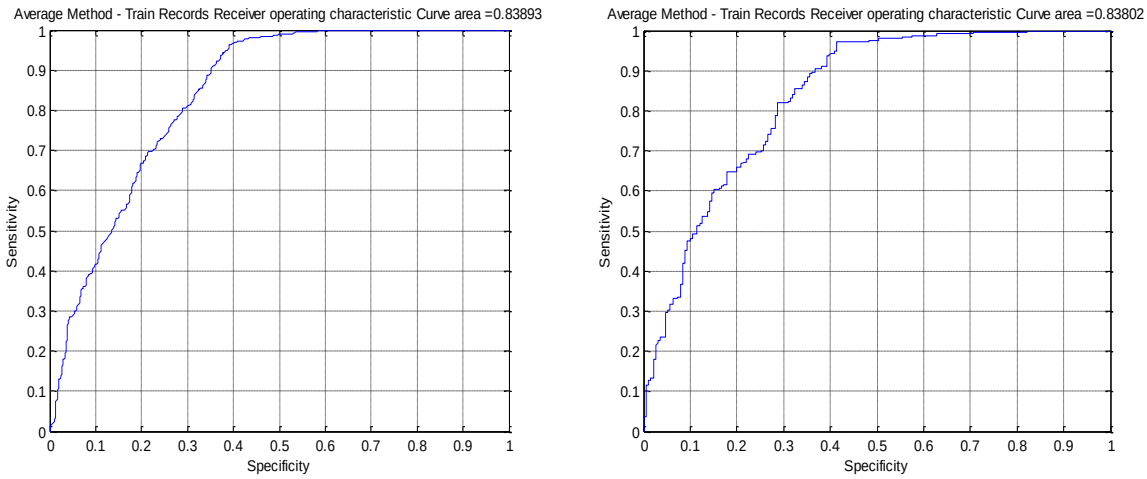


Figure 4.5: ROC training/testing of Contraceptive records

4.3.3 Statlog (Heart)

Table 4.3 shows the performance results of the Statlog data sets. Figure 4.6 shows the ROC of the Statlog data sets.

Table 4.3: Statlog training/testing performance

	Sensitivity	**Specificity**	**Accuracy**	**AUC**	**RMS**
Training	90.8046	95.098	93.1217	0.95876	0.2711
Testing	96.9697	97.9167	97.5309	0.97727	0.2219

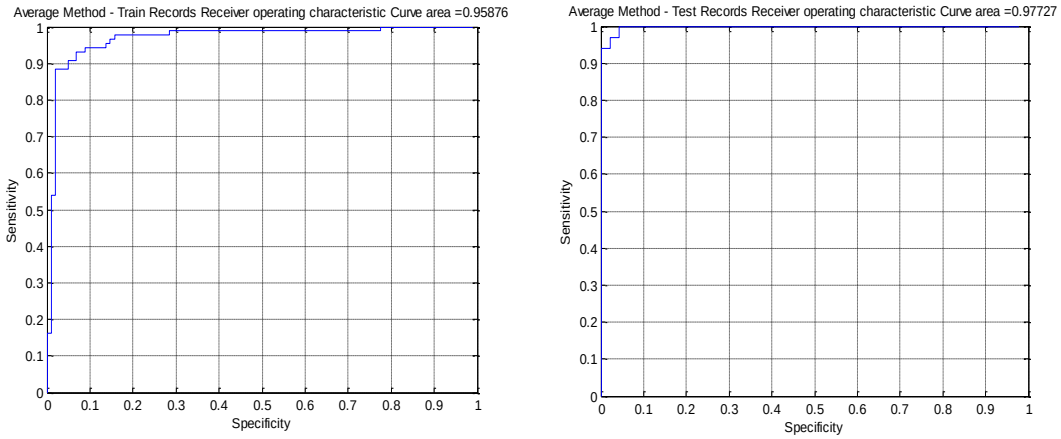


Figure 4.6: ROC training/testing of Statlog (Heart) records

4.3.4 German Credit

Table 4.4 shows the performance results of the German data sets. Figure 4.7 shows the ROC of the German data sets.

Table 4.4: German training/testing performance

	Sensitivity	Specificity	Accuracy	AUC	RMS
Training	50.6849	95.4262	81.4286	0.88451	0.3690
Testing	55.5556	96.347	85.3333	0.89926	0.3443

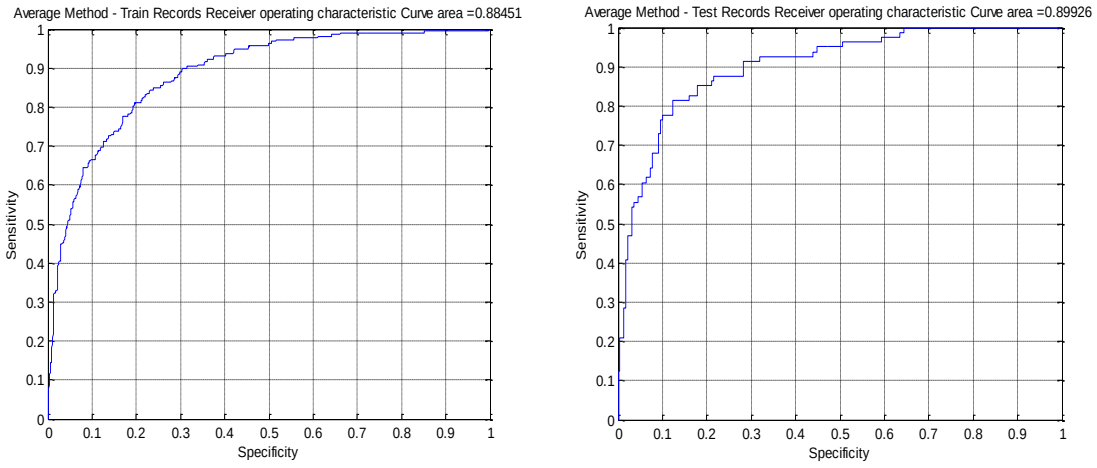


Figure 4.7: ROC training/testing of German records

4.3.5 Australian Credit Approval

Table 4.5 shows the performance results of the Australian data sets. Figure 4.8 shows the ROC of the Australian data sets.

Table 4.5: Australian training/testing performance

	Sensitivity	Specificity	Accuracy	AUC	RMS
Training	94.4186	82.7103	88.5781	0.95399	0.2856
Testing	91.3043	91.3043	91.3043	0.96467	0.2626

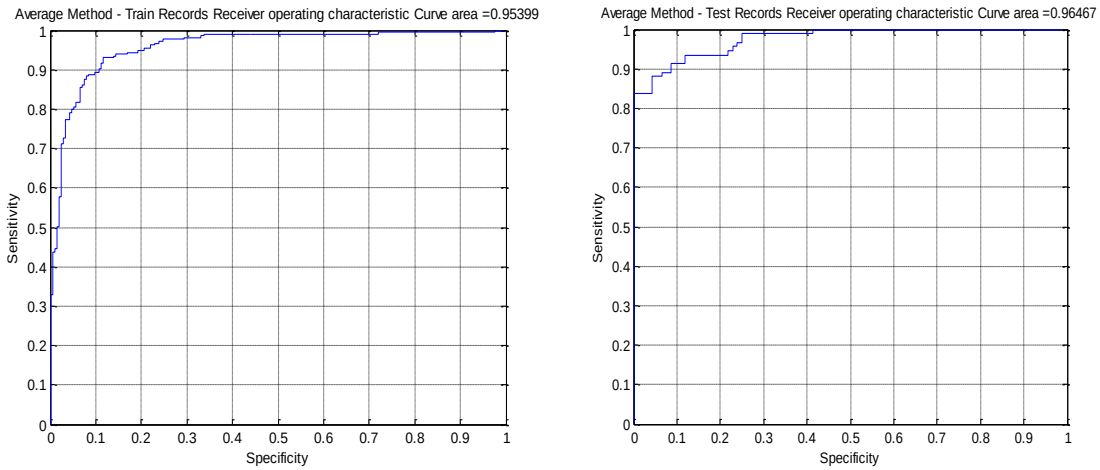


Figure 4.8: ROC training/testing of Australian records

4.3.6 Breast Cancer Wisconsin

Table 4.6 shows the performance results of the Wisconsin data sets. Figure 4.9 shows the ROC of the Wisconsin data sets.

Table 4.6: Breast cancer training/testing performance

	Sensitivity	**Specificity**	**Accuracy**	**AUC**	**RMS**
Training	93.7888	99.3691	97.4895	0.99065	0.1712
Testing	93.5897	98.4127	96.5686	0.98168	0.1869

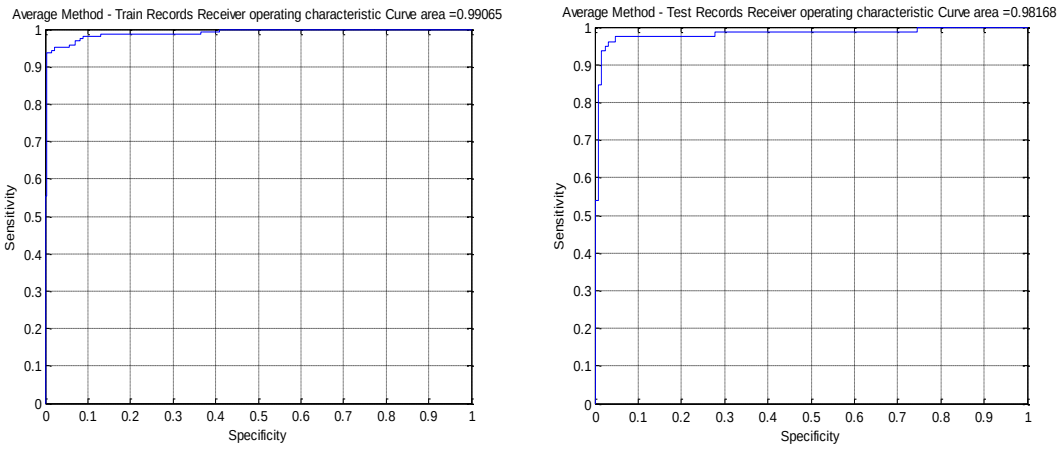


Figure 4.9: ROC training/testing of Breast cancer records

4.3.7 Bladder Cancer

Table 4.7 shows the performance results of the bladder data sets. Figure 4.10 shows the ROC of the bladder data sets.

Table 4.7: Bladder training/testing performance

	Sensitivity	Specificity	Accuracy	AUC	RMS
Training	72.2222	90.1235	82.963	0.89712	0.3698
Testing	58.3333	79.4118	70.6897	0.79534	0.4094

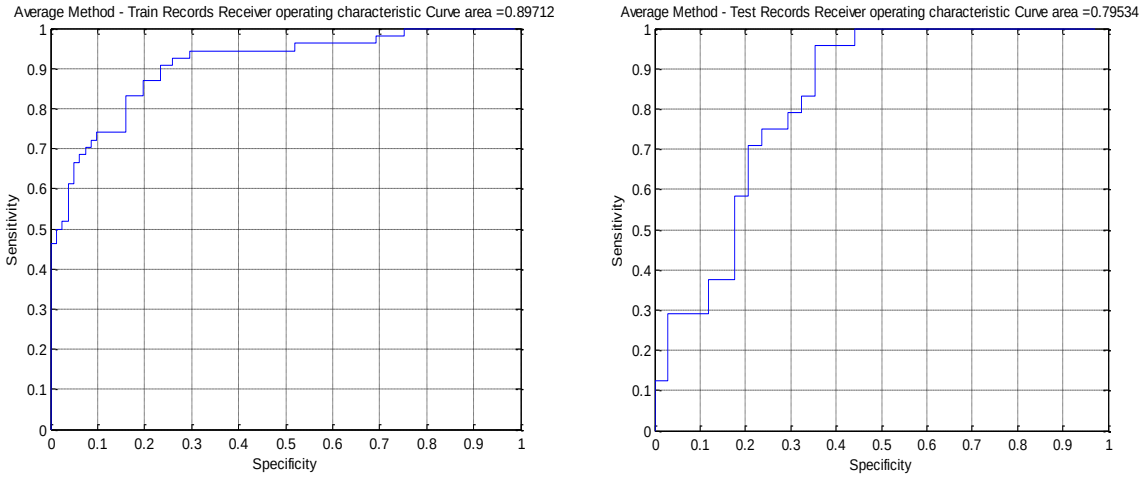


Figure 4.10: ROC training/testing of Bladder cancer records

So far, the ensemble model has been presented with training performance and testing performance. This has been done in order to measure the overall performance of the ensemble model by applying 7 data sets which are: Breast cancer, Statlog (Heart), Contraceptive, German Credit, Australian Credit and Ionosphere. Each data set was analysed using 5 different analysis parameters such as: Sensitivity, Specificity, Accuracy, AUC and RMS in order to take a look at the testing performance values of these parameters and to compare them with the training performance ones. The next step is to plot the tables in order to present graphs showing the performance of each data set training wise and testing wise.

The main objective of employing ensemble models is to reduce dataset error by taking the best result in ensemble. Through the combination of 10 different windows, this can result in

developing an ensemble model for datasets for the sake of improving the classification and prediction results.

Generally, the main trend that is observed when performing these experiments is that the accuracy of any ensemble design is far better than the best individual base classifier.

4.4 Summary

Ensemble learning has been proven to operate significantly better than conventional models. They have shown to give better results when there is a significant amount of diversity in the models. Most ensemble methods tend to encourage diversity among the models they combine. However, more random algorithms can be employed to generate stronger ensemble than non-random algorithms (Polikar, 2009).

Despite the fact that ensemble methods have not broken all of the barriers of applied science yet, it is known for a fact that the future of these techniques is very promising and will provide us with many benefits in all sorts of fields, especially scientific ones. The application of neural networks should be integrated with conventional mathematical operations in order to provide more success in pattern recognition and classification when it comes to comparing them to conventional methods. To become a strong and dependable field, researchers should always try to develop new models in order to manage and monitor the analysis of new complicated real-world problems (Papik, 1998).

Keeping in mind that the most powerful and systematic learning machine is our brains, machine learning researchers are paying much interest to applying ensemble learning in neuroscience and medicine so as to treat diseases, coming up with new theories and ways of monitoring the different organs and systems in the human body (Navone, 2001).

Chapter 5: Classifiers Combination

5.1 Introduction

Combining classifiers is a shared field of research among machine learning and statistical pattern recognition. It is known in different forms such as committees of learners, mixing experts, multi-classifier systems, etc. It is only logical that with the availability of different classifiers, combining them would improve the accuracy of the prediction. In addition, it is encouraged that the classifiers ought to be diverse. If they were identical, no improvement would be obtained from combining them. For that reason, diversity among the committee of learners has been recognised as an important issue. In theory, when the majority vote is used, a group of independent classifiers improve according to the single best classifier. However, a dependent set of classifiers can either be better than the independent set or maybe worse than the single worse member of the team. Thus, diversity can be beneficial or not, depending on the problem(Shipp and Kuncheva, 2002). Many techniques exist that are capable of improving the efficiency of classifier ensembles through modifying the data set on which classifiers are trained. These techniques are bagging, boosting and arching. They are considered as guidelines when generating classifier ensembles. The advantage of the methods over a simple grouping of independently trained classifiers is increasing the classification margins while reducing the variance of error. It is possible that these methods modify the diversity of the classifiers resulting in their success or sometimes their failure (Shipp and Kuncheva, 2002).

5.2 Combination Methods

5.2.1 Score Combination and Decisions Combination

Combination methods function on the outputs of each classifier and are classified into two categories. In the first type, the outputs are handled as inputs to a simple “generic” classifier

and the combination classifier algorithm is generated through training a "secondary" classifier. The main advantage behind using a generic combiner is that it is capable of learning the combination algorithm and thus automatically calculating the strengths and scores of each classifier. As for the second type, it depends on a function or a rule to combine the classifier score in a pre-defined approach.

The most important rule of a classifier combination is to generate a classifier functioning on the same type of inputs as the base classifiers and also to separate the same types of class. By supposing the score assigned to a class i by base classifier j is denoted by s_i^j, then the general combination rule is a function f and the final combined score is:

$$S_i = f\left(\{s_i^j\}_{j=1,\ldots,M}\right) \qquad (5.1)$$

The sample belongs to the classification of $\arg max_i S_i$. Therefore, the combination rules can be seen as a classifier functioning on scores of base classifiers including a particular combination function f and the $\arg max$ decision.

Conventional classifiers are not necessarily generated using the above method, however practically this is the method frequently used. For instance, in Multilayer Perceptron (MLP) classifiers, the last layer has nodes that contain each final score for a class. These scores are compared and the maximum will be chosen. In other words, a k-nearest neighbour classifier can generate scores for all classes in the form of the ratio of the number of representatives of a class in a particular neighbourhood to k. After that, the class containing the highest ratio will be assigned to a sample (Tulaykov and Jaeger, 2007).

5.2.2 Fixed Classifiers and Ensemble of Classifiers Combinations

One of the main parameter distinguishing a combination is to understand whether the combination uses a fixed set of classifiers (usually less than 10) compared to a large number of classifiers (probably infinite) where one assigns and thus generates new classifiers. The first type claims that classifiers are trained upon different features or sensor inputs. The main advantage is the diversity in the classifiers' strengths on different input patterns. While the second one claims large number of classifiers or the capability of generating classifiers. In other words, the large numbers of classifiers are made available through selecting multiple subsets of training samples from a single large training set or through selecting multiple

subsets of features from the available feature set and then training the classifiers depending on the assigned training subset or the subset of features (Tulaykov and Jaeger, 2007).

5.2.3 Classifiers Operation Level

Another method of grouping combination methods is dependent on the level at which they function. The first group of combinations operates at the feature level. The features of each classifier are combined to generate a joint feature vector and the classification is therefore established in the new feature space. The good thing about this method is that using features from two sets simultaneously can provide extra information about the classes. For instance, if two digit recognisers are combined in this method, and one of the recognisers employs a feature conveying the enclosed area, while the other has a feature of for example, number of contours, then combining these two features in a single recogniser will result in allowing class "0" to be easily separated from the other classes. In addition, if they operate individually, the first recogniser might face difficulty in distinguishing "0" from "8" and the second recogniser would face the same difficulty in separating "0" from "6" or "9". However, the downside of using this approach is that the large number of features requires a large training set in addition to a more complicated classification system. If the features used in different classifiers have no relevance to each other, then it is not necessary to perform a combination at the feature level.

Combinations can also function on a decision level. In other words, they employ the classifier outputs for generating combinations. This is a common method because knowing the inside structure of classifiers in addition to their feature vectors is not required. Despite the possibility that representational information is lost in these combinations, this is often compensated for by the simplicity of the combination method in addition to the superior training of the final system (Tulaykov and Jaeger, 2007).

As an application of utilising the feature level is handwriting recognition. Bertolami and Bunkei employed a handwriting recogniser that functions on a sliding window to obtain pixels and geometric features for the sake of hidden Markov model (HMM) matching (Tulaykov and Jaeger, 2007). At the feature level, the combination has a single HMM trained on the composite vector of these features. However, the combination at the decision level has two HMMs each solely trained on pixel and geometrical feature vectors and when word recognition occurs, the recognition results are combined along with the language model. The feature level combination has been shown to provide a better outcome.

5.2.4 Combined Classifiers Output Type

Another practical way of categorising combinations of classifiers is by the outputs of these classifiers. These levels of classification are:

- Abstract level: considered the lowest level due to the classifier providing the least amount of data in this level. As for the classifier output, it is a simple single class label or an unordered set of candidate classes.

- Rank level: in this level, the classifier output is an ordered sequence of candidate classes. The one positioned first is the most likely type of class, while one positioned at the end is the most unlikely. In addition, there are no confidence values included in the class labels in the rank level. However, their position in the n-best list shows their relative likelihood.

- Measurement level: in this level, the classifier output has confidence values appointed to each entry of the n-best list. These confidences can be arbitrary numbers according to the classification structure used. Therefore, the measurement level contains the most information compared to the other output levels (Tulaykov and Jaeger, 2007).

Basically, a certain combination method can function on any of these levels. For combinations depending only on label sets or class rankings (abstract and rank level), many voting methods have been introduced and analysed. The good thing about abstract and rank level is that different confidence characteristics do not affect negativity of the final output due to the fact that confidence has no control on the decision process. However, the confidence of a classifier in a certain candidate class often provides beneficial information that a simple class ranking cannot provide. This results in employing combination methods which function on the measurement level, capable of benefiting from the confidence provided to each candidate class. Recently, most of the classifiers provide information on the measurement level, so when applying combination methods on the measurement level, it can be more convenient for practical applications. However, it should be noted that each classifier may give out different confidence values with different parameters such as range, scale, mean, etc. This is not a huge issue for Bagging and Boosting as all classifiers have the same structure with different training sets, with the result that each classifier output is similar. On the other hand, if the classifiers have different classification structures, this output will be different.

If the combination includes classifiers with different outputs, then the output will have to be converted to one of the three levels of categorisation mentioned above, the abstract, rank and

measurement levels. As mentioned earlier as well, the most used classifier output level in combination research is the measurement level due to the availability of fixed-structure combinations such as sum of scores or the type of combinations that can be trained through the available training samples such as logistic regression, weighted sum, neural networks, etc. (Tulaykov and Jaeger, 2007).

5.2.5 Ensemble Combination Rules

Some of the algorithms have already built-in combination rules such as majority voting for bagging, separate classifiers for stacking etc. However, as mentioned earlier, ensembles can be trained by the following methods:

- Different subsets of the training data

- Different parameters of the classifiers

- Different subsets of the features (as in random subspace models)

The classifiers then go through combination using different combination rules. Some of them operate on only class labels while others require continuous outputs that can be considered as support given by the classifier to each class. There are three types of base model outputs for classifier combination which are:

- Abstract-level output: each classifier generates a unique class label for each input

- Rank-level output: each classifier generates a list of ranked class labels for each input

- Measurement-level output: each classifier produces a vector of class-related confidence values corresponding to the support for the potential classification hypotheses or a continuous valued vector measures corresponding to the class posterior estimates. For the abstract-level outputs, the decision of the t^{th} classifier is represented as:

$$d_{t,j} \in \{0,1\}, t = 1, \cdots, T; j = 1, \cdots, C \qquad (5.2)$$

where

T: number of classifiers

C: number of classes

If the t[th] classifier picks class ω_j, then $d_{t,j}$ would equal 1 and 0 for the combination rules that require continuous outputs. The outputs of the classifiers are known as T and C. As for combination rules that require continuous outputs, then the output of the classifiers are represented as $d_{t,j} \in [0,1]$. These outputs are often normalised so as to add up to 1, which can represent the normalised support given to class j by classifier t or the estimate of the posterior probability $P_t(\omega_j|x)$ (Polikar, 2006).

5.2.6 Algebraic Combiners

Algebraic combiners are defined as non-trainable combiners in which the classifiers' continuous outputs are combined via an algebraic expression. These expressions range from sum, product, mean, minimum, maximum, etc. Generally, the final decision $h(\boldsymbol{x})$ is in the class j that receives the largest support $\mu_j(\boldsymbol{x})$ after applying the algebraic expression to each support obtained by each class or in equation form (Polikar, 2006):

$$h_{final}(\boldsymbol{x}) = arg_j max \mu_j(\boldsymbol{x}) \qquad (5.3)$$

The arguments of the maxima (abbreviated arg max or $argmax$) are the points of the domain of some function at which the function values are maximized. In contrast to global maxima, referring to the largest outputs of a function, $arg\ max$ refers to the inputs, or arguments, at which the function outputs are as large as possible.

Note that $arg_j max$ (argument of the maxima) was devised in the final decision in order to refer to the maximum input which is the final class supports $\mu_j(\boldsymbol{x})$ in the class of j and they are computed as follows (Planetmath, 2016):

To find the mean:

$$\mu_j(x) = \frac{1}{T}\sum_{t=1}^{T} d_{t,j}(x) \qquad (5.4)$$

For T number of classifiers ,the sum is calculated as follows:

$$\mu_j(x) = \sum_{t=1}^{T} d_{t,j}(x) \qquad (5.5)$$

this equation gives an identical final decision as in the mean equation

The weighted sum equation as follows:

$$\mu_j(x) = \sum_{t=1}^{T} \omega_t d_{t,j}(x) \qquad (5.6)$$

where ω_t corresponds to the weight given to the *t*th classifier *ht* according to some performance criteria

For finding the Product:

$$\mu_j(x) = \prod_{t=1}^{T} d_{t,j}(x) \qquad (5.7)$$

For a T number of classifiers, to know the maximum value of the inputs:

$$\mu_j(x) = max_{t=1,\cdots,T}\{d_{t,j}(x)\} \qquad (5.8)$$

As for the minimum inputs:

$$\mu_j(x) = min_{t=1,\cdots,T}\{d_{t,j}(x)\} \qquad (5.9)$$

For finding the median rule, this equation is used:

$$\mu_j(x) = \mu_j(x) = med_{t=1,\cdots,T}\{d_{t,j}(x)\} \qquad (5.10)$$

As for the generalisation mean rule, the equation used is shown below:

$$\mu_{j,\alpha}(\boldsymbol{x}) = \left(\frac{1}{T}\sum_{t=1}^{T} d_{t,j}(x)^a\right)^{\frac{1}{a}} \qquad (5.11)$$

where α is a scalar value range from

- $\alpha \rightarrow -\infty \rightarrow$ Minimum equation

- $\alpha \rightarrow \infty \rightarrow$ Maximum equation

- $\alpha = 0 \rightarrow$ Geometric mean equation

- $\alpha = 1 \Rightarrow$ Mean equation

5.3 Combination Methods Theory

The proposed method is ensemble combination and is divided into 10 models. The results obtained are input in a committee machine, and by taking the best results in ensemble, the records are divided into 10 block window groups via ANN models for n times. Finally, the best results are found by taking the average of the 10. This method is done for each ensemble, then are combined into the committee machine by choosing the best results. All these steps are done through a classifier with n predictor.

Moreover, the process of performing combination methods is done by combining the outputs for each classifier and the predictors are processed by summing each predicted point for each classifier then dividing the number of classifiers, in this case 3, and each of them is predicting an output point. After that, based on that method applied for example average method, an average of the three predictions is taken then compared to the actual output.

5.3.1 Averaging Method

Ensemble averaging is defined as the method of generating and combining multiple models in order to arrive at the desired output instead of just creating one model. Generally, the ensemble of models gives out better results than individual models due to the fact that the multiple errors of the models are averaged. Unlike the standard network design which depends on generating many networks and keeping just one, ensemble averaging tends to hold the poor performance networks and give them fewer weights. In ensemble averaging, there are two parameters which should be fulfilled. The first one is that the bias can be decreased on the count of increasing variance. The second parameter implies that in a group of networks, the variance can be deceased without affecting the bias.

Generally, ensemble averaging generates a group of networks and each of these networks has low bias and high variance. Therefore, it resolves the issue of the bias-variance dilemma in machine learning. In detail, the idea is to create a set of experts with alternating parameters. These parameters are considered as initial synaptic weights, although some additional factors such as momentum, learning rate, etc., may also be altered. Furthermore, the steps of generating the average are as follows:

First of all, the researcher generates N experts, with each expert having their own initial value and these values are often picked from a distribution randomly. After that, each expert is trained solely and finally, the experts are combined and their values are averaged.

A comprehensive model of ensemble averaging represents the final result, not as a good average of all the experts, but as a weighted sum (Naftaly, et al., 1997).

Supposing each expert is y_i, the overall average $\tilde{y}$ is:

$$\tilde{y}(\boldsymbol{x}; a) = \sum_{j=1}^{p} a_j y_j(\boldsymbol{x}) \qquad (5.12)$$

where a represents a set of weights.

The optimisation issue of knowing the value of a is resolved by neural networks. Therefore, a meta-network or a network that has neurons that are specifically trained neural networks is resolved. In addition, the synaptic weights are the weight that is applied to each expert. This method is known as linear combination of experts (Naftaly, et al., 1997).

Most forms of neural networks are a kind of linear combination, for example, the standard neural network that uses a single expert is basically a linear combination with $a_j = 0$ and one $a_k = 1$. Furthermore, a raw average has all a_j having constant values, basically one over the number of experts.

Advantages:

The resulting committee of learners is often simpler than a single network with the same preference level.

The output can be easily trained on small set of inputs.

The output often shows better performance compared to single networks.

The problem of over fitting is decreased with less weights needed to be set.

Now, the principle of ensemble averaging is based on the existing of independent statistical event. To further explain the principle, imagine a number of individuals flipping unbiased coins at the same time. If head was 1 and tail was 0, then the arithmetic average is:

$$X_n = \frac{1}{N}\sum x_n \qquad (5.13)$$

where n^{th} flip is x_n and N corresponds to the total number of flips. If all the coins were the same, it doesn't matter if one coin is flipped N times or N coins flipped a single time. The main thing is that they must be independent events. In other words, the probability of a head or a tail in a flip must be independent of the other flips. By taking a random variable x approach, the n^{th} times of x is x_n and thus the ensemble average of x is X or $\langle x \rangle$ shown as:

$$X = \langle x \rangle = \lim_{N \to 1} \frac{1}{N} \sum_{n=1}^{N} x_n \qquad (5.14)$$

Accordingly, it is impossible to obtain the ensemble average in practice, due to the fact that an infinite number of independent trials is impossible. The closest thing achievable is the arithmetic mean for the number of trials available. For that reason, the arithmetic mean is known as the estimator in ensemble averaging. The idea of ensemble averaging is quite useful despite the fact that it cannot be obtained. Specifically, it can always be assured that ensemble average exists even if it is only estimated. However, that does not mean that it is practically easy to obtain (George, 2011).

A well-known example of ensemble averaging is face recognition average. The human capability to acknowledge and recognise objects cannot be remotely matched with intelligent machines. For example, it takes a person a small glance to recognise someone else's "familiar" face. Face recognition has recently gained interest among researchers with the aim of developing better recognition by machines. The current face recognition method depends on recognising current face features.

Depending on the human's glance and face familiarity, the glance effect is analysed using image pre-processing and pattern averaging. When humans glance at faces, they do not focus on the detailed features but instead obtain a general impression of the face. In machine training, this can be analysed by averaging the face image rather than looking for face features. The averaged models thus represent the face excluding the facial expressions and orientations. The glance effect is preceded by face familiarity, and thus is done by neural network training using different orientations of face images. Recently, an intelligent recognition system scored a rate of recognition accuracy of 98.99% using 90 face images of

30 individuals in multiple orientations. This system enjoyed a quick runtime of 0.21 seconds (Khashman, 2006).

5.3.2 Weighted Average

Ensemble methods add up the outputs of several neural networks and combine them. This output is a weighted average of network outputs. In addition, the ensemble weights are represented as a function of the relative error of each trained network and thus the resulting network provides better results than each of the combined networks. A lot of research has been done on ensemble methods in order to develop performance improvements, leading to training the networks to de-correlate with one another with respect to their errors. In weighted average, the ensemble weights are dynamically determined. In other words, the weights are determined in each propagation in the network, unlike in static determination. The weights are proportional to the certainty of the respective outputs. This certainty measures how the output is close to any of the target values (Jimenez, 1988).

Supposing the function computed by the i^{th} network is $f_i(x)$. The networks are programmed to give out 0 or 1 for a negative or a positive classification with a threshold of 0.5 on the network output to determine the class for an instance of the problem. This approach is known as the naïve approach and it is often used as a cross-validation set $CV = (x_l, y_l)$ and for choosing the network f_{Naive} in which the MSE (mean square error) is minimised on CV. Therefore, the mean square error for each network is represented as:

$$MSE\ [f_i] = E_{CV}\left[\left(y_m - f_i(x_m)\right)^2\right] \qquad (5.15)$$

The purpose of this method is to exclude any knowledge within the other networks, despite the fact that f_{Naive} enjoys the best performance on the cross-validation set. However, some of the f_i's may result in correct classification while f_{Naive} does not (Jimenez, 1988).

A conventional approach in output network combination is in averaging them, as discussed earlier. However, this approach does not take into consideration the fact that some networks might be more accurate than others. The generalisation method is aimed at finding the weight for each output that reduce the ensemble's MSE. The generalised ensemble method (GEM) is represented as:

$$F_{GEM} = \sum_{i=1}^{n} a_i f_i(x) \qquad (5.16)$$

where a_i are picked to minimise MSE with respect to f or the target function estimated using the CV set and sum to 1. The error of the network f_i is:

$$\varepsilon_i(x) = f(x) - f_i(x) \qquad (5.17)$$

as for the correlation matrix:

$$C_{ij} = E[\varepsilon_i(x)\varepsilon_j(x)] \qquad (5.18)$$

now, the a_i that minimises the MSE is determined through:

$$MSE[f_{GEM}] = \sum_{i=1}^{n}\sum_{j=1}^{n} a_i a_j C_{ij} \qquad (5.19)$$

from the previous work, the optimal choice for a_i:

$$a_i = \frac{\sum_{j=1}^{n} C_{ij}^{-1}}{\sum_{k=1}^{n}\sum_{j=1}^{n} C_{kj}^{-1}} \qquad (5.20)$$

The GEM shows better performance than the conventional averaging approach in addition to giving better estimations when compared to the naïve classifier. However, GEM depends on a proper estimate of C and the chance that C is non-singular so as to be inverted. Practically, errors are highly correlated and as a result the rows of C are nearly linearly independent so when C is inverted, a significant round off of errors takes place. However, there are some methods which avoid this issue, including ignoring the networks with highly correlated errors, in other words, employing special methods for inverting near-singular matrices and then training the networks to correlate with one another.

Basically, the output of a neural network in the form of $y = f_i(x)$ can be considered as the probability of an instance x within a class. As y approaches 1, it can be assured that the instance is present in the class, while if it approaches 0, it is not present in the class. Thus, the certainty $c(y)$in the output of a neural network is represented as:

$$c(y) = \begin{cases} y \; if \; y \geq \frac{1}{2} \\ 1 - y \; othewise \end{cases} \qquad (5.21)$$

According to the above equation, the certainty increases for outputs y less than 0.5 as y falls and for outputs above or equal to 0.5 as y rises. In addition, one network output y_1 is less certain than another output y_2 if $c(y_1) < c(y_{2)}$. This leads to the idea that the certainty is symmetrical when it comes to positive and negative decisions. In other words, the certainty of 0.1 is the same as of 0.9; only the decision that these outputs are certain about is different (Jimenez, 1988).

Instead of using static weights from the input space performance of f_i, the weights are modified to become proportional to the output certainties leading to providing better performance. This weighted average network is defined as:

$$F_{DAN} = \sum_{i=1}^{n} \omega_i f_i(x) \qquad (5.22)$$

and ω_i is defined as:

$$\omega_i = \frac{c(f_i(x))}{\sum_{j=1}^{n} c(f_i(x))} \qquad (5.23)$$

According to the above equations, the ω_i's add up to 1, that is why F_{DAN} is considered as the weighted average of the outputs. The difference here is that the weight vector is recomputed each time the output is evaluated. In this way, it provides the best decision in a certain instance instead of choosing weights statically by giving an optimal decision depending on a CV set. In addition, each contribution from the network to the sum is proportional to its certainty. This technique is similar to the principle of using agreement for a set of classifiers to ensure certainty in the decision, but the certainty of each classifier contributes to reaching the final decision (Jimenez, 1988).

5.3.3 Optimised Weighted Average

Optimised weighted average, or Multi-objective optimisation or Pareto optimisation is a method of multiple decisions making that is based on the principle of mathematical optimisation including multiple objective functions being simultaneously optimised. It is

concerned with applications requiring optimal decisions to be instated in addition to considering some trade-offs among two or more conflicting parameters. Maximising battery life and minimising the cost of a smart phone is a simple example of a multi-objective optimisation involving two objectives. However, in practice, more than three objectives are analysed. For a problem of non-trivial multi-objective nature, no single solution exists that will optimise each objective at the same time. The objective functions are thus conflicting and possibly an infinite number of Pareto-optimal solutions might exist. Therefore, a solution is known as Pareto-optimal if the objective functions cannot be improved in value without decreasing the value of the other objectives (Yaochu, 2008).

Machine learning is considered as a multi-objective task. Either one objective is employed as a cost function or multiple objectives are combined to a scalar cost function. This can be related to the idea that most learning algorithms can deal with a scalar cost function only. Pareto-based multi-objective learning is proved to be more powerful and efficient in comparison to learning algorithms with scalar cost functions, with subjects such as clustering, knowledge extraction, ensemble generation, etc.. One advantage of this approach is that a more detailed insight into the problem itself can be obtained through the analysis of the Pareto front comprising multiple Pareto-optimal solutions. It is only recently, about a decade ago, that the Pareto approach to address multiple objectives was applied in machine learning. This was due to the fact that conventional learning algorithms and optimisation algorithms are relatively not so efficient when it comes to solving multi-objective problems. In this method, the objective function is a vector instead of a scalar. Consequently, a number of Pareto-optimal solutions are obtained instead of one solution (Yaochu, 2008).

Learning is in general multi-objective. In supervised learning, although the objective of memorising the training data is the most significant one, it is still not the only objective. Many other objectives should be taken into account in supervised learning. One of the main objectives is that in order to prevent the model from the issue of overfitting of data, the model's complexity should be monitored. In addition, objectives such as comprehensibility and interpretability should be taken into account, especially when supervised learning is targeting the knowledge discovery from data. Moreover, interpretability in models significantly depends upon the complexity of the model. In other words, the lower the complexity, the easier it is to comprehend the model. Therefore, controlling the complexity requires combining two objectives into a scalar objective function, represented as follows:

$$f = E + \lambda\Omega, \qquad (5.24)$$

where,

E: Common error function

Ω: Measurement of the complexity of the model (e.g. the number of parameters in the model)

λ: A hyperparameter predefined by the user (λ>0)

In this way, the model is capable of optimising two objectives while taking in to consideration the operation of a scalar function.

Pareto optimality is the most essential aspect in this method. Taking a look the m-objective minimisation problem:

$$min\, F(X), \qquad (5.25)$$

$$F = \{f_1(X), f_2(X), \ldots, f_m(X)\} \qquad (5.26)$$

According to the above equation, a solution X can dominate a solution Y if $\forall j = 1, 2, \ldots, m, f_j(X) \leq f_j(Y)$, and there exists $k \in \{1, 2, \ldots, m\}$, such that $f_k(X) < f_k(Y)$. Solution X is considered as Pareto-optimal if it is not dominated by another solution. As mentioned earlier, there is often more than a single Pareto-optimal solution when the objectives conflict one another. The surface containing the Pareto-optimal solutions is called the Pareto front. Practically, it is not known where this Pareto front resides in a certain problem. As a result, non-dominated solutions produced by a multi-objective ensemble averaging do not have to be considered as Pareto-optimal. On the other hand, non-dominated solutions produced by multi-objective optimisation algorithms are lightly considered as Pareto-optimal solutions. As mentioned above, Pareto-based multi-objective learning depends on the method of Pareto-based multi-objective optimisation to deal with learning problems. The scalar bio-objective of equation 5.24 can be modified as a Pareto-based multi-objective optimisation:

$$min\{f_1, f_2\} \qquad (5.27)$$

$$f_1 = E \qquad (5.28)$$

$$f_2 = \Omega. \tag{5.29}$$

Thus, the most frequently utilised measure of error is the MSE. The parameter of complexity in a neural network can either be the sum of the squared weights or the sum of the absolute weights shown in the equations below:

$$\Omega = \sum_{i=1}^{M} w_i^2 \tag{5.30}$$

$$\Omega = \sum_{i=1}^{M} |w_i| \tag{5.31}$$

where $w_i, i = 1, \ldots, M$ is the weight and M is the total number of weights. It is still necessary for the user to choose one or more than a solution from the obtained Pareto-optimal solutions depending on the user's choice after learning (Yaochu, 2008).

5.3.4 Voting

Voting algorithms have been widely used in control systems, image processing, pattern recognition and human organisation systems in order to decide on redundant software versions and hardware models. Generally, voting methods can be classified to agreement based voters such as majority and plurality and to voters that generate output in spite of the existence of the agreement among the redundant variables results. Sometimes it is essential to employ the second type including the median and weighted average. Both of these voters are the most popular ones used in applications. However, the weighted average was proven to be more efficient than the median. While the median voter generally selects the mid-value of results, the weighted average operates systematically by appointing weight to each input depending on their pre-chosen priority. This is done for the sake of sharing more trustable inputs and excluding low inputs with lower probability of correctness (Zarafshan, et al., 2010).

Voting-based methods operate on labels only, where $d_{t,j}$ is 1 or 0 depending on whether classifier t chooses j, or not, respectively. The ensemble then chooses class J that receives the largest total vote (Rokash, 2010):

$$\sum_{t=1}^{T} d_{t,J}(x) = max_{j=1,\cdots,C} \sum_{t=1}^{T} d_{t,j} \tag{5.32}$$

If the classifier outputs are independent, the majority voting combination will always result in performance enhancement. For a two-class problem, if there are a total of T classifiers, the decision will be correct only if at least $\lfloor T/2+1 \rfloor$ classifiers pick the correct class. Suppose each classifier has a probability p of producing a correct decision. The ensemble's probability of a good decision will be a binomial distribution. In other words, if the probability of picking $k > \lfloor T/2+1 \rfloor$ correct classifiers from T is:

$$P_{ens} = \sum_{k=(T/2)+1}^{T} \binom{T}{k} p^k (1-p)^{T-k} \qquad (5.33)$$

Then,

$P_{ens} \to 1$,

When T approches ∞ if $p > 0.5$, and

$P_{ens} \to 0$,

In the case of T approaching ∞ if $p < 0.5$.

Notice that the condition of $p > 0.5$ is integral and important for a two-class problem. It is also sufficient for multi-class problems however it is not that necessary (Rokash, 2010).

Majority voting Generally, generating a neural network classifier suffers the lack of scalability. With the increase in dimension or complexity or both in a classification problem, the ability to generate a neural network decreases due to the tight coupling between the weights. This will lead to crosstalk and partial over-learning to under-learning, in addition to a decrease in the efficiency and speed in learning. However, neural networks architectures have been recently improved to solve this issue. The basic principle is to decompose complex classification tasks into simpler tasks and to solve each of these simple tasks using simpler modules. However, the overall performance of the classification depends on the success of integrating the "module decisions" into a global one representing the whole network. Module decision making is done by either a competitive or a cooperative method. "Voting" is known to be the most popular method for multi-module cooperation. Activating the output of each module representing a particular candidate decision is known as a "vote". The final decision depends on all the activations through a voting technique (Russell and Rubin, 2009).

According to the voting system represented in figure 5.1, the main goal is to assume to correct an unknown sample group depending on the information provided in the output of the modules (voters). Some voting systems are considered and evaluated depending on the accuracy of the classification. The higher the accuracy, the better the representation of information in the bids. That in particular is the main goal of voting system used in real-life elections (Russell and Rubin, 2009).

However, some criteria should be taken into consideration in networks such as:

- The highest activation is the best guess of the network and if not, the second one is considered the second best guess and so on. Most importantly, the correctness probability is not dependent on the input values. From previous research, the probability of having the correct outcome as the fourth best guess is almost zero if their values are 0.2 to 1.0. Otherwise, the network would require further training. In conclusion, despite the fact that the "order" of the output carries information of the correct class, the lowest output values are seen as information noise.

- Some voting systems cannot be applied in neural network systems due to the points mentioned above. In other words, in the negative voting system, each voter provides a single negative vote for a single candidate. The candidate having the lowest votes is considered the winner, while approval voting gives the opportunity to each module to give a positive vote for a number of alternatives and the highest number of votes is considered the winner. To focus on the preference, votes ought to be given to groups which are above the average point of the "bid's intensities". Disapproval voting is considered the opposite of this system and the two systems can be combined to provide another voting scheme. Despite their wide use in real-life applications, applying them in neural networks will be sensitive to the lowest value as they are proven to carry little to no information (Russell and Rubin, 2009).

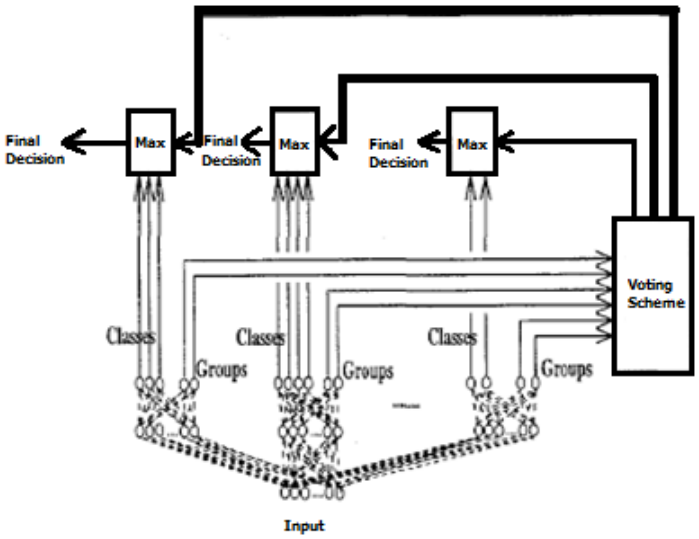


Figure 5.1: The winner module is decided through a voting system prior to deciding upon the class (Russell and Rubin, 2009)

5.4 Experimental Results

In this section, the analysis of each method will be discussed, in addition to the basic parameters affecting each method. Basically, there are 4 combination methods in this section and each of these methods will be applied on the 7 data sets. Based on that data, the model(s) that deliver the best results will be concluded.

Three methods were employed to obtain the results; 90% training to 10% testing, 80% to 20% and 70% to 30%, respectively. As a result, the second method which is 80% training to 20% testing was proven to be the best among the three methods for it provides the best results.

Basically, the average method is the sum of the ANNs divided by the number of ANNs models, in this case are 3. Then each data is trained 10 times then the average of networks are taken. as for weighted average and optimized average ANNs, optimizers are added. These optimizers play a role in tuning the weights of the weighted average and optimized weighted average.

5.4.1 Average Method

Table 5.1 shows the performance results of the training experimental data sets of the average combination method, compared to the actual data. Table 5.2 shows the performance results of the testing set. The data in the testing set are there to predict the data sets' output. In the experimental results, the data records have been divided into 80% for performance training and 20% for performance testing. This division has been chosen as it delivers the best results.

Furthermore, for training and testing, different percentages for each proportion was made. For example, the ratio of 90% of training to 10% testing was made in addition to 70% training to 30% testing. However, the best performance was obtained from having 80% for training and 20% for testing, therefore we have utilized the 80 to 20 percentage as a default method throughout the research.

In each table, the rows represent the 7 data sets which are: Breast Cancer Bladder Cancer, Statlog (Heart), Contraceptive, German Credit and Australian Credit. In addition, the columns represent the analysis parameters which are: Sensitivity, Specificity, Accuracy, AUC and RMS. Each cell contains a number which refers to a percentage (for the first three columns) while the other two columns represent measurements of the area under the curve and the root mean square.

Table 5.1: Experimental data training performance

	Sensitivity	**Specificity**	**Accuracy**	**AUC**	**RMS**
Breast Cancer	93.865	99.0476	97.2803	0.85605	0.1797
Bladder Cancer	72.7273	88.75	82.2222	0.90443	0.3654
Statlog (Heart)	92.9412	97.1154	95.2381	0.9664	0.2516
German Credit	58.6047	96.701	85	0.92694	0.3461
Australian Credit	100	90	96.3115	0.95993	0.2127
Ionosphere	100	90	96.3115	0.9761	0.2131
Contraceptive	99.6581	63.4703	84.1642	0.88259	0.4166

Table 5.2: Experimental data testing performance

	Sensitivity	Specificity	Accuracy	AUC	RMS
Breast Cancer	93.3333	99.2248	97.0588	0.82786	0.1685
Bladder Cancer	60.8696	80	72.4138	0.69006	0.4310
Statlog (Heart)	94.2857	93.4783	93.8272	0.95248	0.2791
German Credit	94.2857	93.4783	93.8272	0.79275	0.3991
Australian Credit	100	91.4286	97.1429	0.97061	0.1843
Ionosphere	100	91.4286	97.1429	0.97061	0.1754
Contraceptive	93.1727	39.4737	69.9317	0.72402	0.5271

Figures 5.2 to 5.8 show the ROC of the training experimental data sets and the ROC curve of the testing data set (predictors of output data records). Each of the 7 data sets mentioned above have both trained and tested figures and each figure represents a plot of Specificity vs. Sensitivity. According to the results, the best prediction would yield a point in the upper left corner or coordinate (0,1) of the ROC space, representing both 100% sensitivity (no false negatives) and 100% specificity (no false positives). In addition, the (0,1) point is considered a perfect classification. In the ROC plot, when the curve is nearly touching the left top corner coming closer to 1, that means that this ROC enjoys high accuracy and when it goes down that corresponds to low accuracy ROC. Therefore, the more higher and closer to the corner the curve is, the better results are obtained in terms of accuracy.

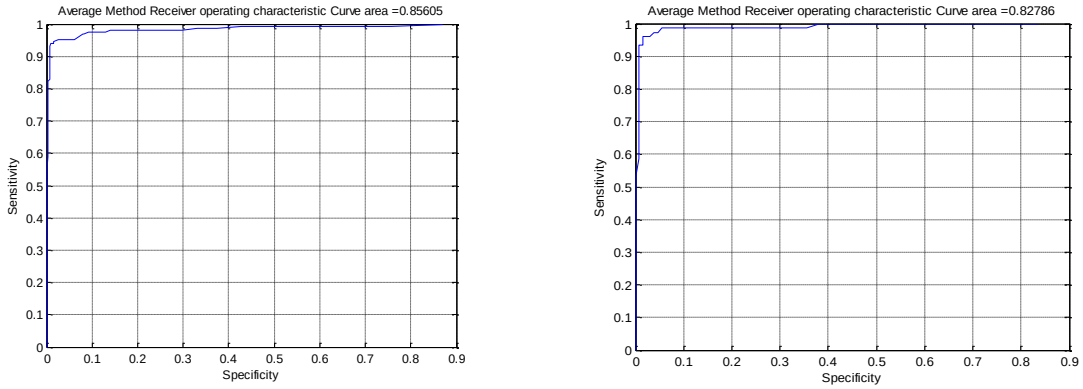


Figure 5.2: ROC training/testing of Breast Cancer records

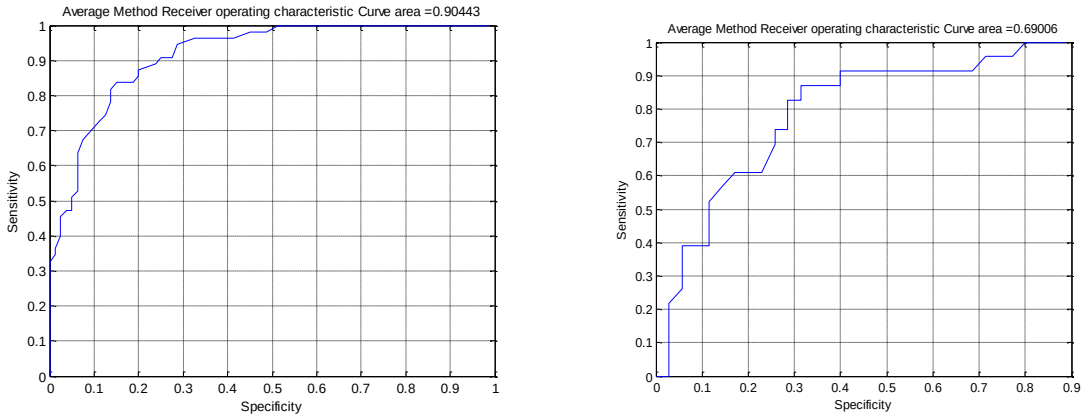


Figure 5.3: ROC training/testing of Bladder Cancer records

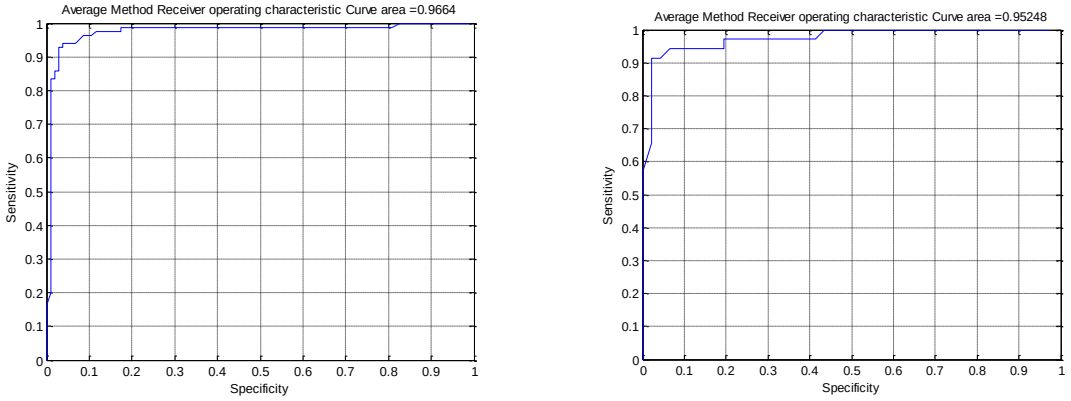


Figure 5.4: ROC training/testing of Statlog (Heart) records

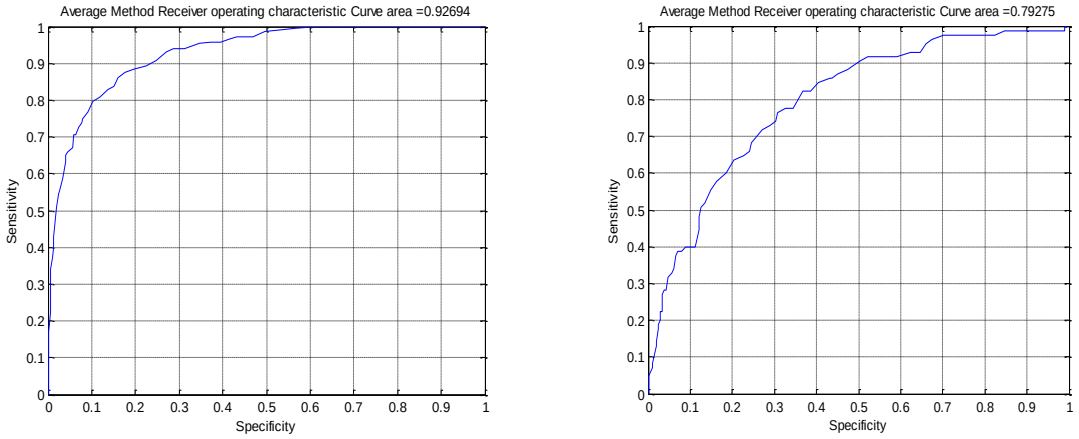


Figure 5.5: ROC training/testing of Contraceptive records

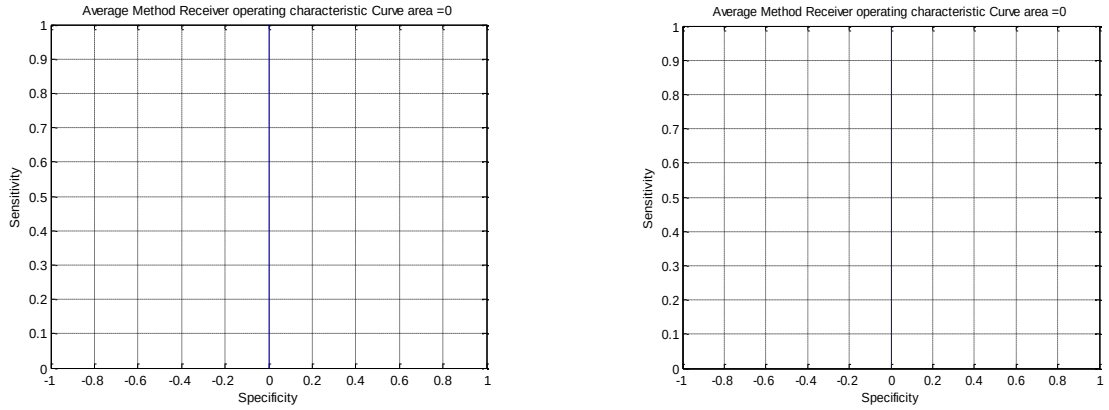


Figure 5.6: ROC training/testing of German credit records

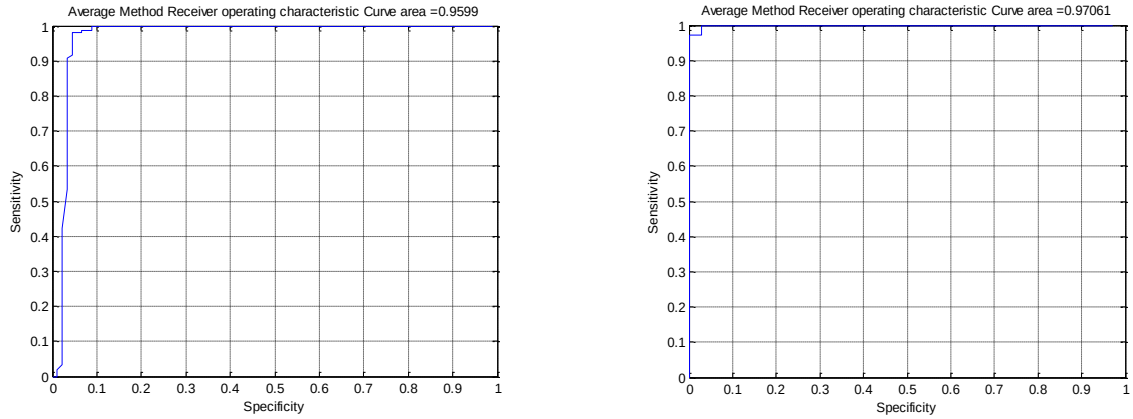


Figure 5.7: ROC training/testing of Australian credit records

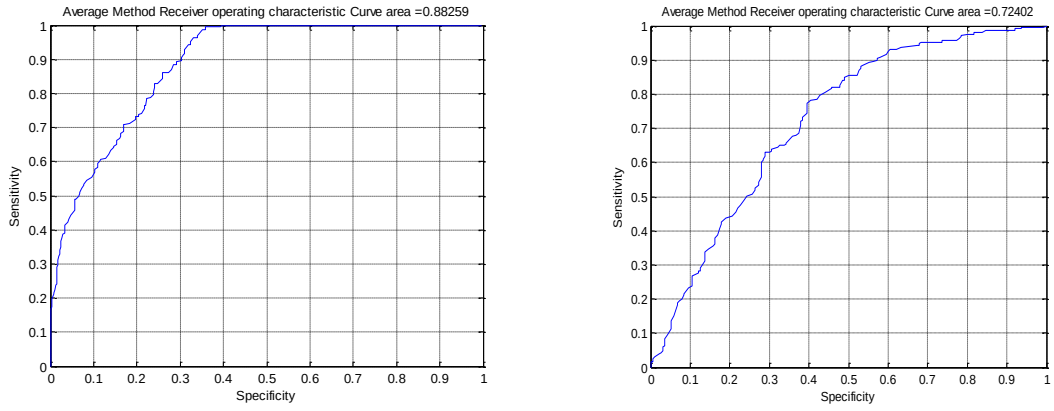


Figure 5.8: ROC training/testing of Ionosphere records

5.4.2 Weighted Average

The weights in the weighted average are the result of summing the outputs of multiple neural networks. The weighted average is similar to the conventional average, however instead of

each data point contributes to the final average equally, some data points contribute more than others in the final output. These weights are represented correspondingly as a function of relative error for each network and that is why the resulting network delivers better results when it is compared to each network combined in a certain problem. The weights are dynamically determined in each propagation. They are proportional to the certainty of the outputs, in which this certainty is basically a representation of how the output presented is close to any of the desired values. To measure the weights, the first process is to find the minimum squared error (MSE) of each network in which these weights are dependent on in representation. This MSE in turn filters any irrelevant knowledge of other networks.

The main difference between the conventional generalised ensemble method F_{GEM} and the weighted average is, most importantly, that dynamic weights are used from the input space f. Moreover, the weight vector ω_i is recomputed each time the output is evaluated. As a result, this ensures that the best decision is received in a given instance rather than a static choice of weights. This provides an optimal decision depending on a cross validation set (CV). Therefore, each contribution is proportional to the certainty so it behaves as an agreement to assure certainty and accuracy in decision making. The weights of the combiner are as follows: [0.39 0.33 0.27]

Table 5.3 shows the performance results of the training experimental data sets of the weighted average combination method compared to the actual data, while Table 5.4 shows the performance results of the testing set. The data in the testing set are there to predict the data sets' output. In the experimental results, the data records have been divided into 80% for performance training and 20% for performance testing. This division has been chosen as it delivers the best results.

In each table represented below, the rows represent the 7 data sets which are: Breast Cancer Bladder Cancer, Statlog (Heart), Contraceptive, German Credit and Australian Credit. In addition, the columns represent the analysis parameters which are: Sensitivity, Specificity, Accuracy, AUC and RMS. Each cell contains a number which refers to a percentage (for the first three columns) while the other two columns represent measurements of the area under the curve and the root mean square.

Table 5.3: Experimental data training performance

	Sensitivity	Specificity	Accuracy	AUC	RMS
Breast Cancer	93.865	99.0476	97.2803	0.8765	0.1792
Bladder Cancer	72.7273	88.75	82.2222	0.90818	0.3605
Statlog (Heart)	94.1176	97.1154	95.7672	0.96697	0.2462
German Credit	58.1395	96.701	84.8571	0.92693	0.3461
Australian Credit	93.1507	86.1905	89.7436	0.96941	0.2645
Ionosphere	100	90	96.3115	0.96011	0.2126
Contraceptive	99.4889	63.242	84	0.88482	0.4175

Table 5.4: Experimental data testing performance

	Sensitivity	Specificity	Accuracy	AUC	RMS
Breast Cancer	93.3333	99.2248	97.0588	0.84284	0.1665
Bladder Cancer	65.2174	80	74.1379	0.76708	0.4237
Statlog (Heart)	94.2857	95.6522	95.0617	0.95466	0.2542
German Credit	38.8235	92.093	77	0.79327	0.3970
Australian Credit	94.3182	82.2917	88.0435	0.92525	0.3122
Ionosphere	100	91.4286	97.1429	0.97122	0.1824
Contraceptive	93.1174	40.9574	70.5747	0.7122	0.5580

Figures 5.9 to 5.15 show the ROC of the training experimental data sets and the ROC curve of the testing data set (predictors of output data records). Each of the 7 data sets mentioned above have both trained and tested figures and each figure represents a plot of Specificity vs. Sensitivity. According to the results, the best prediction would yield a point in the upper left corner or coordinate (0,1) of the ROC space, representing both 100% sensitivity (no false negatives) and 100% specificity (no false positives). In addition, the (0,1) point is considered a perfect classification. In the ROC plot, when the curve is nearly touching the left top corner

coming closer to 1, that means that this ROC enjoys high accuracy and when it goes down that corresponds to low accuracy ROC. Therefore, the more higher and closer to the corner the curve is, the better results are obtained in terms of accuracy.

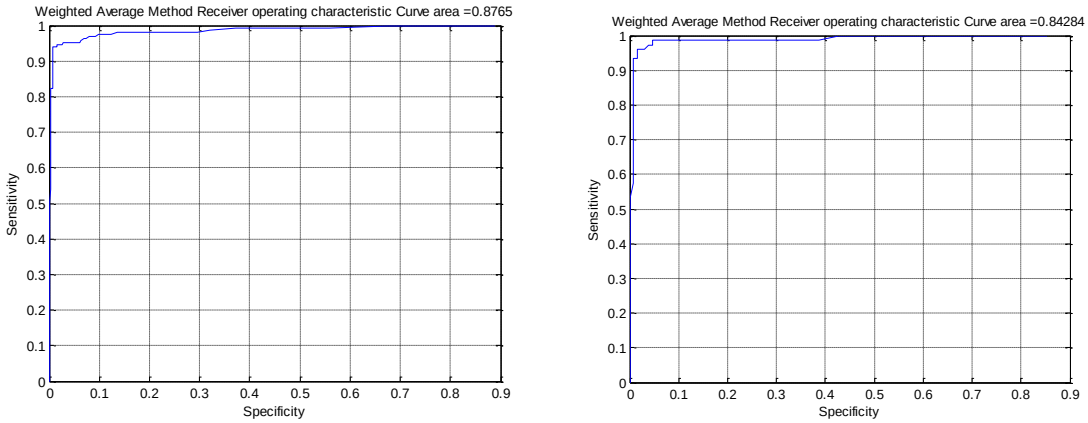


Figure 5.9: ROC training/testing of Breast cancer records

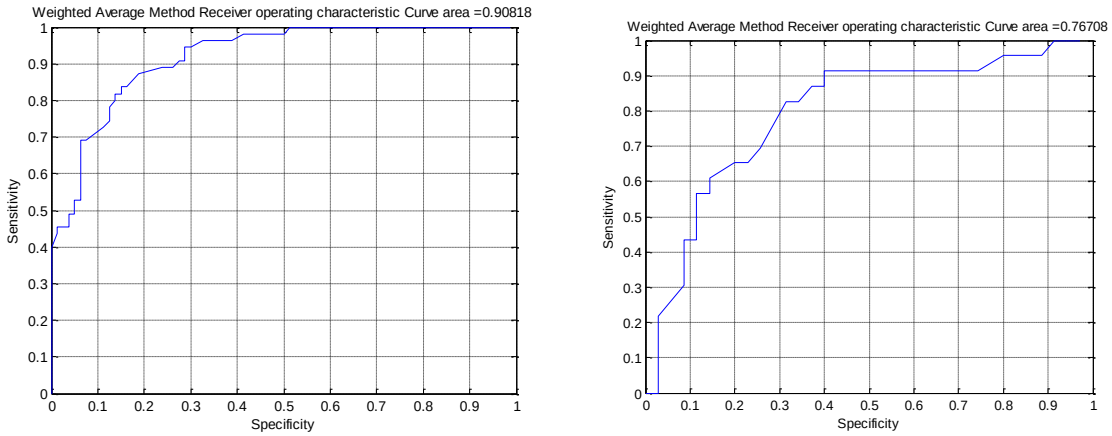


Figure 5.10: ROC training/testing of Bladder cancer records

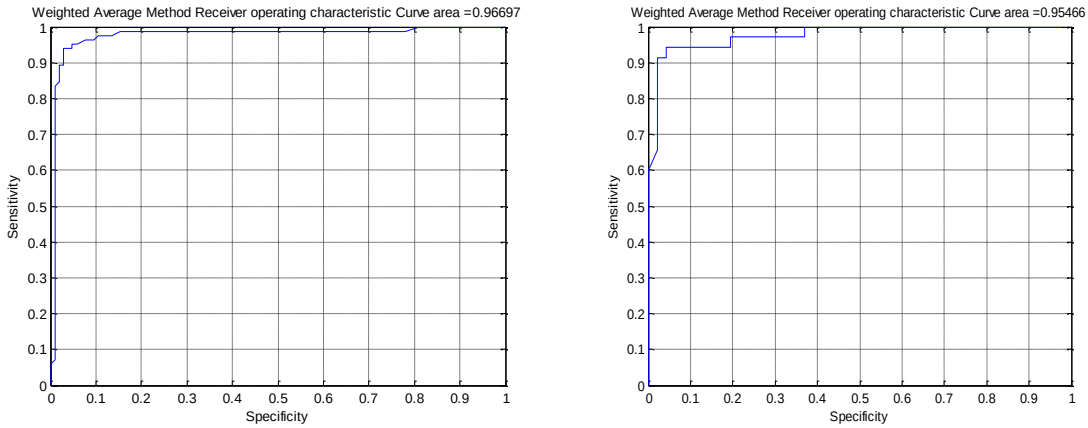


Figure 5.11: ROC training/testing of Statlog(Heart) records

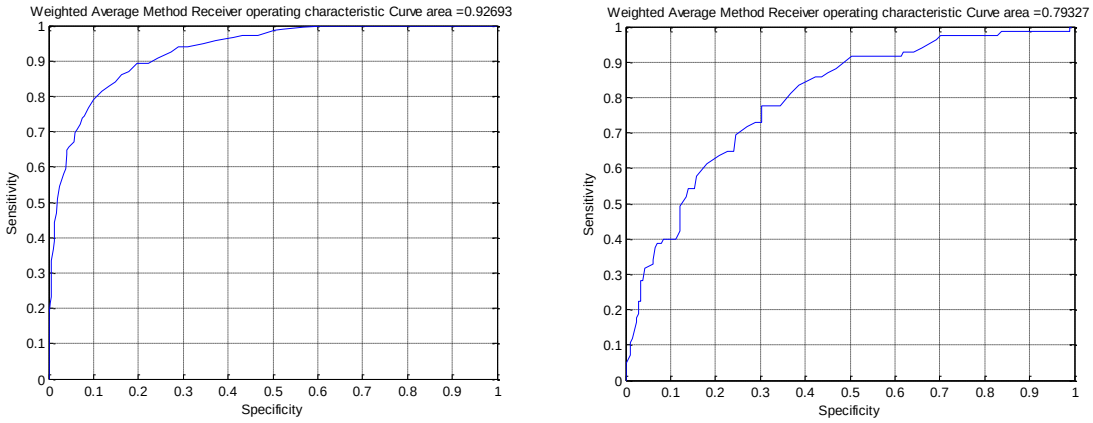


Figure 5.12: ROC training/testing of Contraceptive records

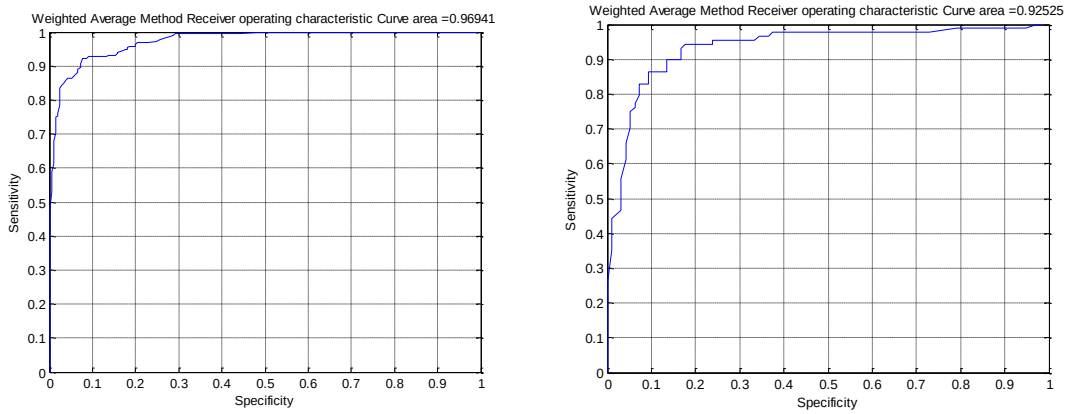


Figure 5.13: ROC training/testing of German credit records

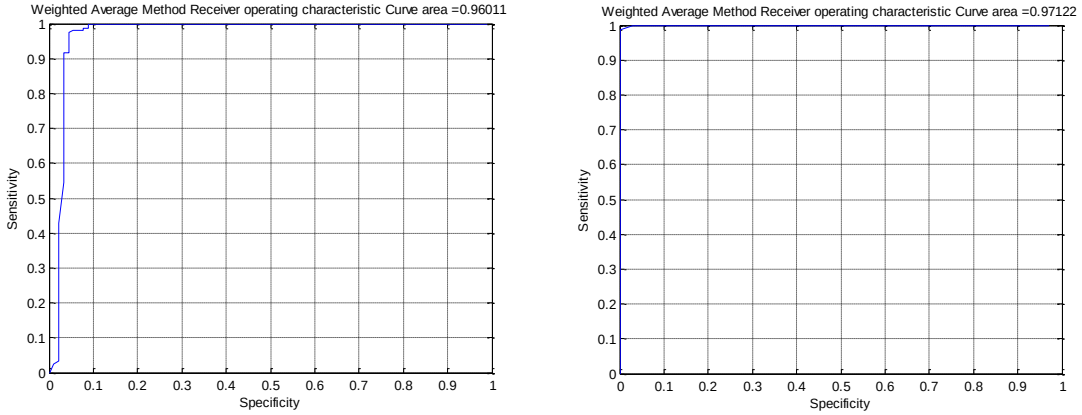


Figure 5.14: ROC training/testing of Australian credit records

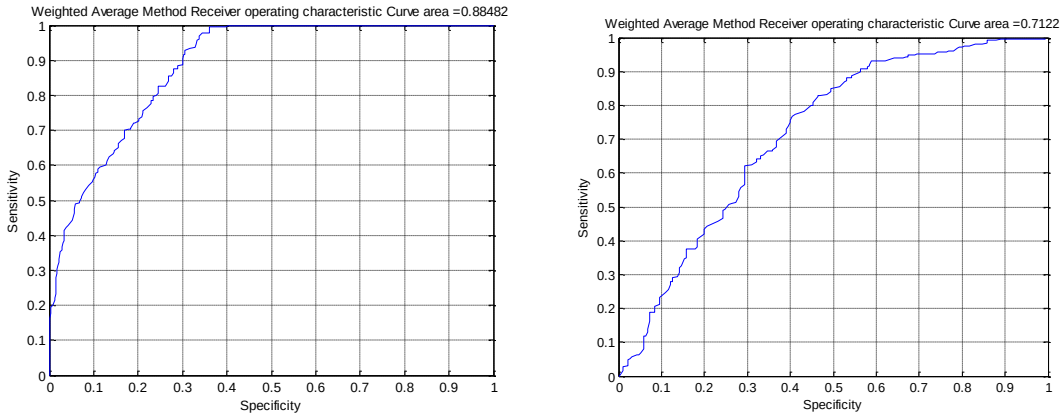


Figure 5.15: ROC training/testing of Ionosphere records

5.4.3 Optimized Weighted Average

In optimised weighted average or Pareto optimisation, multiple decision making occurs based on simultaneous mathematical optimisation. Since the learning in this method is multi-objective, much has to be done to control these objectives. Some of these parameters are the error function E, the complexity Ω and interpretability. The Pareto optimal solution occurs if and only if it is not dominated by another solution obtained, as there are other Pareto optimal solutions in a model. In this model, the measurement used is the MSE. In the optimised weighted average, the Pareto optimality is the most significant aspect in this model. The parameter of complexity Ω can be represented in two main forms: either it is the sum of squared weights ω_i or the sum of the absolute weights.

After testing the model and comparing it to scalar multi-objective learning, the most significant aspect of practicality is that while there is no need to identify the positive predetermined hyperparameter (λ), it is still very important to pick out one or more optimised Pareto solutions; this all depends on the user's preference and the problem in question. The GA algorithm optimiser which is based on the Matlab toolbox was set with the following parameters:

Population Size: 1000

Cross Over Rate: 0.95

Mutation rate: 0.15

Table 5.5 shows the performance results of the training experimental data sets of the optimised weighted average combination method compared to the actual data, while Table 5.6 shows the performance results of the testing set. The data in the testing set are there to predict the data sets' output. In the experimental results, data records have been divided into

80% for performance training and 20% for performance testing. This division has been chosen as it delivers the best results.

In each table represented below, the rows represent the 7 data sets which are: Breast Cancer Bladder Cancer, Statlog (Heart), Contraceptive, German Credit and Australian Credit. In addition, the columns represent the analysis parameters which are: Sensitivity, Specificity, Accuracy, AUC and RMS. Each cell contains a number which refers to a percentage (for the first three columns) while the other two columns represent measurements of the area under the curve and the root mean square.

Table 5.5: Experimental data training performance

	Sensitivity	Specificity	Accuracy	AUC	RMS
Breast Cancer	93.865	99.0476	97.2803	0.85605	0.1797
Bladder Cancer	72.7273	88.75	82.2222	0.9042	0.3655
Statlog (Heart)	92.9412	97.1154	95.2381	0.9664	0.2516
German Credit	58.6047	96.701	85	0.92678	0.3456
Australian Credit	93.1507	86.1905	89.7436	0.96931	0.2644
Ionosphere	100	90	96.3115	0.95993	0.2127
Contraceptive	99.6575	63.4703	84.1487	0.88259	0.4166

Table 5.6: Experimental data testing performance

	Sensitivity	Specificity	Accuracy	AUC	RMS
Breast Cancer	92	99.2248	96.5686	0.82522	0.1804
Bladder Cancer	60.8696	80	72.4138	0.71863	0.4184
Statlog (Heart)	94.2857	93.4783	93.8272	0.95248	0.2576
German Credit	38.8235	92.093	77	0.79283	0.3971
Australian Credit	94.3182	82.2917	88.0435	0.92543	0.3123
Ionosphere	100	91.4286	97.1429	0.97061	0.1843
Contraceptive	93.1727	39.4737	69.9317	0.72398	0.5485

Figures 5.16 to 5.22 show the ROC of the training experimental data sets and the ROC curve of the testing data set (predictors of output data records). Each of the 7 data sets mentioned above have both trained and tested figures and each figure represents a plot of Specificity vs. Sensitivity. According to the results, the best prediction would yield a point in the upper left corner or coordinate (0,1) of the ROC space, representing both 100% sensitivity (no false negatives) and 100% specificity (no false positives). In addition, the (0,1) point is considered a perfect classification. In the ROC plot, when the curve is nearly touching the left top corner coming closer to 1, that means that this ROC enjoys high accuracy and when it goes down that corresponds to low accuracy ROC. Therefore, the more higher and closer to the corner the curve is, the better results are obtained in terms of accuracy.

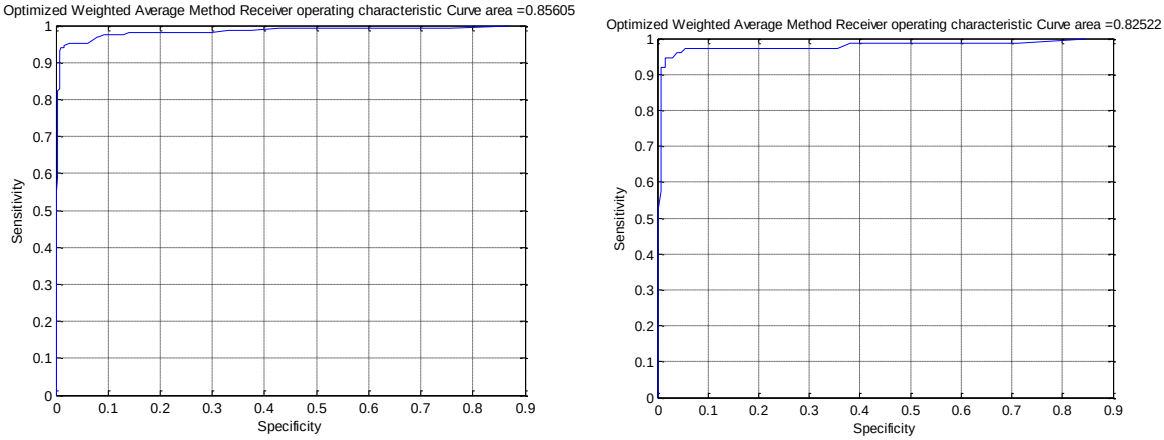


Figure 5.16: ROC training/testing of Breast cancer records

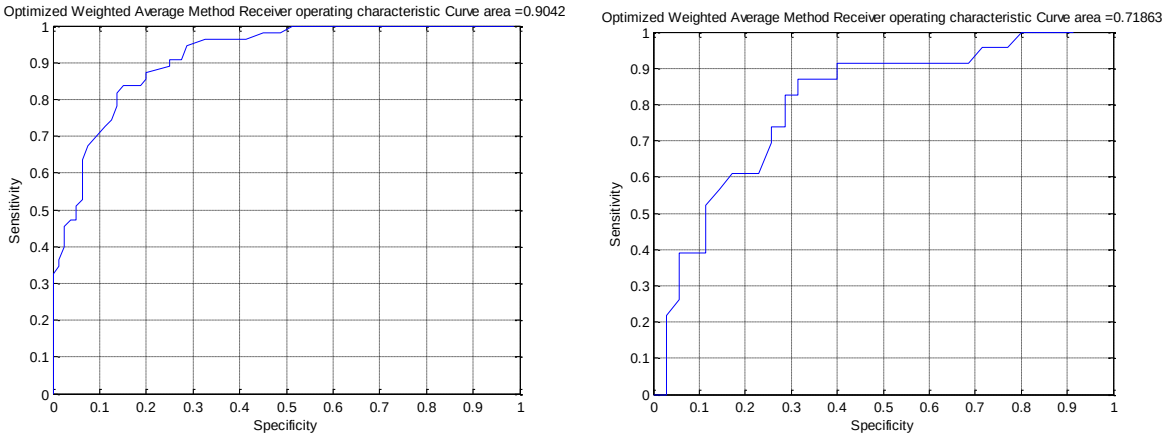


Figure 5.17: ROC training/testing of Bladder cancer records

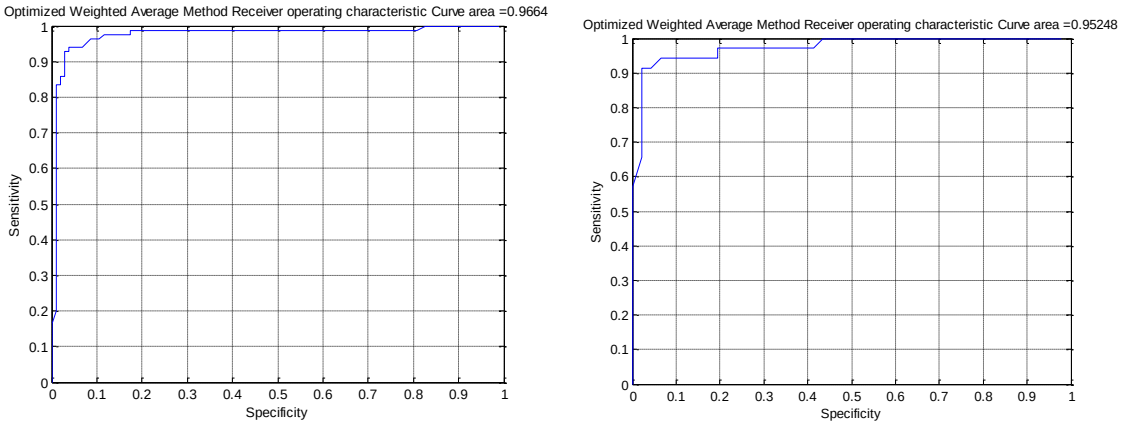


Figure 5.18: ROC training/testing of Statlog(Heart) records

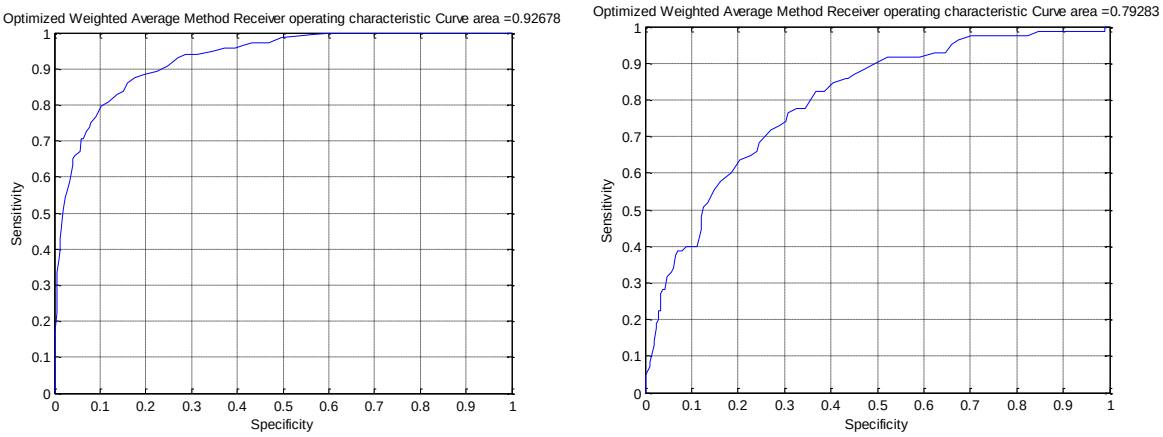


Figure 5.19: ROC training/testing of Contraceptive records

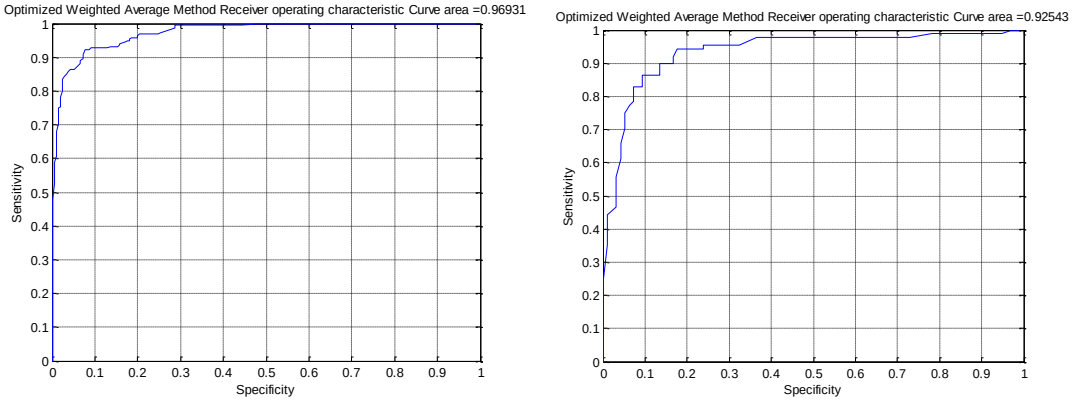


Figure 5.20: ROC training/testing of German credit records

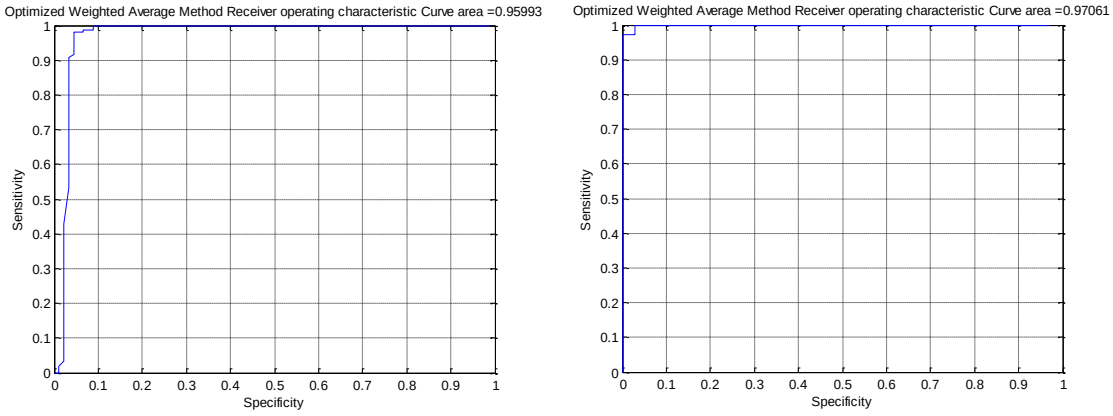


Figure 5.21: ROC training/testing of Australian credit records

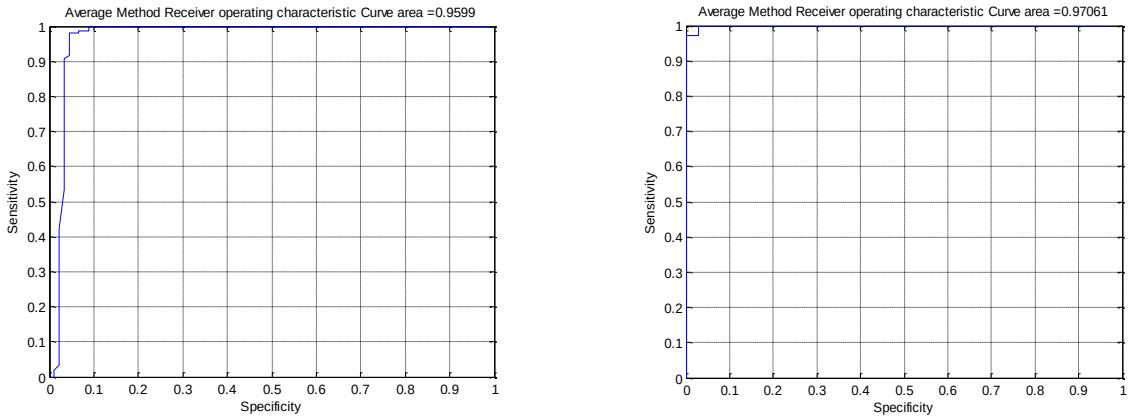


Figure 5.22: ROC training/testing of Ionosphere records

5.4.4 Voting

Voting methods can take two forms: either they are based on agreement-based voters or voters generating output without the need of agreement among the results. The most popular type is the second type in which the median and the weighted average are used in its applications. Voting-based methods function exclusively on labels in which $d_{t,j}$ can have a value of 1 or 0 according to the classifier's choice of j. Therefore, the largest total vote J is chosen by the ensemble. Accordingly, if the outputs of the classifier are independent, the majority voting will always enhance the overall performance of the model.

In the case of a two-class problem, the probability of making a good decision will take a binomial distribution nature. Therefore, according to the binomial distribution, the ensemble's probability is close to 1 when T (number of classifiers) approaches ∞ if the probability of making a good decision p is larger than 0.5. On the other hand, the ensemble's

probability is close to 0 when T approaches ∞ at p less than 0.5.With the increase in complexity in classification problems, the method of generating a neural network will decrease the complexity due to the tight coupling between the weights that lead to overlapping and crosstalk, which will result in partial- and under-learning. To solve this issue, the complex classifications are decomposed into simpler tasks and each one is solved separately. The overall performance depends on the integration of these module decisions in a general global one.

Table 5.7 shows the performance results of the training experimental data sets of the voting combination method compared to the actual data, while Table 5.8 shows the performance results of the testing set. The data in the testing set are there to predict the data sets' output. In the experimental results, the data records have been divided into 80% for performance training and 20% for performance testing. This division has been chosen as it delivers the best results.

In each table represented below, the rows represent the 7 data sets which are: Breast Cancer Bladder Cancer, Statlog (Heart), Contraceptive, German Credit and Australian Credit. In addition, the columns represent the analysis parameters which are: Sensitivity, Specificity, Accuracy, AUC and RMS. Each cell contains a number which refers to a percentage (for the first three columns) while the other two columns represent measurements of the area under the curve and the root mean square.

Table 5.7: Experimental data training performance

	Sensitivity	Specificity	Accuracy	AUC	RMS
Breast Cancer	93.865	99.0476	97.2803	0.8605	0.1783
Bladder Cancer	70.9091	90	82.2222	0.90045	0.3648
Statlog (Heart)	90.5882	96.1538	93.6508	0.96437	0.2565
German Credit	57.2093	96.4948	84.4286	0.92443	0.3460
Australian Credit	92.6941	85.7143	89.2774	0.9692	0.2662
Ionosphere	100	90	96.3115	0.96004	0.2102
Contraceptive	99.322	64.3836	84.4358	0.87863	0.4152

Table 5.8: Experimental data testing performance

	Sensitivity	Specificity	Accuracy	AUC	RMS
Breast Cancer	92	99.2248	96.5686	0.82811	0.1789
Bladder Cancer	56.5217	82.8571	72.4138	0.68323	0.4203
Statlog (Heart)	88.5714	93.4783	91.358	0.94907	0.2640
German Credit	38.8235	91.1628	76.3333	0.79554	0.3976
Australian Credit	94.3182	81.25	87.5	0.92507	0.3129
Ionosphere	100	91.4286	97.1429	0.97061	0.1828
Contraceptive	91.6	40.2116	69.4761	0.69451	0.5657

Figures 5.23 to 5.29 show the ROC of the training experimental data sets and the ROC curve of the testing data set (predictors of output data records). Each of the 7 data sets mentioned above have both trained and tested figures and each figure represents a plot of Specificity vs. Sensitivity.

According to the results, the best prediction would yield a point in the upper left corner or coordinate (0,1) of the ROC space, representing both 100% sensitivity (no false negatives) and 100% specificity (no false positives). In addition, the (0,1) point is considered a perfect classification. In the ROC plot, when the curve is nearly touching the left top corner coming closer to 1, that means that this ROC enjoys high accuracy and when it goes down that corresponds to low accuracy ROC. Therefore, the more higher and closer to the corner the curve is, the better results are obtained in terms of accuracy.

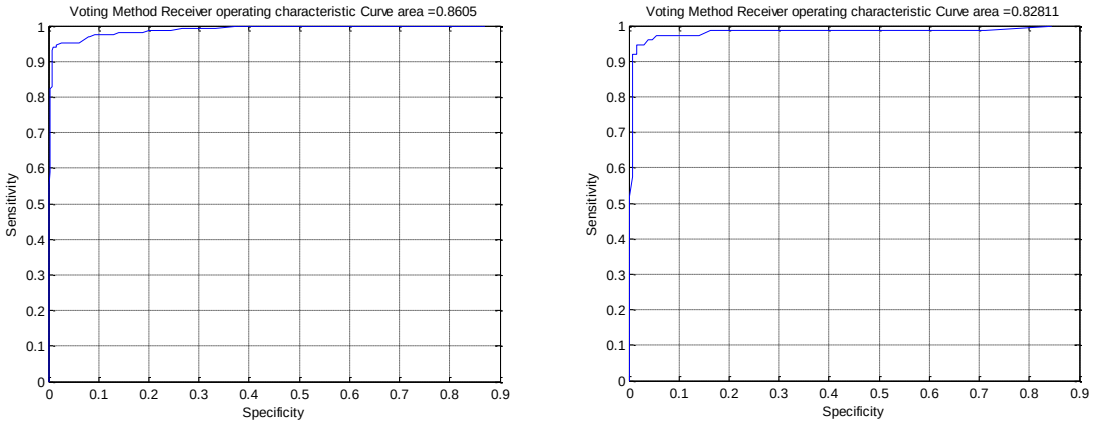


Figure 5.23: ROC training/testing of Breast Cancer records

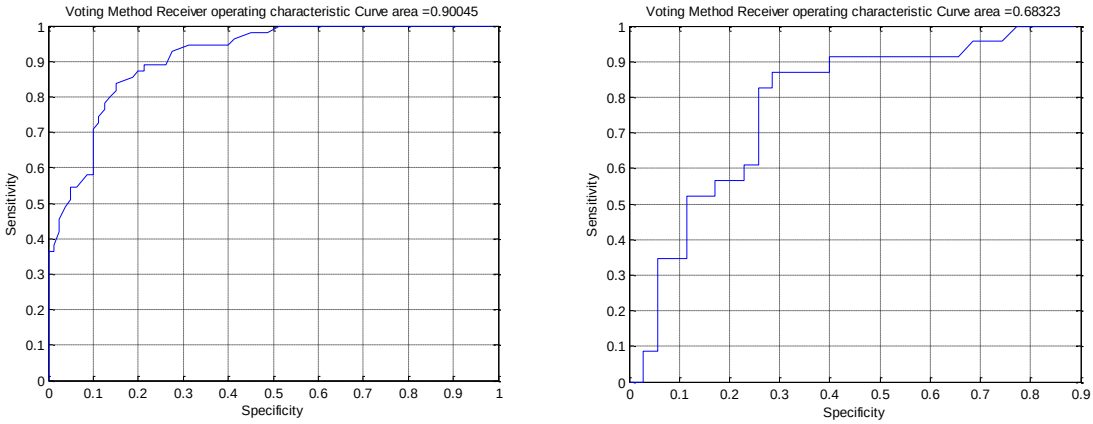


Figure 5.24: ROC training/testing of Bladder cancer records

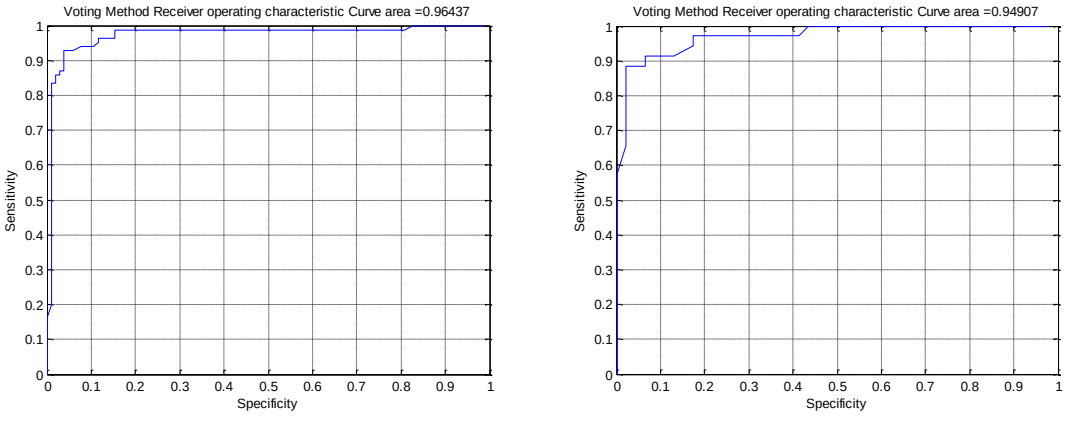


Figure 5.25: ROC training/testing of Statlog (Heart) records

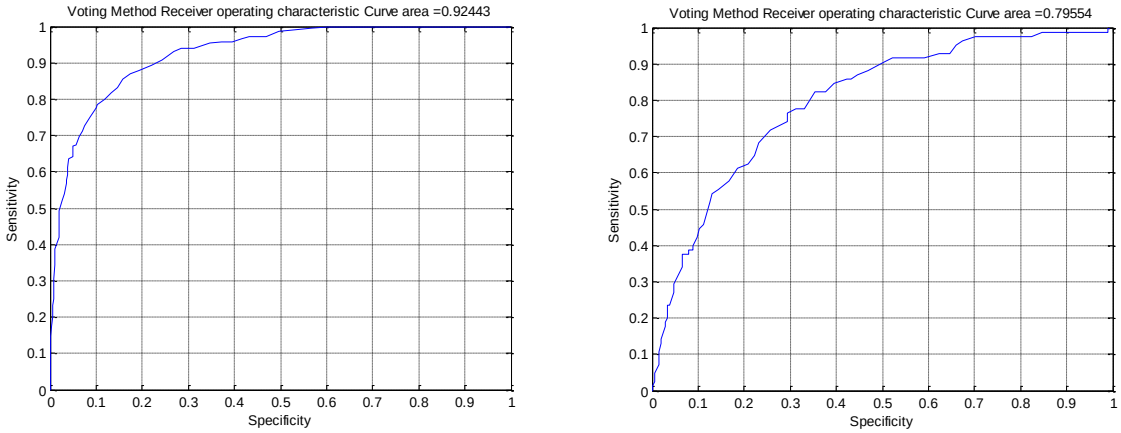


Figure 5.26: ROC training/testing of Contraceptive records

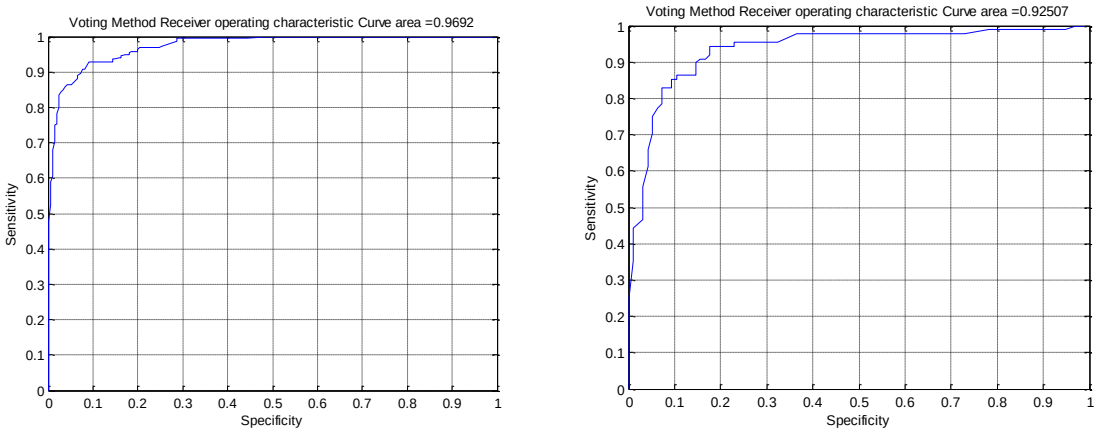


Figure 5.27: ROC training/testing of German credit records

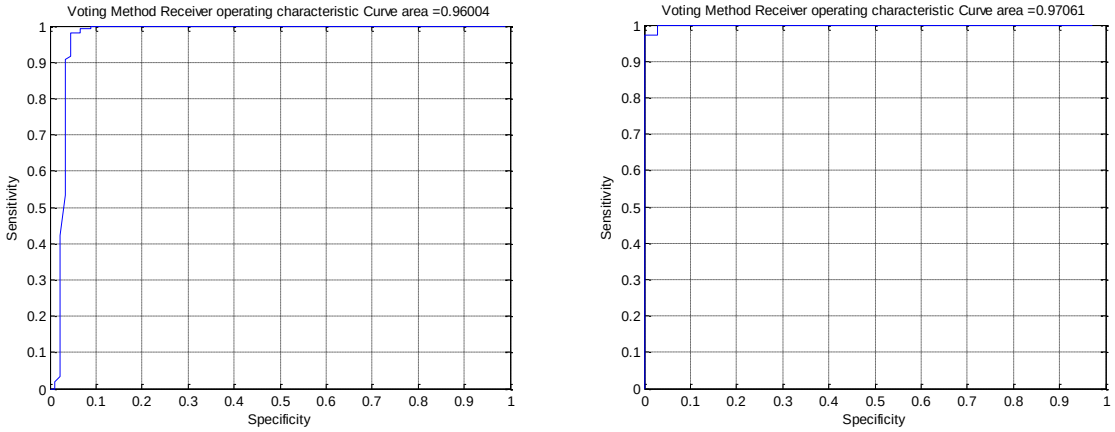


Figure 5.28: ROC training/testing of Australian credit records

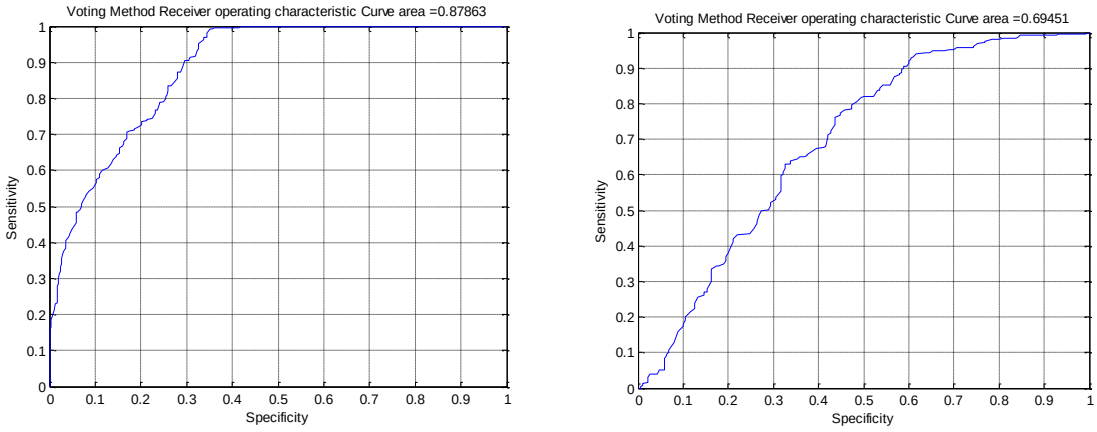


Figure 5.29: ROC training/testing of Ionosphere records

Four methods of combination methods have been devised which are average, weighted average, optimized weighted average and voting. Though different ratios of training to testing have been employed, 80% training to 20% voting is proved to give out the best results. After the ensemble method is done, the models are all collected using committee machine.

Committee machine is working side by side to solve a particular problem. Therefore, the outputs from each level are combined via the next layer so as to generate the output of that layer. As for the input, it consists of a set of subspaces between the input level perceptrons. Many different networks are trained, then the best trained data are kept while the rest is discarded depending on their performance. To overcome these issues, the networks are combined together to generate a committee machine. The main advantage of this method is that it can result in prominent performance enhancements for new data, with a bit of effort in computations. Basically, the committee's performance is proven to be better than the best single network (HAYKIN,1999) .

The results represented in tables 5.9 and 5.10 shows the comparison the results of different combination methods average, weighted average, optimized weighted average and voting,based on AUC output results for all datasets using all training dataset sizes. The table clearly shows that weighted average method leads to greater output for data sets in general.

Table 5.9: AUC Comparison of Four combination methods training performance

Training	Average	Weighted Average	Optimzed Weighted Average	Voting
Breast Cancer	0.85605	**0.8765**	0.85605	0.8605
Bladder Cancer	0.90443	**0.90818**	0.9042	0.90045
Statlog (Heart)	0.9664	**0.96697**	0.9664	0.96437
Contraceptive	0.92694	**0.92693**	0.92678	0.92443
German Credit	0.95993	**0.96941**	0.96931	0.9692
Australian Credit	0.9761	0.96011	0.95993	0.96004
Ionosphere	0.88259	**0.88482**	0.88259	0.87863

Table 5.10: AUC Comparison of Four combination methods testing performance

Training	Average	Weighted Average	Optimzed Weighted Average	Voting
Breast Cancer	0.82786	**0.84284**	0.82522	0.82811
Bladder Cancer	0.69006	**0.76708**	0.71863	0.68323
Statlog (Heart)	0.95248	**0.95466**	0.95248	0.94907
Contraceptive	0.79275	**0.79327**	0.79283	0.79554
German Credit	0.97061	0.92525	0.92543	0.92507
Australian Credit	0.97061	**0.97122**	0.97061	0.97061
Ionosphere	0.72402	**0.7122**	0.72398	0.69451

5.5 Summary

This research was done on one of the most popular topic of research in the field of machine learning which is classifier combination and combination methods. Combination methods are used in many applications in machine learning and pattern recognition and has shown some prominent performance compared to conventional methods. A brief comparison between fixed classifiers and ensemble of classifiers was discussed to further explain the advantage combination has over the fixed classification. Furthermore, the levels of operation and the types of output the combined classifiers operate is mentioned as well. Combination rules including algebraic combiners were mentioned so as to understand a general idea about the different mathematical approaches to how this method operates. Moving further in the research to discuss the main topic which are the different combination methods and the main points about each method in addition to the different equations which further assisted with understanding which method provides the best outcome in a particular application. The combination methods mentioned in this research were the averaging method in which both weighted average and optimized weighted average were discussed in details, in addition to voting methods of majority voting and weighted majority voting.

Chapter 6: Intelligent Combination Methods

6.1 Introduction

Combination methods have been shown to be very practical and effective techniques to improve the pattern recognition performance of various applications. The field of automating the combination of decisions has been studied and focused on by researchers since the 1950s. These studies have included various applications ranging from economics, forecasting natural disasters, medicine and technology. These combinations take either a mathematical or a behavioural approach. The mathematical approach depends upon logic and statistics to generate the models and derive the combination rules. However, the behavioural approach assumes discussions among experts in addition to direct human monitoring throughout the combination process. Generally, the mathematical approach has become more popular with the introduction of computer expert systems (Tulaykov and Jaeger, 2007).

Moreover, pattern classification has been implemented in assigning input signals to two or more classes. The combined experts are classifiers and the combination of these classifiers is also a classifier. The classifiers' outputs can be represented as vectors and the dimension of these vectors corresponds to the number of classes. Thus, the combination problem can be further explained as finding the combination function inputting N-dimension vectors from M number of classifiers and the output is N classification scores, as shown in Figure 6.1.

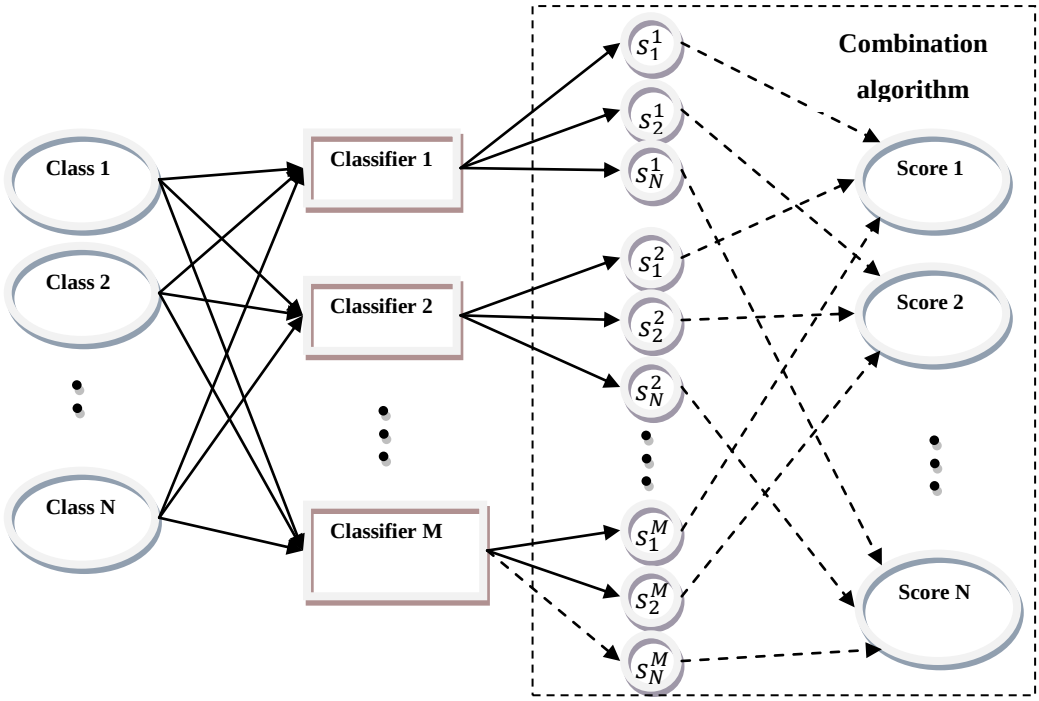


Figure 6.1: Combination method block diagram (Tulaykov and Jaeger, 2007)

Accordingly, the classifier combination employs a set of s_i^j score for class i by classifier j in order to create combination scores Si for each class i with initial condition. In addition, the function can be considered optimal since it successfully minimised the cost of misclassification (Tulaykov and Jaeger, 2007). With further research, more advanced combination methods have been introduced with the vast variety of current applications. Two of the most popular advanced combination methods will be discussed in this research in detail: genetic programming (GP) and the Coalition-based (CB) ensemble algorithm. Coalition formation is known as the technique is which groups are generated and certain problems are solved with the employment of cooperation.

6.2 Genetic Programming

One of the most challenging issues in computer science is getting a computer to do a certain task without the need to tell it how to do it. GP takes this issue in consideration by proposing a method for generating computer programs automatically. It achieves this goal by breeding a population of computer programs by employing the "Darwinism natural selection" principle among other biological-based principles. These methods include reproduction, crossover, mutation and architecture-modifying techniques structured after gene deletion and

duplication. In the field of artificial intelligence, genetic programming can be considered as an evolutionary algorithm type of method influenced mainly by biological evolution in order to obtain computer programs that perform user pre-defined tasks. Basically, a GP is a set of instructions and fitness functions both contributing to measuring how well a computer has performed a certain given task. It is a subfield of Genetic Algorithms (GA) in which each individual is a computer program. It is used to optimally fit a population of computer programs based on a fitness platform determined by the ability to execute a certain task (Koza, 2003).

Out of a number of potential programs which are often small functions embedded in larger applications, the most efficient programs are the ones that survive, compete and/or cross breed with other programs for the sake of approaching towards the solution needed. Genetic programming is a method that enjoys practicality with problems that have a huge number of alternating variables, the same ones that are in artificial intelligence. GP models are often programmed and implemented in software such as (LISP) "LIST Processor" a family of computer programming languages. in addition to Scheme programming languages and C language. However, the most challenging thing about genetic programming is evaluating the extent to which a program is helping to reach to the desired goal, or what is known as the "fitness function". The simplest example to illustrate generally how genetic programming works is by generating a program to fire a gun in which the distance the bullet would miss its target is determined by the "fitness function" (Rouse, 2005).

6.2.1 Preparatory Steps for Genetic Programming

The user links the high-level statement of the problem to the GP system by employing some preparatory steps. These steps of genetic programming oblige the user to identify the following:

- Set of terminals: some of these terminals include zero argument functions, independent variables and random constants. These are included for each branch of the potentially evolving program
- Primate functions set: for each branch of the potentially evolving programs
- Fitness parameter: to directly and indirectly measure the individuals' fitness in the population
- Few parameters needed to control the run

- The operation of assigning the result of the run in addition to the termination standard (Koza, 2003).

6.2.2 Executional Steps of Genetic Programming

Generally, GP begins its operation with a population of computer programs generated randomly consisting of the available programmatic elements. GP modifies a certain population iteratively into a new generation of programs population with the help of implementing natural occurrence genetic methods. These methods are applied to individuals chosen from the population. The individuals are chosen probabilistically to take part in the genetic methods according to their fitness (as discussed in the third preparatory step). The iterative population transformation is implemented inside what is known as "the main generational of the run of genetic programming" (Koza, 2003). Therefore, the executional steps of GP or in other words, the flowchart of Genetic programming is explained as follows:

1. Generating a random initial population called generation "0" of individual programs consisting of the available functions and terminals.

2. Performing the next sub-steps iteratively, known as a "generation" of the population until the termination standard is achieved. These generation sub-steps are:

a. Implementing each program and verifying its fitness through using the fitness measure
b. Choosing one or two programs from the population with a probability according to fitness for the sake of taking part in the genetic operations in the next step (reselection is allowed).
c. Producing new individual programs for the population via employing the following genetic operations:

   i. **Reproduction:** copying the chosen individual program to the new population
   ii. **Crossover:** generating a new offspring program for the population. This is done by recombining randomly selected parts from two chosen programs
   iii. **Mutation:** generating a new offspring program for the new population via mutating randomly selected part of one chosen program randomly
   iv. **Architecture-altering operations:** picking an architecture-altering operation from the available collection of related operations and thus generating one new offspring

program for the new population through applying the selected architecture-altering operation to one chosen program.

3. After the termination standard is achieved, the single best program in the population generated during the run is extracted and assigned as the result of the run. If the run was a success, the result might be a solution or an approximate to the problem. The steps mentioned are represented in the flowchart as shown in Figure 6.2.

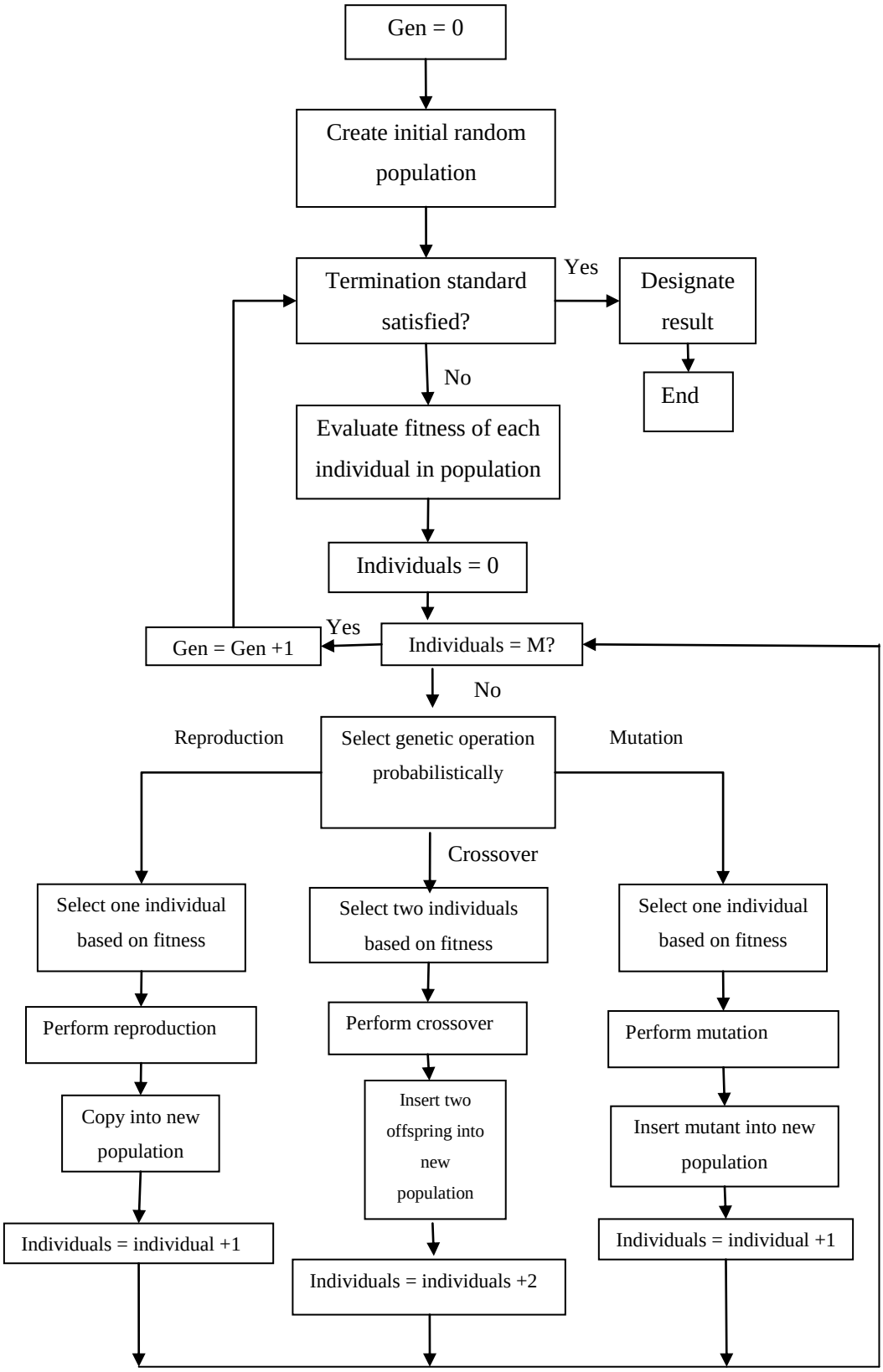


Figure 6.2: Genetic Programming flowchart (Koza, 1992)

6.2.3 Fitness Function

Fitness function is considered the most important principle in genetic programming. It determines the capability of a certain program to solve the problem. It is different from one type of program to another. If for example a genetic program was generated to set the time of a clock, then the fitness function would be the time by which the clock is wrong. Of course not all problems have such easy fitness functions. Mostly, they require a few alterations in the problem so as to find the fitness function (Koza, 1992).

Gun Firing Program This example is slightly more complex and is basically training a genetic program to fire a gun at a moving target. In this program, the fitness function would be the distance the bullet `is away from the target. However, the program has to consider a number of variables, for example, wind velocity, gun type, target distance, target height and velocity and acceleration of target. This shows that genetic programming can handle this kind of problem by assigning a fitness function with large number of variables (Koza, 1992).

Water Sprinkler System This program is concerned with controlling water-flow through a water sprinkler system. In this program, the fitness function would be how evenly the water is distributed over a surface. However, there is no variable for this measurement. Therefore, the problem must be modified to obtain a numerical fitness. One possibility is by installing water-collecting measuring equipment at some intervals on the surface. As a result, the fitness would be the standard deviation in water level obtained from all the measuring devices. Moreover, another fitness measure would be the difference between the ideal amount of water and the lowest measured level. Unfortunately, this value would not consider the watermarks at other measuring devices, which might not be the ideal mark (Koza, 1992).

Maze Solving Program In order to construct a program to finding the way out of a maze, the program has to be trained with several known mazes. The solution to the maze would be represented by a path of dots. In this program, the fitness would be the number of dots the program is able to locate. In addition, a time limit is included in the fitness function to prevent the program from wandering in the maze for a long time (Koza, 1992).

6.2.4 Functions and Terminals

The function and terminal sets are considered an integral part of the program that is going to be created. The terminal set includes both the constants and the variables of a program. For the maze program, Terminal set has basically three commands ,one is left, one is right and

one is forward. The function set represents the program's function. For example, the maze game will contain: "pseudo-code" (It has to be the same) would be something like: "if "dot" then do "X", else do "Y" ". While in the gun firing algorithm, the terminal set would be the set of different variables in the program such as the gun's velocity and acceleration, the target and the bullet while the function set includes some mathematical operations such as addition, multiplication, division, etc. (Koza, 1992).

6.2.5 Crossover Operation

Basically, there are two operations for altering structures in GP. The first and most significant operation is known as the crossover operation. In this method, two solutions are combined for the sake of presenting two new solutions. The parents are picked out of the population via a fitness function. There exists three methods to pick the crossover operation solutions. The first one includes employing probability according to the fitness function. Basically $f\left(s_i(t)\right)$ corresponds to the solution fitness S_i and the following function:

$$\sum_{j=1}^{M} f\left(s_j(t)\right) \qquad (6.1)$$

corresponds to the sum of the population individuals. Therefore, the probability weather S_i is copied to the next generation is:

$$\frac{f\left(s_i(t)\right)}{\sum_{j=1}^{M} f\left(s_j(t)\right)} \qquad (6.2)$$

The second technique for copying the solution to the next generation is known as tournament selection. Generally, the program randomly picks two solutions. The solution having the highest fitness will win. This method mimics biological mating systems where two individuals of the same gender create a competition in which the winner mates with a third party of different sex. The third method is rank selection. In this method, selection is made depending on the rank of the fitness values and not their numerical value (Koza, 1992).

6.3 Coalition-Based Ensemble Algorithm

To understand the coalition-based (CB) algorithm in machine learning, one must first understand the basic unit of a coalition which is an intelligent agent (IA). An intelligent agent is an independent unit which uses sensors to observe and react with the environment by using actuators and focusing its activity on fulfilling goals. IAs also learn or use knowledge to fulfil their goals. They might be very simple or very complex. A basic example of an intelligent agent is a thermostat (Russell and Norvig, 2003).

Intelligent agents are often represented as an abstract system quite similar to a computer program. Therefore, intelligent agents are sometimes known as abstract intelligent agents (AIA) in order to differentiate from their real-world applications such as computer systems, biological systems and even organisations. In computer science, intelligent agents can be used to define a software agent that enjoys intelligence even if it is not a rational agent (where intelligence means to have a goal-directed behaviour). The dependent programs that are used for both data mining and operation assistance are also referred to as intelligent agents. Intelligent agents need to have certain characteristics. First of all, they should be able to incrementally contain new problem solving techniques. They should also be able to evaluate themselves when it comes to behaviour, error and success. Learning and improvement should be done through interaction with the environment and from large amounts of data it should be done quickly. In addition, they should contain memory-dependent exemplar storage and retrieval capacities, and for short and long-term memory, they should have specialised parameters (Russell and Norvig, 2003).

6.3.1 Basic Architecture

A conventional intelligent agent system can be represented mathematically as an "agent function". The agent function plans every percept sequence to an action the agent can execute or to parameters that can affect the actions such as feedback, coefficient, function or constant. According to the equation:

$$f: P^* \rightarrow A \qquad (6.3)$$

where f represents an the agent function and it is an abstract principle decision making. This decision making includes utility of individual options, fuzzy logic, deduction over logic rules etc. (Russell and Norvig, 2003).

6.3.2 Coalition formation

Coalition formation is known as the method to generate a group and solve a particular problem through cooperation. Thanks to the development of networks, each computing device can interact via networks. These resources can be clustered together and be utilised in coalition formation. In a multi-individual environment, an agent might require to work together with other agents to fulfil a task. In other words, an agent needs to negotiate with other agents to collect the required resources. Recently, coalition formation has been implemented in several applications. Based on that, some researchers have enhanced the coalition formation process depending on its environment. In addition, other researchers have analysed the agent's features and thus made some adjustments to improve these features (Rahwan, 2007).

Coalition formation is a basic form of action that is based on creating coherent groups of unique, self-operating intelligent agents for the sake of fulfilling their goals efficiently either individually or collectively. Generating efficient coalitions has been a major research interest in the field of multi-agent systems in machine learning. Within this achievement lies the main issue of deciding upon which of the probably coalitions are used to achieve the desired tasks. This includes finding a value for every possible coalition, known as the "coalition value". This coalition value is a parameter of how beneficial a certain coalition would be if it were generated. Since there is an exponential relationship between the number of possible coalitions and the number of involved agents, it would be more beneficial to distribute the values over all the agents rather than a single agent to calculate all of the values. As a result, this will efficiently use up all available computational resources in the system which in turn eliminates all points and possibilities of failure. Another significant characteristic of coalition organisations is the idea that among each coalition, the agents tend to organise their activities in order to fulfil their goals. However, no coordination happens for agents belonging to different coalitions unless their goals are related. In addition, the organisational architecture in each coalition is mostly flat, however, in some cases there would be a coalition leader assigned as a representative for the whole group (Rahwan, 2007).

Based on that, coalition formations have gained popularity in current research and have proven to be a very useful tool in many multi-agent systems. Some examples include:

- e-commerce: where the customers form a group to buy products in quantities to take advantage of the discounts available

- e-business: where agents form a coalition in order to fulfil the market needs

- Distributed sensor networks: where a group of sensors operate cooperatively to track specific targets

- Distributed automotive routing: by sharing deliveries, transportation costs can be cut down when coalitions of delivery agents are formed

- Information Acquisition: several information servers can form coalitions to answer queries making it easier to acquire information

Based on the examples above, the process of coalition formation should include three main operations. These operations are:

- Coalition value calculation: calculating the value of each coalition formed

- Coalition structure generation: calculating the disjoint coalitions set that carry the maximum total value

- Payoff distribution: assigning rewards to each agent in the coalition depending on the actions performed by the coalition as a whole (Rahwan, 2007)

These operations will be mentioned in detail below.

6.3.3 Coalition Value Calculation

There have been several algorithms designed for coalition formation in order to determine which coalitions can be formed. In order to do that, they must calculate the "coalition value" which corresponds to the outcome assumed if that specific coalition were formed. After calculating all coalition values, the decision is made about forming the coalition. Calculating the coalition value depends on the nature of the problem and the complexity of the calculation varies accordingly from linear to exponential. In situations where the agent's rationality is limited due to complexity, the coalition value may provide the optimum outcome with limited computational resources. However, one of the main issues in value calculation is with the number of values calculated. As mentioned earlier, the number of values is exponentially related to the number of agents. To solve this issue, they must distribute these calculated values among the agents instead of depending on a single agent.

As a result, the calculation process can be formed faster in addition to sharing the computation process among the agents (Rahwan, 2007).

6.3.4 Coalition Structure Generation

Another issue that is present in the coalition formation operation is the coalition structure generation (CSG). CSG means that by providing coalition values, the set of agents can be divided into comprehensive and disjoint coalitions. These divisions are known as coalition structures. For example:

$$A = \{a_1, a_2, a_3\} \qquad (6.4)$$

is a set of agents, therefore, there are five possibilities for coalition structures:

$$\{\{\boldsymbol{a_1}\},\{\boldsymbol{a_2}\},\{\boldsymbol{a_3}\}\},\{\{\boldsymbol{a_1}\},\{\boldsymbol{a_2},\boldsymbol{a_3}\}\},\{\{\boldsymbol{a_2}\},\{\boldsymbol{a_1},\boldsymbol{a_3}\}\},\{\{\boldsymbol{a_3}\},\{\boldsymbol{a_1},\boldsymbol{a_2}\}\},\{\{\boldsymbol{a_1},\boldsymbol{a_2},\boldsymbol{a_3}\}\} \qquad (6.5)$$

It is usually presumed that every coalition performs equally well. In other words, the coalition value is not related to the actions of non-members. This principle is known as characteristic function games (CFGs). In such settings, the coalition value is provided by a characteristic function. However, not all settings are considered CFG. Sometimes, the coalition value depends on non-members' actions because of the positive and negative externalities. The general case where coalition values depend on the non-members' actions is known to be the normal form games (NFGs) and CFGs are considered as a strict subset of NFGs. On the other hand, many, though not all, real-world multi-agent issues are CFGs. This is due to the fact that in many real-world applications, a coalition's possible actions and payoffs are not so affected by the non-members' actions (Rahwan, 2007).

Based on this setting CFG, the problem of coalition structure generation becomes a complete set partitioning problem. Specifically, by assuming a group of subsets of a ground set, and assigning a weight for each subset, the set partitioning problem becomes all about locating an optimal method to partition the ground set. This method is similar to CFG since in both cases it is necessary to find an optimal method of partitioning the set of agents by providing a number of coalitions and assigning a value to each coalition. In addition, in both cases every possible coalition should be taken into consideration. As a result, any algorithm designed to solve one of these problems is capable of solving the other. Moreover, the CFG problem is also similar to another problem in the aspect of combinatorial auctions, specifically in determining the winner in an auction. In this system, a number of subjects are being

auctioned at the same time and a number of bidders place bids on combinations of these subjects, a reason why this method was called by its name (combinatorial auctions). When the auction is closed, the auctioneer is required to partition the set of assets, thus providing the bids places (weights) on every combination (subset) of these subjects in a way that the resultant sum of bids, which is the revenue, is thus maximised. How this system is similar to the CFG problem is that the bids are allowed on every assets combination possible. (Rahwan, 2007)

6.3.5 Payoff Distribution

When determining which coalition is to be formed, it is necessary to determine the rewards that should be assigned to each agent in order to stay in a coalition to an extent that the coalition is considered to be stable. In this situation, stability indicates the state where agents have no motivation to deviate from their assigned coalitions. This is efficient because it verifies that the agents will use their resources exclusively on their assigned coalitions instead of moving to other coalitions. This in turn assures that coalitions can last long enough to fulfil their goals.

The analysis of the incentives has been studied along with cooperative game theory and many solutions have been introduced based on various stability concepts. In addition, transfer systems have been introduced to transfer non-stable payoff distributions to stable distributions while keeping the coalition structure intact. However, in the situation of cooperative environments, the agents' goal is to maximise the outcome of the system and as a result, maximise the social welfare despite their coalition values. Generally, the payoff distribution is relatively less significant and the main thing to focus on is coalition structure generation in order to maximise social welfare (Rahwan, 2007).

In all systems, groups of agents operate within a team to carry out certain tasks. In game theory, this is known as coalition formation. The output of this method is either known as the "grand coalition" which is the set containing all the agents, or a "coalition structure" which contains the disjoint coalitions (a part of the agents' set). In a transferable utility, there are no limitations on how agents can distribute the total payoffs among each other. Particularly, agents from a particular coalition can make a payment to agents that are not in the coalition. However, there are many cases in which this is not practical and this is due to two particular reasons. The first one is that in many cases it can only be possible to provide the best output if agents can belong to more than a single coalition at the same time. Moreover, agents

always need to distribute their resources among the coalitions which they enrol. The overlapping coalition formation is a model that is implemented in environments where agents need to assign different shares of their resources to provide different tasks at different members in different coalitions" (Chalkiadakis, 2008).

There are many subjects in the field of multi-agent systems, and coalition formation is an example in that field. The main issues are raised in questions such as: how to decompose a certain task? How to construct a coalition? How to find suitable agents for a coalition? etc. One method of answering these inquiries is by using an "organisation-based distributed system". This system consists of a hierarchy where each "parent" has a set of "children". Parents divide a task into subtasks and assign these subtasks to their children. After that, each child forms a coalition to fulfil the task given by the parents. The children also use reinforcement learning to optimise the agent's local decision. Grid computing is a famous topic related to coalition formation. The principle behind it is quite simple, it can be thought of as a large virtual computing system that allows the user to enjoy many services. This system comprises a large number of computers all linked by network connections. In addition, another popular system is known as ubiquitous computing. It is a somewhat mobile computing system embedded in an intelligent platform. Its basic principle is the fact that it performs computing anywhere, anytime and in many devices, in other words, as if all mobile devices were to contribute or sell their computer capability at any time in any place. As a result, both the recipient and the provider can benefit from this. For grid computing, long-term coalitions are needed to execute long-term tasks while for a mobile network (ubiquitous computing), stable coalitions are required to achieve tasks one at a time (Lee and Chen, 2006).

6.3 Experimental Results

In this section, there are two combination methods in this research and each of these methods will be applied on 7 data sets. Based on that, the method(s) that generate the best results will be concluded and the results will be presented. The results will be represented in three different formats: tables, graphs and trees for genetic programming. The rows in the tables represent the 7 data sets which are: Breast Cancer, Bladder Cancer, Statlog (Heart), Contraceptive, German Credit, Australian Credit and Ionosphere. In addition, the columns represent 5 analysis parameters which are: Sensitivity, Specificity, Accuracy, AUC and RMS.

Each cell will contain a number which refers to a percentage of accuracy (for the first three columns) while the other two columns represent measurements of the area under the curve and the root mean square. The performance of each method will be based 80% on network training and 20% on network testing. Furthermore, for training and testing, different percentages for each proportion was made. For example, the ratio of 90% of training to 10% testing was made in addition to 70% training to 30% testing. However, the best performance was obtained from having 80% for training and 20% for testing, therefore we have utilized the 80 to 20 percentage as a default method throughout the research. The method representing higher accuracy in addition to higher AUC and RMS will be considered as the one representing the best result.

The GP algorithm was set with the following parameters:
Population Size: 1000
Cross Over Rate: 0.95
Mutation rate: 0.15
Genetic programming methods having a set of instructions and fitness functions with initial conditions is known to present good results when compared to other classifiers, except for coalition method. 7 Data sets have been modeled based on GP. Three ANN models have been employed in addition to different fitness functions used in data sets having initial conditions so as to reach to the questions. Moreover, some of the fitness functions require more time to find the solution. GP combines the results by using best three classifiers

Figures 6.3 to 6.9 show genetic programming of the experimental data, the generated program containing the basic functions (+,-,*,/), and the tree representing the equation solution of each data set.

$$(((X + Z) * (2 * Z) + (24 + Z)) + (-1)) \,/\, ((Y + 3) + (16))$$

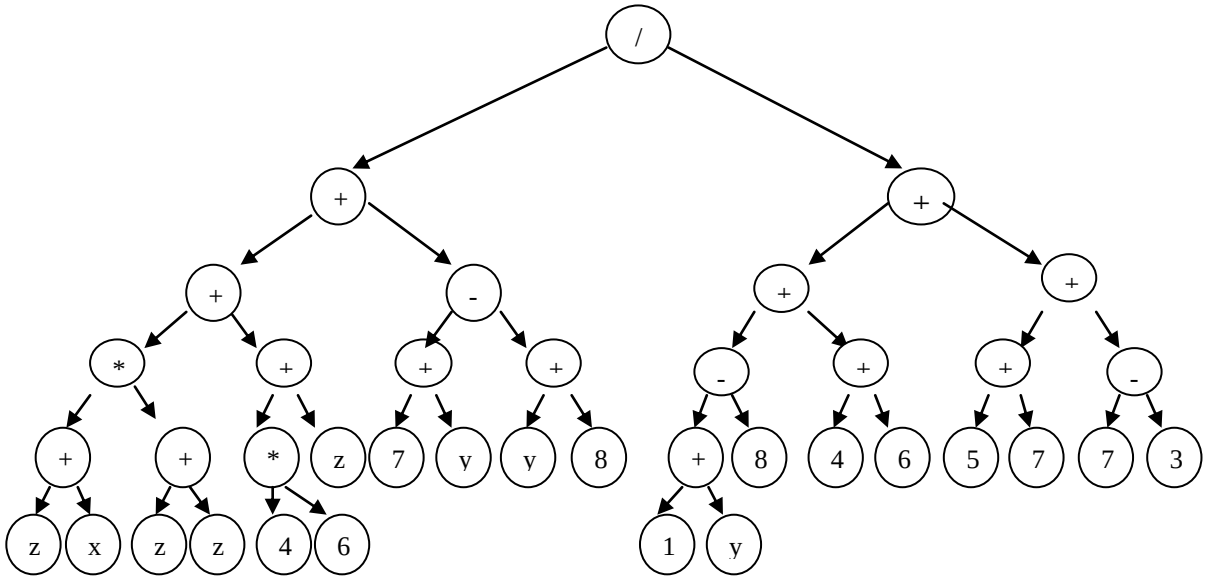


Figure 6.3: Genetic programming Breast cancer

$$[((Z + X - Y) + (ZY + 0.25Y)) * 17] \,/\, (4X + 24)$$

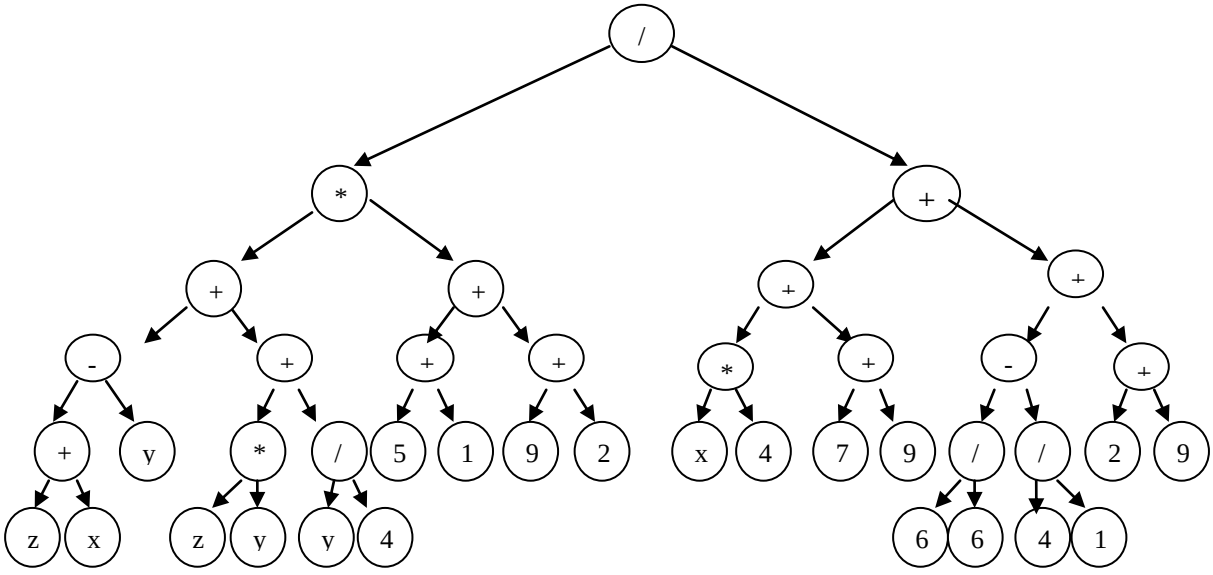


Figure 6.4: Genetic programming Bladder cancer

$$(((12 + Y) * 6 * X) \ * \ ((1 + Y)/(8 + X * Y))) \ / \ ((8 + Z - Y) \ + \ (10 + Y))$$

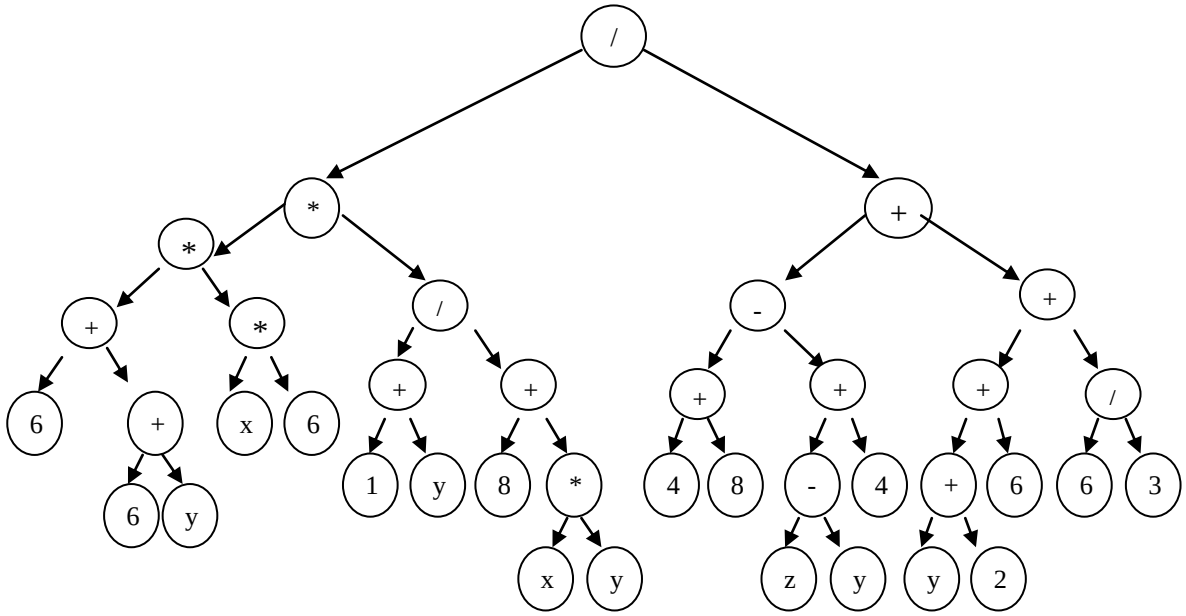


Figure 6.5: Genetic programming Statlog (Heart)

$$(17 + 2Y)/(Y - 30) \ + \ ((11 + X + Z)/(11/Y))$$

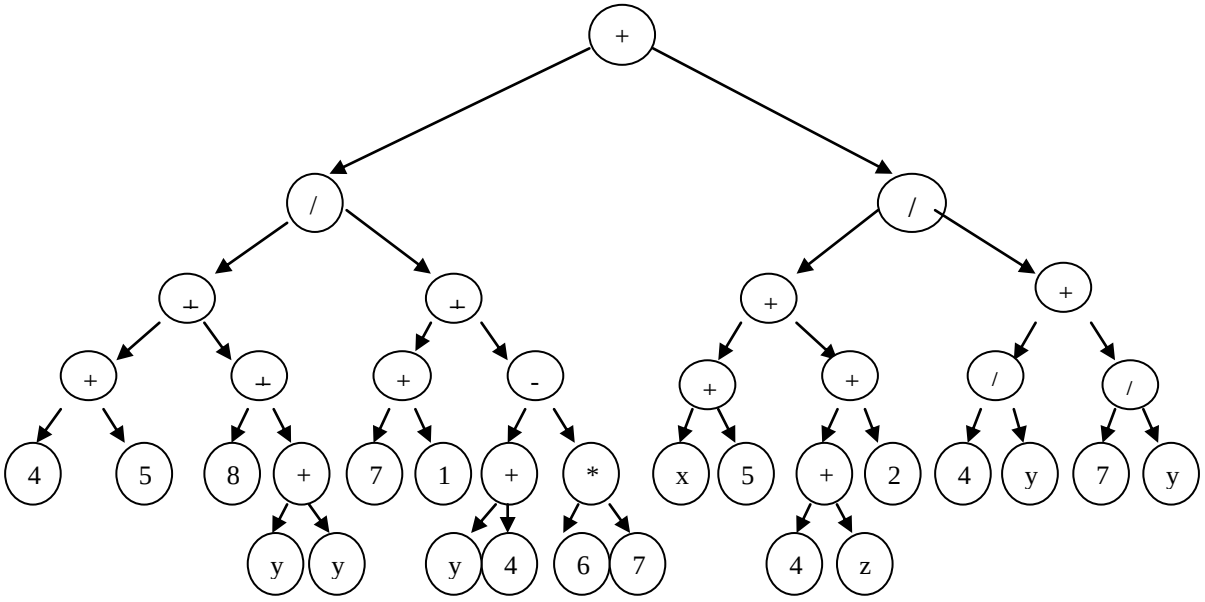


Figure 6.6: Genetic programming contraceptive

$$((5/7) + ((X+Y) * ((9*Z)+1+Z))) / (27+Y)$$

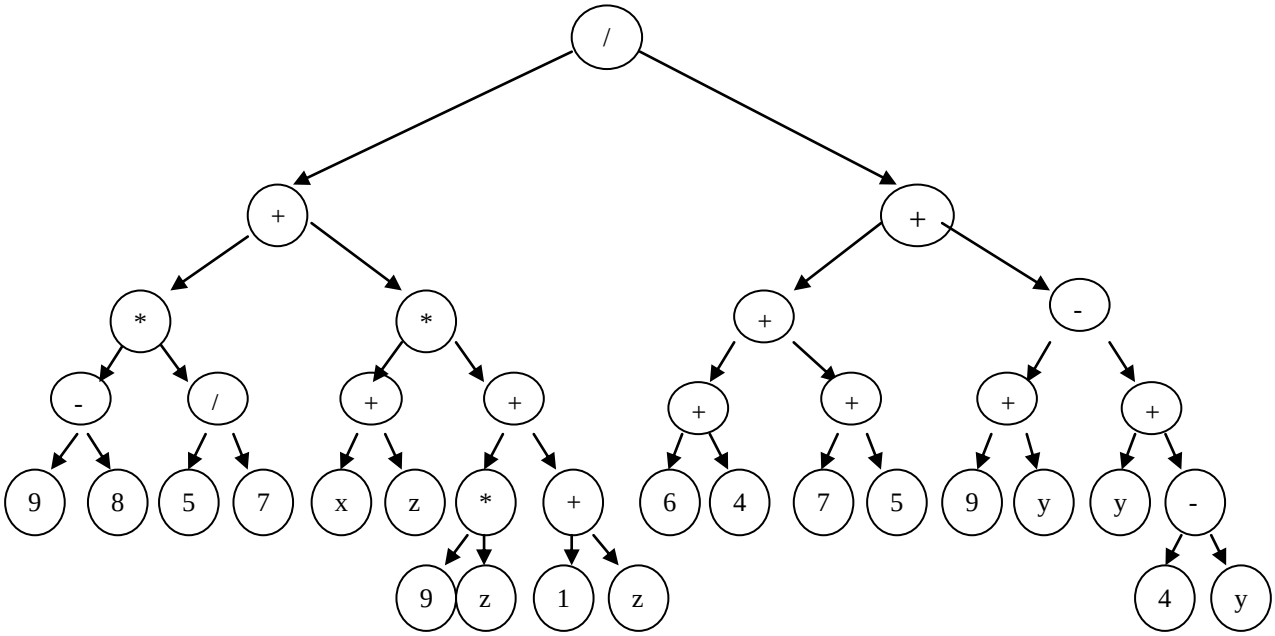


Figure 6.7: Genetic programming German credit

$$[(9+Z+X)+(Y-Z)*9+5+X] / [(((X+Y)/(Y/3+5)))+(5+(8/Z)/X)]$$

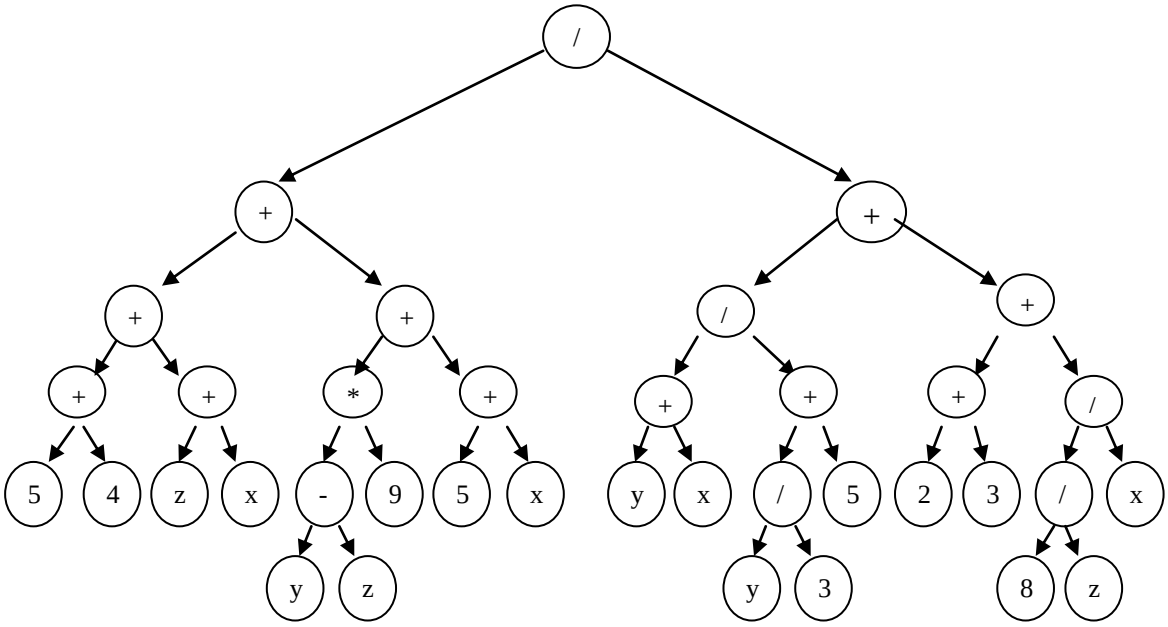


Figure 6.8: Genetic programming Australian credit

$$((16 + Z) + (16 * X/5)) \,/\, ((9 * Z + 7) + (-2 + (7/(X * Y))))$$

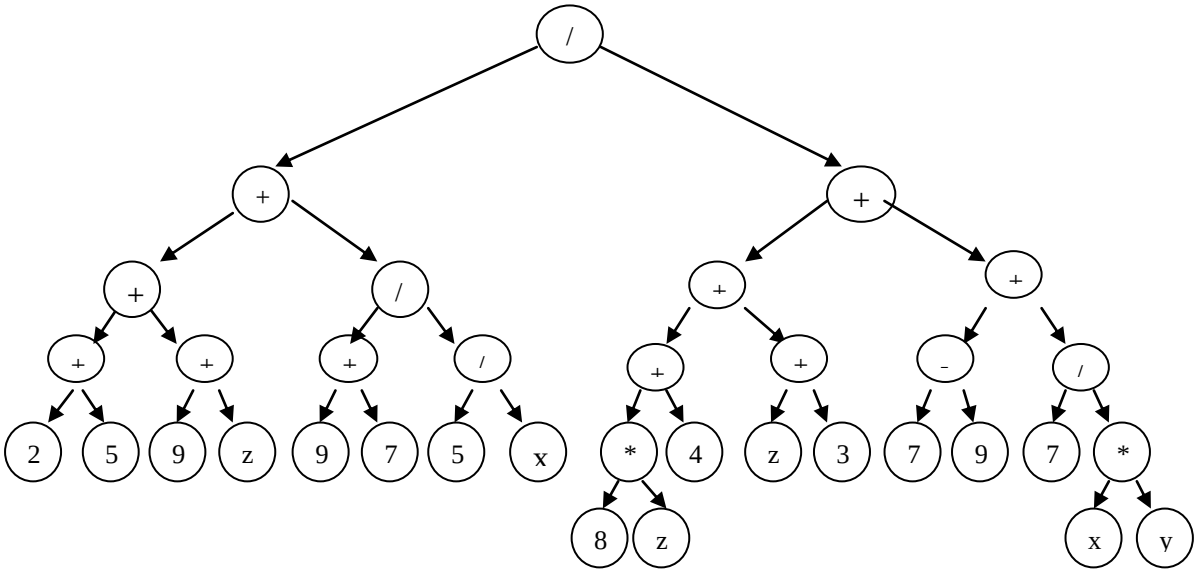


Figure 6.9: Genetic programming Ionosphere

Tables 6.1 and 6.2 show the results of the network training and the network testing performance of genetic programming, respectively.

Table 6.1: Experimental data GP training performance

	Sensitivity	**Specificity**	**Accuracy**	**AUC**	**RMS**
Breast Cancer	91.411	99.3651	96.6527	0.90499	0.0690
Bladder Cancer	80	87.5	84.4444	0.89182	0.121
Statlog (Heart)	94.1176	96.1538	95.2381	0.95831	0.057
Contraceptive	96.3964	64.3021	82.2581	0.89244	0.264
German Credit	74.8837	91.9588	86.7143	0.92667	0.105
Australian Credit	94.0639	96.1905	95.1049	0.9824	0.074
Ionosphere	100	90.1099	96.3265	0.96664	0.0452

Table 6.2: Experimental data GP testing performance

	Sensitivity	Specificity	Accuracy	AUC	RMS
Breast Cancer	94.6667	99.2248	97.549	0.98457	0.0615
Bladder Cancer	69.5652	77.1429	74.1379	0.77391	0.115
Statlog (Heart)	94.2857	95.6522	95.0617	0.95342	0.043
Contraceptive	84.7107	38.2979	64.4186	0.66602	0.288
German Credit	54.7619	84.6512	76.2542	0.78728	0.106
Australian Credit	89.5349	85.4167	87.3626	0.91388	0.070
Ionosphere	100	91.4286	97.1429	0.99531	0.0340

Figures 6.10 to 6.16 represent a clear visualisation for the values presented in Tables 6.1 and 6.2. Each of the 7 mentioned data sets have both trained and tested figures. These figures contain plots of representing Specificity vs. Sensitivity. According to the results, the best prediction would yield a point in the upper left corner or coordinate (0,1) of the ROC space, representing both 100% sensitivity (no false negatives) and 100% specificity (no false positives). In addition, the (0,1) point is considered a perfect classification. In the ROC plot, when the curve is nearly touching the left top corner coming closer to 1, that means that this ROC enjoys high accuracy and when it goes down that corresponds to low accuracy ROC. Therefore, the more higher and closer to the corner the curve is, the better results are obtained in terms of accuracy.

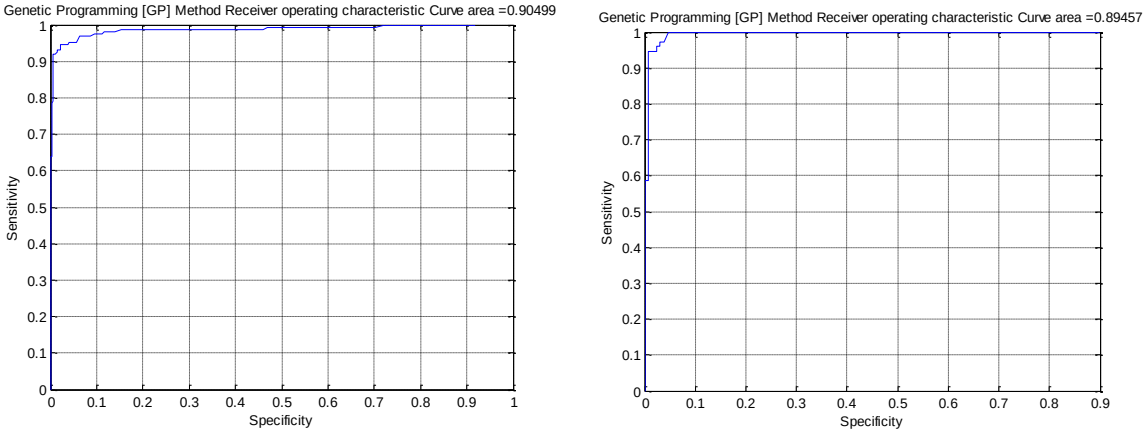


Figure 6.10: ROC training/testing of Breast cancer records

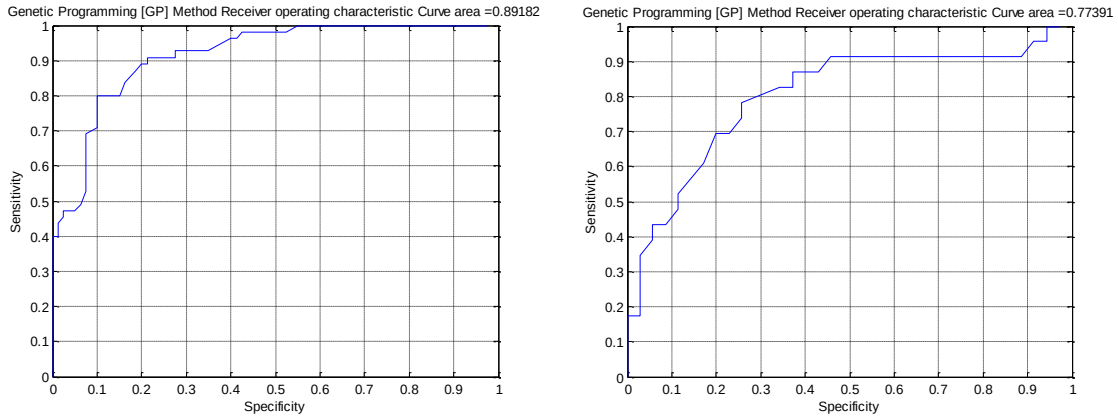


Figure 6.11: ROC training/testing of Bladder cancer records

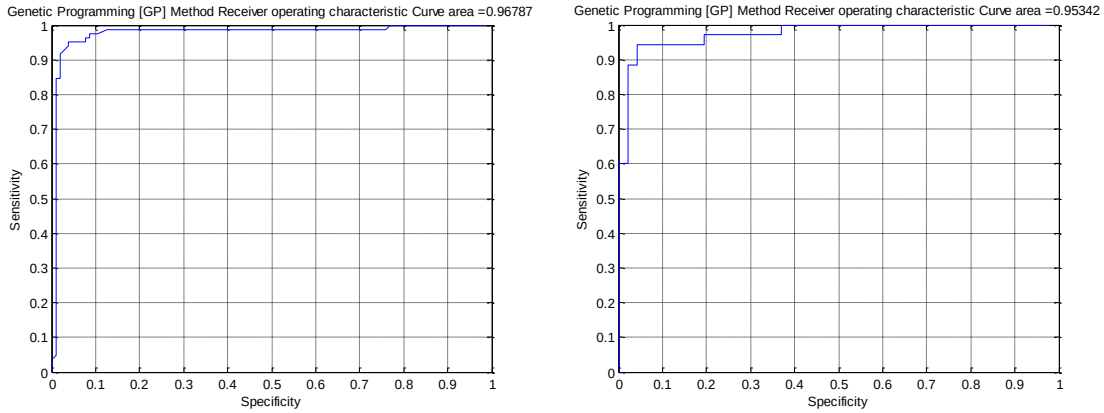


Figure 6.12: ROC training/testing of Statlog (Heart) records

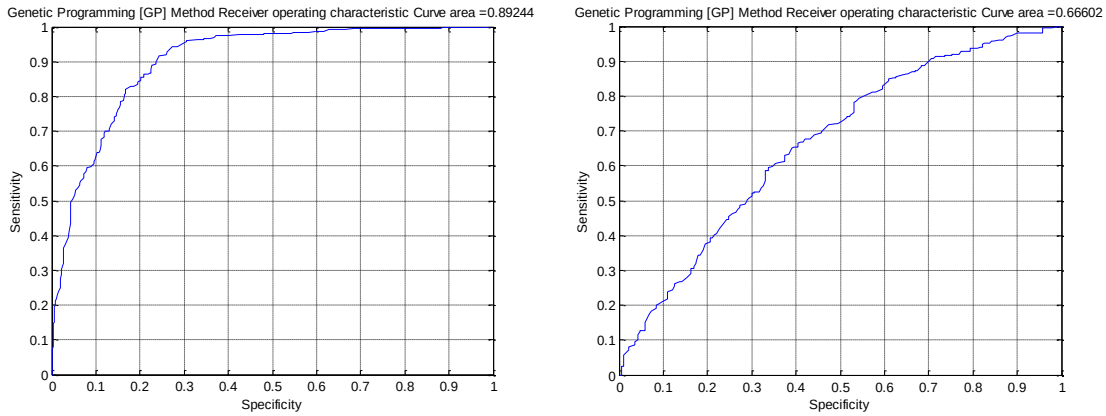


Figure 6.13: ROC training/testing of contraceptive records

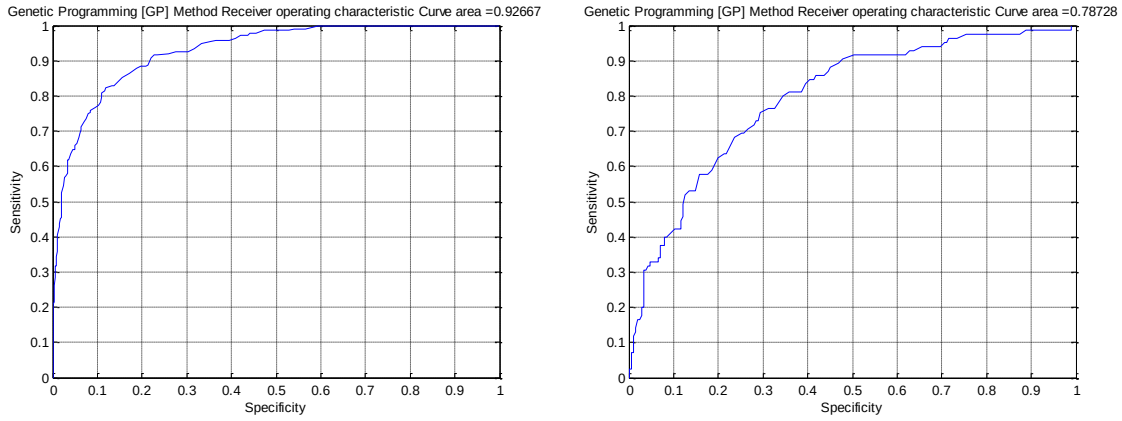


Figure 6.14 : ROC training/testing of German credit records

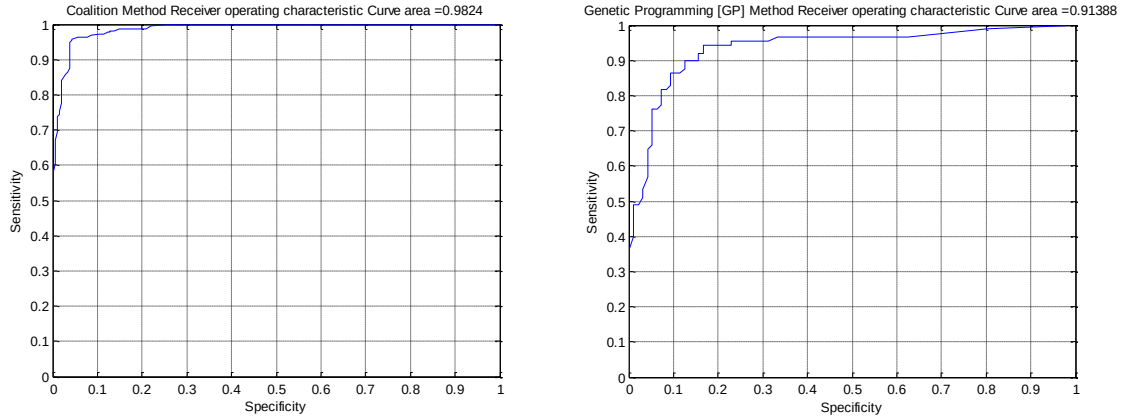


Figure 6.15: ROC training/testing of Australian credit records

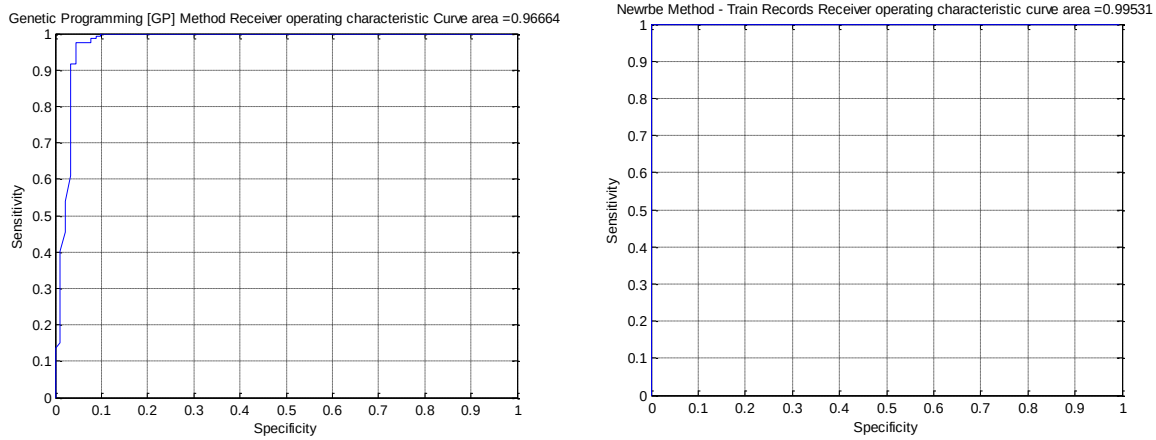


Figure 6.16: ROC training/testing of Ionosphere records

Tables 6.3 and 6.4 show the results of the network training and the network testing performance of the Coalition-based ensemble algorithm, respectively.

Table 6.3: Experimental data CB training performance

	Sensitivity	**Specificity**	**Accuracy**	**AUC**	**RMS**
Breast Cancer	93.7888	99.3691	97.4895	0.99042	0.0399
Bladder Cancer	89.0909	96.25	93.3333	0.97182	0.075
Statlog (Heart)	98.2353	98.913	94.5763	0.95831	0.0218
Contraceptive	99.4932	65.2968	84.9515	0.89598	0.204
German Credit	74.4186	98.9691	91.4286	0.95345	0.076
Australian Credit	92.2018	91.9048	92.0561	0.96051	0.052
Ionosphere	100	90.7216	96.3115	0.96501	0.0398

Table 6.4: Experimental data CB testing performance

	Sensitivity	**Specificity**	**Accuracy**	**AUC**	**RMS**
Breast Cancer	94.6667	100	98.0392	0.97853	0.0156
Bladder Cancer	95.6522	100	98.2759	0.95093	0.072
Statlog (Heart)	100	97.9167	98.7654	0.99057	0.0201
Contraceptive	98.3936	57.3684	80.6378	0.80665	0.226
German Credit	85.8824	99.0698	95.3333	0.95705	0.076
Australian Credit	94.3182	92.7083	93.4783	0.95798	0.046
Ionosphere	100	94.2857	98.0952	0.97122	0.0234

Figures 6.17 to 6.23 represent a clear visualisation for the values presented in Tables 6.3 and 6.4 above. Each of the 7 mentioned data sets have both trained and tested figures. These figures contain two scatter plots which are the actual predicted output and the rounded predicted output, and a line graph representing Specificity vs. Sensitivity. According to the results, the best prediction would yield a point in the upper left corner or coordinate (0,1) of

the ROC space, representing both 100% sensitivity (no false negatives) and 100% specificity (no false positives). In addition, the (0,1) point is considered a perfect classification. In the ROC plot, when the curve is nearly touching the left top corner coming closer to 1, that means that this ROC enjoys high accuracy and when it goes down that corresponds to low accuracy ROC. Therefore, the more higher and closer to the corner the curve is, the better results are obtained in terms of accuracy.

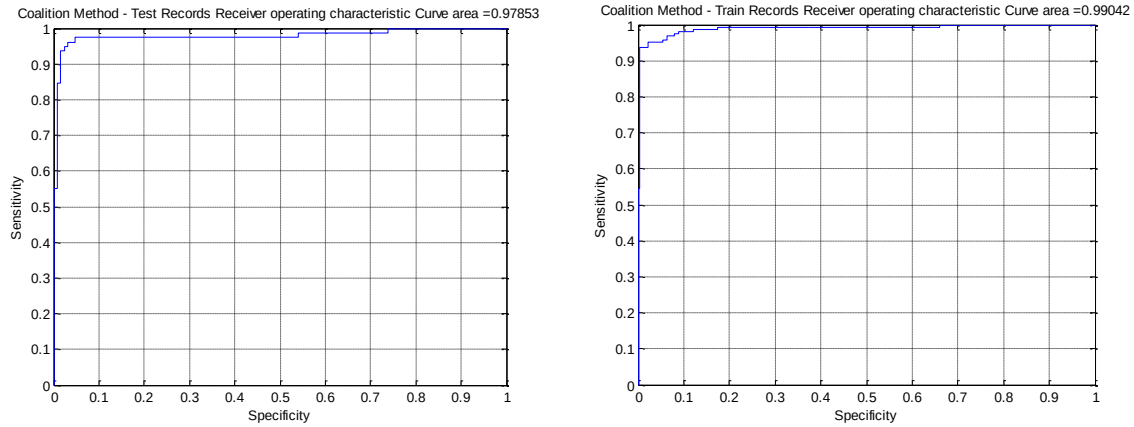


Figure 6.17: ROC training/testing of Breast cancer records

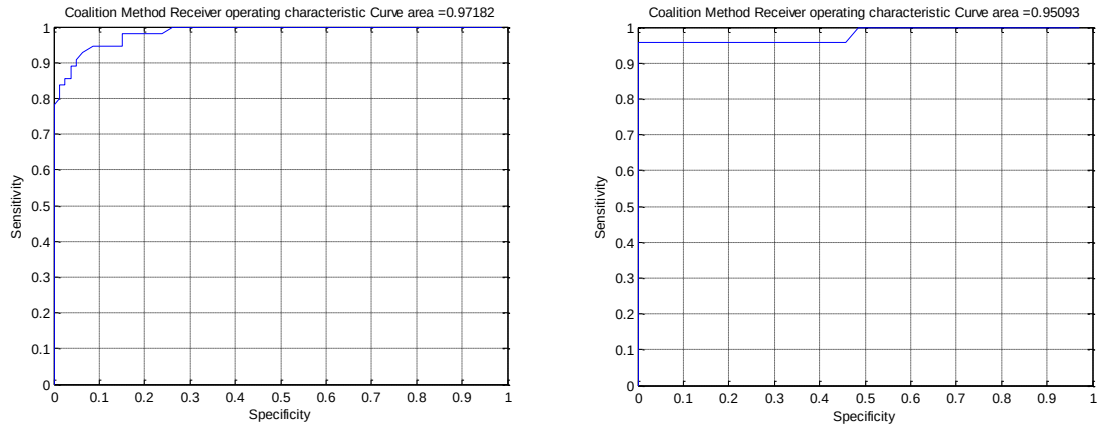


Figure 6.18: ROC training/testing of Bladder cancer records

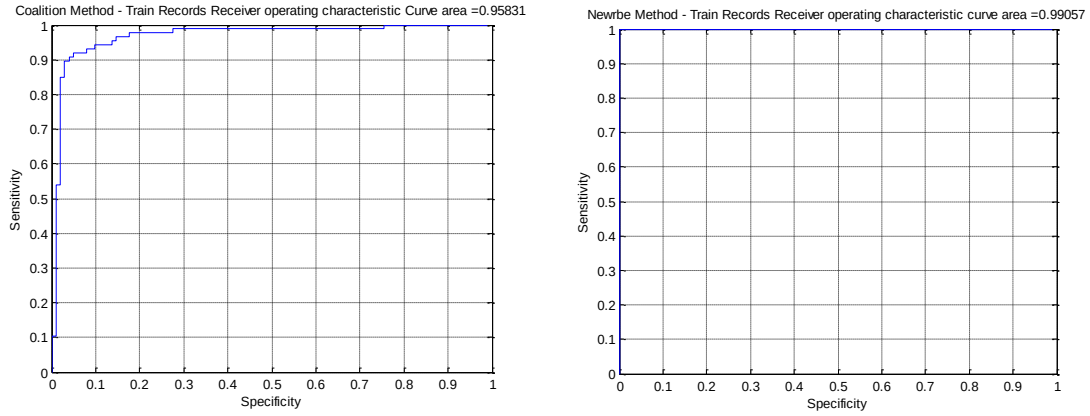


Figure 6.19: ROC training/testing of Statlog (Heart) records

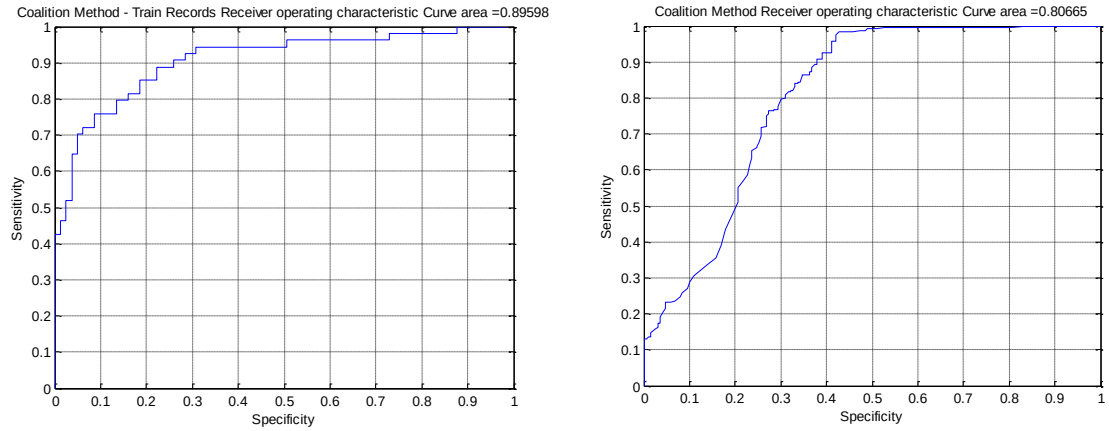


Figure 6.20: ROC training/testing of contraceptive records

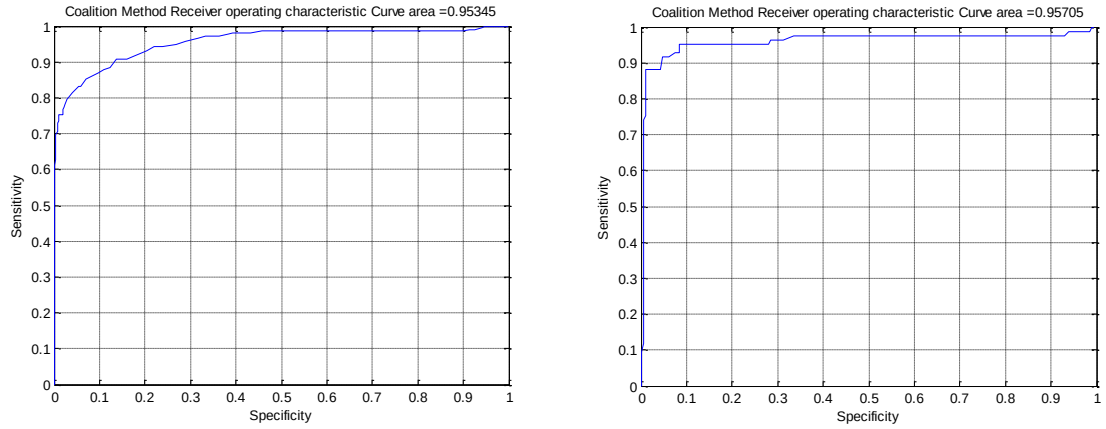


Figure 6.21: ROC training/testing of German credit records

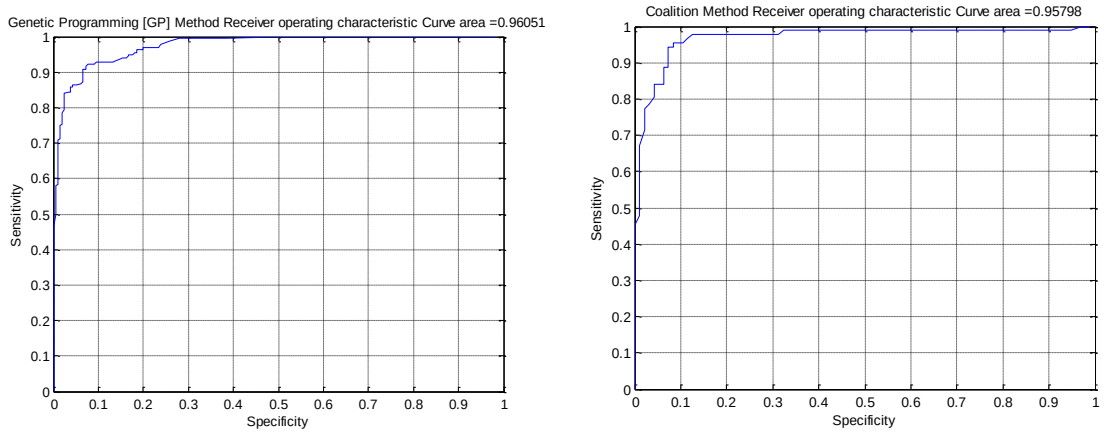


Figure 6.22: ROC training/testing of Australian credit records

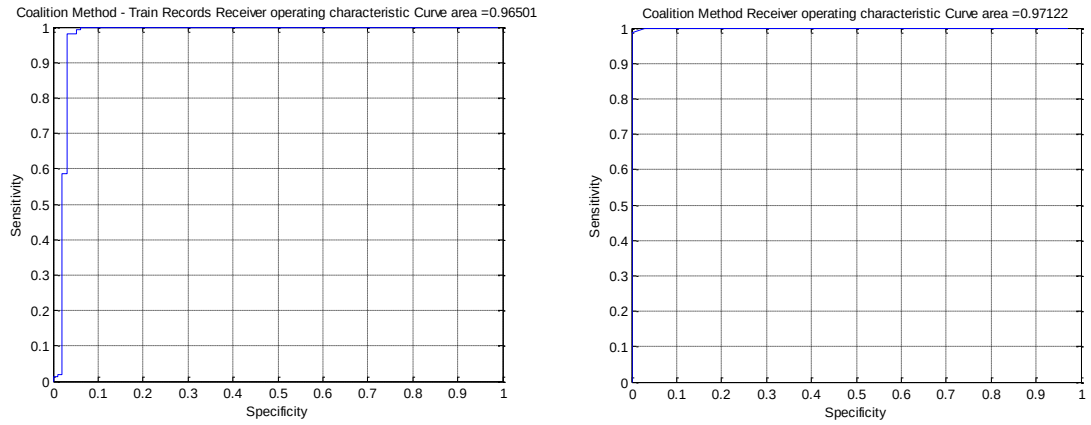


Figure 6.23: ROC training/testing of Ionosphere records

So far, each combination method was presented with training performance and testing performance. This has been done in order to measure the overall performance of the method by applying 7 data sets which are Breast cancer, Statlog (Heart), Contraceptive, German Credit, Australian Credit and Ionosphere. Each data set was analysed using 5 analysis parameters: Sensitivity, Specificity, Accuracy, AUC and RMS. To further understand the behaviour of the values presented in the tables, scatter plots and ling graphs were implemented in the next step.

Based on the results obtained from the tables and figures in this thesis, it can be clearly seen that the Coalition-based ensemble algorithm has been shown to output a greater performance than genetic programming, showing better accuracy in the data sets presented and larger areas under the curve which means a better predicted output.

The results represented in tables 6.5 and 6.6 shows the comparison the results of different combination methods conventional and advanced methods. Average, weighted average, optimized weighted average, voting, GP and coalition method based on AUC output results for all datasets using all training dataset sizes. The table clearly shows that coalition methods produces higher accuracy for all datasets when compared to other methods . A conclusion can be made from both Tables coalition method has shown to give out prominent results when compared to other methods.

Table 6.5: AUC Comparison of six combination methods training performance

Training	Average	Weighted Average	Optimzed Weighted Average	Voting	GP	Coalition
Breast Cancer	0.85605	0.8765	0.85605	0.8605	0.90499	**0.99042**
Bladder Cancer	0.90443	0.90818	0.9042	0.90045	0.89182	**0.97182**
Statlog (Heart)	0.9664	0.96697	0.9664	0.96437	0.95831	**0.95831**
Contraceptive	0.92694	0.92693	0.92678	0.92443	0.89244	**0.89598**
German Credit	0.95993	0.96941	0.96931	0.9692	0.92667	**0.95345**
Australian Credit	0.9761	0.96011	0.95993	0.96004	0.9824	**0.96051**
Ionosphere	0.88259	0.88482	0.88259	0.87863	0.96664	**0.96501**

Table 6.6: AUC Comparison of six combination methods testing performance

Training	Average	Weighted Average	Optimzed Weighted Average	Voting	GP	Coalition
Breast Cancer	0.82786	0.84284	0.82522	0.82811	0.98457	**0.97853**
Bladder Cancer	0.69006	0.76708	0.71863	0.68323	0.77391	**0.95093**
Statlog (Heart)	0.95248	0.95466	0.95248	0.94907	0.95342	**0.99057**
Contraceptive	0.79275	0.79327	0.79283	0.79554	0.66602	**0.80665**
German Credit	0.97061	0.92525	0.92543	0.92507	0.78728	**0.95705**
Australian Credit	0.97061	0.97122	0.97061	0.97061	0.91388	**0.95798**
Ionosphere	0.72402	0.7122	0.72398	0.69451	0.99531	**0.97122**

6.4 Summary

One of the most established areas of research in machine learning and pattern recognition is combination methods. This research talked about two well-known advanced combination methods in machine learning which are Genetic Programming and Coalition-based ensemble algorithms. Genetic Programming is an autonomous program generator that depends on the principle of natural selection in order to breed computer programs. In multi-agent systems, coalition formation works by generating a group in order to cooperatively solve problems. The research began by mentioning a brief introduction to combination methods and its necessity and practicality in machine learning in which many applications have shown to provide a good outcome when combination methods are applied. It then moved to discussing genetic programming and the different processes for creating such algorithms and some examples were provided to further explain how this program operates. The final section talked about Coalition-based algorithms and started by mentioning the basic structure of a coalition and then talked about the different coalition methods and briefly mentioned the principle behind their operation.

Chapter 7: Conclusions and Future Work

7.1 Conclusions

The primary research aim of this thesis is to develop a methodology to improve classifier accuracy by designing a combiner methodology or techniques. This aim was achieved by developing a combiner method using an ensemble combination model and an advanced algorithms such as GP and coalition method based on three model of ANN and different factors such as the ration of training data size, classifier type, size and the type of the combination methods.

In this thesis the introduction of coalition and genetic programming methods has proven that the prediction accuracy can be improved using a simple yet effective approach that is based on diversity to guide the process of creating the optimal combinations of classifiers. The use of a coallation of different classifiers can improve the accuracy by combining the strength of the classifiers (and eliminating the weakness of some delete???) to arrive at an accurate prediction.

The work presented in this thesis proved that the use of an advanced combination method such as GP and coalition methods can outperform the classic combiners such as voting and average methods with the use of the same individual classifiers. The proposed algorithms were compared to stand-alone classifiers and combined classifiers in terms of accuracy (accuracy, sensitivity, specificity, ROC and AUC). The results showed improvements over the other methods, coalition method produces higher accuracy for all datasets when compared to other methods.

Neural network models can usually generate different types of models with different accuracies by just starting with different initial weights each time with the same data sets. This disadvantage can be eliminated with the use of a simple ensemble. The simple ensemble is based on many copies of the same model but with different learning initial conditions.

Despite the fact that all the models are based on the same data, the models are not exactly the same, however, such models are not diverse. In order to introduce diversity to the ensemble, different types of neural networks models are used which provide diversity. However, simple combination methods are not sufficient to extract all the models' accuracy and discard the erroneous areas. Hence other combination methods were used to improve the model with diverse models and better accuracy.

The last contribution of this thesis is the new ensemble combining method based on the consensus theory and another based on evolutionary algorithms. In order to demonstrate the merits of the proposed combination methods, extensive experimentations are carried out on different data bases and compared to the standard and single combiners such as voting, average, weighted average and optimised weighted average. The obtained results demonstrate the merits of the proposed methods.

7.2 Future Work

The research in this thesis has focused primarily on improving the performance of classifier ensembles by building the optimal combiner for a certain data set. However, the combiner method, being a customised algorithm, can be applied to any ensemble technique or method. One of the proposed future work avenues is the use of other ensemble methods and integrating into the current combiner.

The combiner algorithms are tested extensively on several UCI repository data bases as well as other data bases such as the bladder cancer and the credit scoring databases. However, most of the data bases are fairly low-objects, binary class, having a few hundred entries. It is recommended that larger data bases be used with multi-class features. The scale of the data base should consist of tens of thousands of examples such as text categorisation data sets. It will be a more rigorous test of the algorithms in such domains.

The ensembles used in this work are based on a combination of different types of neural networks. The work did not involve the use of other types of modelling tools, such as neuro-fuzzy systems, support vector machines, conventional classifiers, and Bayes' classifiers. An ensemble of a multi-type classifier would be more interesting.

Other data sets with bias and variance can also be used to test the algorithms in terms of the accuracy which will give more insights into the algorithms.

References

Abraham, A. (2001) ‘Neuro-fuzzy systems: State-of-the-art modelling techniques’ *Connectionist models of neurons, learning processes, and artificial intelligence*, Springer Berlin Heidelberg, pp. 269-276.

Adeva, J Beresi, and U Calfo, R., (2005) Accuracy and Diversity in Ensembles of Text Categorisers. *Clei Electronic Journal*. 8 (1), pp. 2-3.

Adrignola (2010) *Artificial Neural Networks* . [Online]Available at: http://en.wikibooks.org/wiki/Artificial_Neural_Networks. Last accessed 16 May 2015.

Amiri, R. and Esna, M. (2012) 'Comparison between Artificial Neural Networks and Mathematical Models for Equilibrium Moisture Content Estimation in Raisin' *CIGR Journal* 12 (1305), pp. 7-9.

Bishop, C. M and Roach, C. M. (1992) 'Fast curve fitting using neural networks' *American Institute of Physics* 10 (63), pp. 4450-4456.

Bull, L (2004) 'Applications of Learning Classifier Systems. 1st ed. UK: `*Springer-Verlag* Berlin Heidelberg.

Bullinaria,J.A.(2014). *Recurrent Neural Networks* . [Online] Available at: http://www.cs.bham.ac.uk/~jxb/INC/l12.pdf. Last accessed 21st May 2015.

C. Klimasauskas.(1993) *Applying neural networks. In Neural Networks in Finance and Investing*, chapter 3, pp. 47–72. Probus Publishing Company.

C. Klimasauskas. (1993) *Applying neural networks. In Neural Networks in Finance and Investing* chapter 3, pp. 47–72. Probus Publishing Company.

Chalkiadakis, G. Elkind, E., Markakis, E and Jennings, N.R. (2008) *Overlapping Coalition Formation. International Workshop on Internet and Network Economics* 4th, pp. 307-321. [Online] Available at: http://users.ecs.soton.ac.uk/gc2/Papers/wine-OCFcemj.pdf [Accessed 01 August 2015].

Cruz, J. A. and Wishart, D. S. (2006) 'Applications of Machine Learning in Cancer Prediction and Prognosis' *Cancer Informatics*. 2 (3), pp. 59-77.

Cung, B Hsueh, J Kolesnikov, L and Lu, K. (2011). *Regression and Machine Learning* p. 5.

Czogała, E., and Łęski, J. (2000) ‘Neuro-fuzzy systems’, *Fuzzy and Neuro-Fuzzy Intelligent Systems*, Physica-Verlag HD, pp. 141-162.

Dheeraj, S. , Anil, K. ,and Vinayak, K. (2014) 'Cascade and Feed Forward Back propagation Artificial Neural Network Models for Prediction of Compressive Strength of Ready Mix Concrete' .*IOSR Journal of Mechanical and Civil Engineering (IOSR-JMCE)* 3 (3), pp1-6.

Dietterich, T. (2014) *Ensemble Methods in Machine Learning*, p. 13.

Dietterich, T. G. (2000) 'Ensemble Methods in Machine Learning' In J. Kittler and F. Roli (Ed.) *First International Workshop on Multiple Classifier Systems, Lecture Notes in Computer Science*. New York: Springer Verlag, pp. 1-15.

Dietterich, T.G. (2009). *Ensemble Methods in Machine Learning* .[Online] Available at: http://web.engr.oregonstate.edu/~tgd/publications/mcs-ensembles.pdf. Last accessed 4th July 2015.

Duin, R. P. W., Juszczak, P., Paclik, P., Pekalska, E., De Ridder, D., Tax, D. M. J., and Verzakov, S. (2000) ‘A matlab toolbox for pattern recognition’, *PRTools version*, *3*, pp. 109-111.

Duin, R. P., and Tax, D. M. (2000), ‘Experiments with classifier combining rules’, In *Multiple Classifier Systems*, Springer Berlin Heidelberg, pp. 16-29.

Ford, W. (2012) Classifying lung cancer recurrence time using novel ensemble method with gene network based input models. *Procedia Computer Science*. 12 (1), pp. 444-449.

George, W.K. (2011) *Introduction to turbulence/Statistical analysis/Ensemble average*. [Online] Available at: http://www.cfdonline.com/Wiki/Introduction_to_turbulence/Statistical_analysis/Ensemble_average. [Accessed 03 August 15].

Godara, S and Gupta, R. (2012) 'Comparison of Different Neural Networks for Iris Recognition: A Review' *Network and Complex Systems*. 2 (4), pp. 8-12.

Godara, S. and Gupta, R. (2012) Comparison of Different Neural Networks for Iris Recognition: A Review. *Network and Complex Systems*. 2 (4), pp. 8-12.

Heaton, J. (2008) *Feedforward Backpropagation Neural Networks* [Online]Available at: http://www.heatonresearch.com/online/introduction-neural-networks-cs-edition-2/chapter-5. Last accessed 5th May 2015.

http://pages.bangor.ac.uk/~mas00a/papers/cslkif02.pdf [Accessed 14 July 2015].

http://www.acsu.buffalo.edu/~tulyakov/papers/tulyakov_MLDAR_comb_review.pdf. [Accessed 09 July 15].

Iliadis, L. (2013) 'Engineering applications of neural networks: 14th International Conference' *EANN 2013, Halkidiki, Greece,* September 13-16, 2013, Proceedings.

Jimenez, D., 1988 'Dynamically weighted ensemble neural networks for classification' *In The 1998 IEEE International Joint Conference on Neural Networks Proceedings*. Anchorage, AK, 4 May 1998. San Antonio, TX: IEEE. pp. 753 – 756, vol.1.

K. Bergerson and D. Wunsch (1993) *A commodity trading model based on a neural network-expert system hybrid. In Neural Networks in Finance and Investing*, chapter 23, pp. 403–410. Probus Publishing Company.

Khashman, A. (2006) *Face Recognition Using Neural Networks and Pattern Averaging*. [Online] Available at: http://link.springer.com/chapter/10.1007%2F11760023_15. [Accessed 22 July 15].

Kim, H. C., & Ghahramani, Z. (2012) 'Bayesian classifier combination', *International conference on artificial intelligence and statistics,* pp. 619-627.

Kononenko, I. (2005) *Machine Learning for Medical Diagnosis: History, State of the Art and Perspective,* p. 5.

Kotsiantis, S. B Kanellopoulos, D and Zaharakis, I. D. (2006) Bagged Averaging of Regression Models. In: Maglogiannis, I Karpouzis, K Bramer, M *Artificial Intelligence Applications and Innovations*. US: Springer US. pp. 53-60.

Koza, J.R, 2003. What is Genetic Programming?. [Online] Available at: http://www.genetic-programming.com/gpanimatedtutorial.html. [Accessed 07 August 15].

Koza, J.R. (1992)' Genetic Programming: On the Programming of Computers by Means of Natural Selection'. 1st ed. Cambridge, MA: The MIT Press.

Krishnapuram, B., Carin, L., Figueiredo, M. A., and Hartemink, A. J. (2005) 'Sparse multinomial logistic regression: Fast algorithms and generalization bounds', *Pattern Analysis and Machine Intelligence, IEEE Transactions on, 27*(6), pp. 957-968.

Lee, H. and Chen, C. (2006) 'Multi-Agent Coalition Formation for Long-Term Task or Mobile Network' *In Computational Intelligence for Modelling, Control and Automation* 2006 and International Conference on Intelligent Agents, Web Technologies and Internet Commerce, International Conference on. Sydney, NSW, 2006. Sydney, NSW: IEEE. 52.

Mache, N. (1995) *TDNN Fundamentals* .[Online] Available at: http://www.ra.cs.uni-tuebingen.de/SNNS/UserManual/node176.html. Last accessed 15 May 2015.

Majors, D. (2002) *An Introduction to Learning Classifier Systems*. CS401 Introduction to Evolutionary Computation. 2004, Missouri University of Science and Technology.

Martin, D. R., Fowlkes, C. C., and Malik, J. (2002) 'Learning to detect natural image boundaries using brightness and texture' In *Advances in Neural Information Processing Systems*, pp. 1255-1262.

McCluskey, P.G., 1993. *Feedforward and recurrent neural networks and genetic programs for stock market and time series forecasting*. Technical Report CS-93-36, Brown University, September 93.

Melville, P. (2003), *Creating Diverse Ensemble Classifiers*, pp. 34.

Moradi, M., and Zulkernine, M. (2004) 'A neural network based system for intrusion detection and classification of attacks' In *Proceedings of the 2004 IEEE international conference on advances in intelligent systems-theory and applications*, pp. 286-295.

Naftaly, U Intrator, N and Horn, D. (1996) 'Optimal ensemble averaging of neural networks' *Network: Computer Neural System*. 8 (2), pp. 283–296.

Naftaly, U., N. Intrator, and D. Horn (1997) Optimal ensemble averaging of neural networks. *Network: Computation in Neural Systems*, 8, no. 3, pp.283–296.

Navone, H.D. (2001) A Learning Algorithm For Neural Network Ensembles. *Revista Iberoamericana de Inteligencia Artificial*. 12 (1), pp. 70-74.

Obst, O, and Riedmiller, M. (2013). *Taming the Reservoir: Feedforward Training for Recurrent Neural Networks* .[Online] Available at : http://ml.informatik.uni-freiburg.de/_media/publications/obstrieijcnn12.pdf. Last accessed 16 May 2015.

Orr, M. J. T.(1996)*Introduction to Radial Basis Function Network.* [Online] Available at: http://www.cc.gatech.edu/~isbell/tutorials/rbf-intro.pdf. Last accessed 5th May 2015.

Oza, N.C. (2008) Classifier ensembles: Select real-world applications. *Information Fusion*. 9 (1), pp. 4-20.

Paiva, R. P., and Dourado, A. (2004) 'Interpretability and learning in neuro-fuzzy systems', *Fuzzy sets and systems*, *147*(1), pp. 17-38.

Papik, K. (1998) Application of neural networks in medicine - a review. *Diagnostics and Medical Technology*. 4 (3), pp. 538-546.

Poggio, G. C. T. (2001) 'Incremental and decremental support vector machine learning' In *Advances in Neural Information Processing Systems 13: Proceedings of the 2000 Conference*, Vol. 13, MIT Press, p. 409.

Polikar, R. (2009), *Ensemble learning* .[Online] Available at: http://www.scholarpedia.org/article/Ensemble_learning. [Accessed 10 February 15].

Polikar, R. (2009) *Ensemble Learning* .[Online] Available at: http://www.scholarpedia.org/article/Ensemble_learning. [Accessed 02 July 15].

Polikar, R. (2006) 'Ensemble based systems in decision making' *Circuits and Systems Magazine IEEE*. 6 (3), pp. 21-45.

Polikar, R. (2006) 'Ensemble based systems in decision making' *Circuits and Systems Magazine, IEEE*. 6 (3), pp.21-45.

R. Amiri and M. Esna. (2012) 'Comparison between Artificial Neural Networks and Mathematical Models for Equilibrium Moisture Content Estimation in Raisin' *CIGR Journal* 12 (1305), pp. 7-9.

Rahwan, T. (2007) Algorithms for Coalition Formation in Multi-Agent Systems. PhD thesis, University of Southampton. Available from: BCS.

Ramoni, M., and Sebastiani, P. (2001) 'Robust Bayes classifiers' *Artificial Intelligence*, *125*(1), pp. 209-226.

Robert J. and Van Eyden (1996) 'The Application of Neural Networks in the Forecasting of Share Prices' Finance and Technology Publishing.

Rokach, L. (2005) Ensemble Methods for Classifiers. In: Maimon, O. *Data Mining and Knowledge Discovery Handbook*. Tel Aviv: Tel-Aviv University. pp. 957-973.

Rokash, L. (2010) Ensemble-based classifiers. *Springer Science + Business Media*. 33 (1), pp. 1-39.

Rouse, M., (2005) Genetic Programming. [Online] Available at: http://whatis.techtarget.com/definition/genetic-programming. [Accessed 11 August 15].

Russel, I. (1996) *Definition of a Neural Network* .[Online] Available at: http://uhaweb.hartford.edu/compsci/neural-networks-definition.html. Last accessed 15 May 2015.

Russel, I. (1996) *Definition of a Neural Network* .[Online] Available at: http://uhaweb.hartford.edu/compsci/neural-networks-definition.html. Last accessed 15 May 2015.

Russell, M. J., Rubin, D.M., Marwala, T and Wigdorowitz, b. (2009) A voting and predictive Neural Network system for use in a new artificial Larynx. *In Biomedical and Pharmaceutical Engineering, 2009. ICBPE '09. International Conference*. Singapore, 2 December 2009. Johannesburg, South Africa: IEEE. pp. 1-4.

Russell, S.J. and Norvig, P. (2003) Artificial Intelligence: A Modern Approach. 2nd ed. Upper Saddle River, New Jersey: Prentice Hall.

Shiffman, D., (2012) 'Neural Networks' In: Fry, S *THE NATURE OF CODE*. USA: Free Software Foundation. pp. 162-180.

Shiffman, D. (2012) 'Neural Networks' In: Fry, S *THE NATURE OF CODE*. USA: Free Software Foundation. pp. 162-180.

Shipp, C.A. and Kuncheva, L.I. (2002) Relationships between combination methods and measures of diversity in combining classifiers. *Information Fusion* 3(2) pp.135-148 [online] Available at:http://pages.bangor.ac.uk/~mas00a/papers/cslkif02.pdf [Accessed 14 July 2015].

Stergiou, C and Siganos, D. (2015) *Neural Networks.* [Online] Available at: http://www.doc.ic.ac.uk/~nd/surprise_96/journal/vol4/cs11/report.html. Last accessed 28th May 2015.

Stergiou, C. and Siganos, D. (2015) *Neural Networks.* [Online] Available at: http://www.doc.ic.ac.uk/~nd/surprise_96/journal/vol4/cs11/report.html. Last accessed 28th May 2015.

TechTarget (2015) *Neural Network Definition* .[Online] Available at: http://searchnetworking.techtarget.com/definition/neural-network. Last accessed 30th May 2015.

TechTarget (2015) *Neural Network Definition.* [Online] Available at: http://searchnetworking.techtarget.com/definition/neural-network. Last accessed 30th May 2015.

Tong, S., and Koller, D. (2002) 'Support vector machine active learning with applications to text classification', *The Journal of Machine Learning Research*, *2*, pp. 45-66.

Tulaykov, S and Jaeger, S. (2007) Review of Classifier Combination Methods. [Online] Available at: http://www.acsu.buffalo.edu/~tulyakov/papers/tulyakov_MLDAR_comb_review.pdf. [Accessed 09 July 15].

Tulaykov, S. and Jaeger, S. (2007) *Review of Classifier Combination Methods*. [Online] Available at: http://www.acsu.buffalo.edu/~tulyakov/papers/tulyakov_MLDAR_comb_review.pdf. [Accessed 09 July 15].

Van Eyden, R.J. (1996) The Application of Neural Networks in the Forecasting of Share Prices. Finance and Technology Publishing, 1996.

Wang, S. (1999) *An adaptive approach to market development forecasting. Neural Computing and Applications*. 8(1): pp. 3-8.

Wolfram, S. (2015) *Radial Basis Function Networks.* [Online] Available at: http://reference.wolfram.com/applications/neuralnetworks/NeuralNetworkTheory/2.5.2.html. Last accessed 16th May 2015.

Xue, J. H., and Titterington, D. M. (2008) 'Comment on discriminative vs. generative classifiers: A comparison of logistic regression and naive Bayes' *Neural processing letters*, *28*(3), pp. 169-187

Yaochu , J. (2008) Pareto-Based Multiobjective Machine Learning: An Overview and Case Studies.*Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions,* [Online]. 38(3), pp. 397 - 415 .[Online] Available at: http://www.soft-computing.de/SMC0805.pdf [Accessed 02 July 2015].

Ye, F., Zhang, Z., Chakrabarty, K. and Gu, X. (2013) Board-Level Functional Fault Diagnosis Using Artificial Neural Networks, Support-Vector Machines, and Weighted-Majority Voting. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, [Online]. 32 (5), pp. 723 - 736 .[Online] Available at: http://www.researchgate.net/publication/260584899_Board-Level_Functional_Fault_Diagnosis_Using_Artificial_Neural_Networks_Support-Vector_Machines_and_Weighted-Majority_Voting [Accessed 04 August 2015].

Zarafshan, F., Gholam, S. and Karimi, A.(2010) Notice of Retraction A novel weighted voting algorithm based on neural networks for fault-tolerant systems. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions*, [Online]. 9, 135 - 139. [Online] Available at: http://www.researchgate.net/publication/224173078_A_novel_weighted_voting_algorithm_based_on_neural_networks_for_fault-tolerant_systems [Accessed 08 July 2015].

Zhang, G. P. (2000) 'Neural networks for classification: a survey', *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, *30*(4), pp. 451-462.

Zhou, H. (2015). *Machine Learning for Medical Applications*, p. 1.

Zhou, Z. H., Wu, J., and Tang, W. (2002) 'Ensembling neural networks: many could be better than all', *Artificial intelligence*, *137*(1), pp. 239-263.