STOCK SELECTION USING GENERAL GROWING AND PRUNING RADIAL BASIS FUNCTION (GGAP-RBF) NEURAL NETWORK

NG WEE DING

SCHOOL OF ELECTRICAL & ELECTRONIC ENGINEERING

A DISSERTATION SUBMITTED TO THE NANYANG TECHNOLOGICAL UNIVERSITY IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE DEGREE OF MASTER OF SCIENCE IN COMMUNICATION SOFTWARE AND NETWORKS

2006

Statement of Originality

I hereby certify that the content of this dissertation is the result of work done by me and has not been submitted for a higher degree to any other University or Institution.

Date

Ng Wee Ding

Acknowledgement

I would like to express my sincere gratitude and appreciation to my supervisor, Dr. Quah Tong Seng, for his valuable guidance, support, patient and ideas given to me throughout the duration of my candidature for this dissertation. I would also like to thank the entire technical support staff of Information Communication Institute of Singapore (ICIS) practicum laboratory for their logistic support and technical know-how.

Summary

Stock Selection using GGAP-RBF (General Growing and Pruning Radial Basis Function) Neural Network

In the stock picking or selection problem, the motivation is to design system such that it is able to systematically select stocks or equities that are empirically predicted to perform well in the future. Thus, it should achieve four objectives: 1) to achieve monetary returns in the future; 2) to beat the average market performance; 3) to avoid the bias in human-based stock selection due to psychological reasons; 4) automatically crunching the data to extract useful information based on past information, which can be hardly done manually. Hence, this dissertation presents methodologies to select equities based on soft-computing models and the information gathered from fundamental analysis of stock performance.

It first gives a broad literature review on neural networks and fuzzy logic; follow by an introduction on their existence in financial market. It then defines the equities selection problem which includes identifying the relevant features. With the necessary work on data preprocessing, it then compare GGAP-RBF (General Growing and Pruning Radial Basis Function) with another two soft-computing models, namely MLP (Multi-level Perceptrons) and ANFIS (Adaptive Neuro-Fuzzy Inference Systems). It studies their computational time complexity; applies several benchmark metrics to compare their performance, such as generalization rate, recall rate, confusion matrices, and correlation to appreciation. With these extensive experiments, we demonstrate that MLP and ANFIS both perform better in terms of correlation of the predicted output with respect to the actual appreciation of the stocks. On top of this, we also empirically demonstrated that ANFIS and MLP have better computational complexities than the GGAP-RBF network in this equities selection problem.

Based on the benchmarking results presented in the first set of experiments, this dissertation proposes a systematic methodology to select equities by utilizing the

concept of ROC (Relative Operating Characteristics) curve. This is presented in the second set of experiments in the dissertation. Having quantitatively analyzed the results of ROC curves, we systematically identify the cut-off-point of the system to achieve the optimal results. The test results of these experiments are encouraging. Finally, we demonstrates that our system achieve better appreciation in equities comparing with others when we choose to trade the top ten selected stocks in the market.

Table of Contents

List of Figures

List of Tables

Chapter 1: Introduction

1.1 Motivation

There has been active research on applying soft-computing models on investment since decade ago. The major motivation is to develop an expert system to resemble the decision making process of investment experts. Soft-computing models are attractive as it offers a method to formulate the non-linear, noisy and non-deterministic environments. As the cost of computational power decreases, we can afford to have more complex techniques which are expected to have lower signal-to-noise ratio. Soft-computing models are able to have better investment returns.

There are two branches for applying soft-computing models on investment, which are technical analysis and fundamental analysis. Technical analysis is the most popular area in research due to its less noisy environment. Generally, it applies time-series prediction and pattern recognition. Such work includes equities price and volume movement. Technical analysis does not consider the underlying factors of equities' financial health profile. Intuitively, this is only useful for short-term trading decision making.

Fundamental analysis is mostly for long-term investment decision. Accounting variables and financial ratios are usually used for inspecting the health of the investment products. There are fewer research papers on this area as compared to technical analysis. However, we believe that fundamental data will impact the price movement in the long-run. This motivates us to conduct a study on fundamental analysis with the three selected soft-computing models.

1.2 Objectives

The objectives of this dissertation are: -

- To formulate the problems definition for our studies, which include feature selection and data preparation
- To collect the financial data from Bloomberg system
- To determine the architectures of the soft-computing models to be used
- To do experiments and study the performance of the models
- To propose a systematic method for equities picking based on confusion matrices and validate the method with experiment results

1.3 Organization of Dissertation

This dissertation consists of a total of five chapters. Chapter 1 gives brief introduction on the motivation as well as a list of the objectives to be fulfilled. Chapter 2 discusses the background knowledge of the neural network, fuzzy logic as well as the relationship of soft-computing in financial market. It also illustrates the current challenges of applying soft-computing in financial market.

Chapter 3 defines our equities selection problems. It discusses the features selection and the architecture of the selected under-studied soft computing models. The last part of Chapter 3 explains the confusion matrix, which will be applied for performance analysis.

We present the experiments results in Chapter 4. The details of the data preparation and training results are shown. The results of first experiment validate the performance of the three soft-computing models. The results of the proposed equities picking method are shown in the last part of Chapter 4.

We conclude our studies in Chapter 5. A brief discussion on future work is also included.

Chapter 2: Literature Reviews

2.1 Basics of Soft Computing

Soft computing is useful to solve problems described by multiple variables and multiple parameters. These problems may have non-linear coupling among these variables and parameters which are extremely difficult to find mathematical solutions. Therefore, it can be very costly to find solutions for such problems. To deal with such problems, one has to trade off the complexity with the uncertainties and imprecision. Thus, soft computing comes into picture. Soft computing exploits the tolerance for imprecision, uncertainty, partial truth and approximation to achieve tractability, robustness and low solution cost [2]. The principal constituents of soft computing are: Fuzzy logic, artificial neural networks, probabilistic reasoning, evolutionary computation, machine learning and chaos theory, etc.

Literature reviews in Section *2.1.1 Introduction of Neural Network* and Section *2.1.2 Introduction of Fuzzy Logic* are mostly summarized from MATLAB Help Documentation [19].

2.1.1 Introduction of Neural Network

Neural network are composed of simple elements operating in parallel. These elements are inspired by biological nervous systems. As in nature, the network function is determined largely by the connections between elements. We can train a neural network to perform a particular function by adjusting the values of the connections (weights) between elements [19].

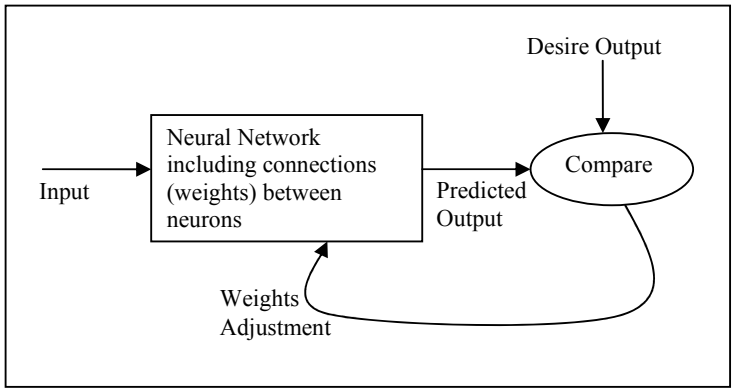

Figure 1: Basic structure of supervised-learning Neural Network

Supervised training methods are most commonly used. As shown in Figure 1, the network is adjusted based on the comparison of the predicted output and the desire output, or target, until the network output matches the target. These input/target pairs are provided to train the network.

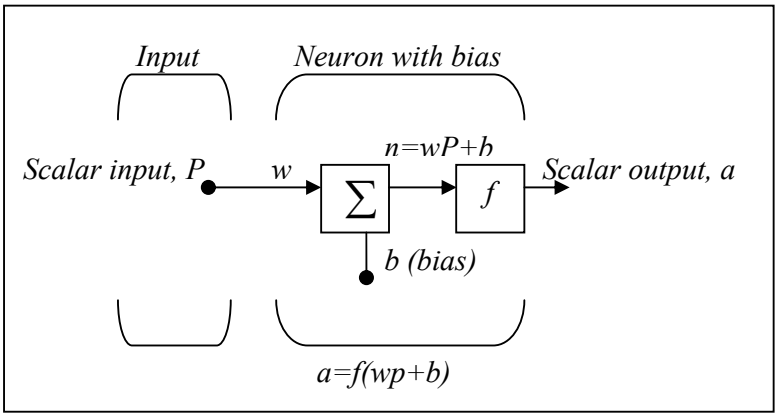

Figure 2: Single Neuron (Neuron with bias)

A typical single neural network structure is shown as Figure 2. The sum of weighted scalar input *(wP)* with the bias *(b)* forms the input to the transfer function, *f*. Transfer functions transfer inputs to defined outputs. Typical transfer functions are hard-limit transfer function, linear transfer function and log-sigmoid transfer function. For example, hard-limit transfer function limits the output of the neuron to either 0, if the net input *(n)* is less than 0; or 1, if *n* is greater than or equal to 0. Linear transfer function transfers the inputs to outputs linearly. The sigmoid transfer function takes the input, which may have any value between plus and minus infinity, and squashes

the output into the range 0 to 1. Multilayer networks often use the log-sigmoid transfer function.

2.1.1.1 Introduction of Multi-level Perceptrons (MLP)

Figure 3 shows a single layer of neurons. It contains *S* neurons and *R* inputs in the layer. In the network, each element of input vector *P* is connected to each neuron input through the weight matrix *w*. The *ith* neuron has a summation that gathers its weighted inputs and bias to form its own scalar output *n(i)*. The various *n(i)* taken together form an *S*-element net input vector *n*. Finally, the neuron layer outputs form a column vector *a*. Multi-layer network can be created by feeding the outputs of layer to be input of next layer.

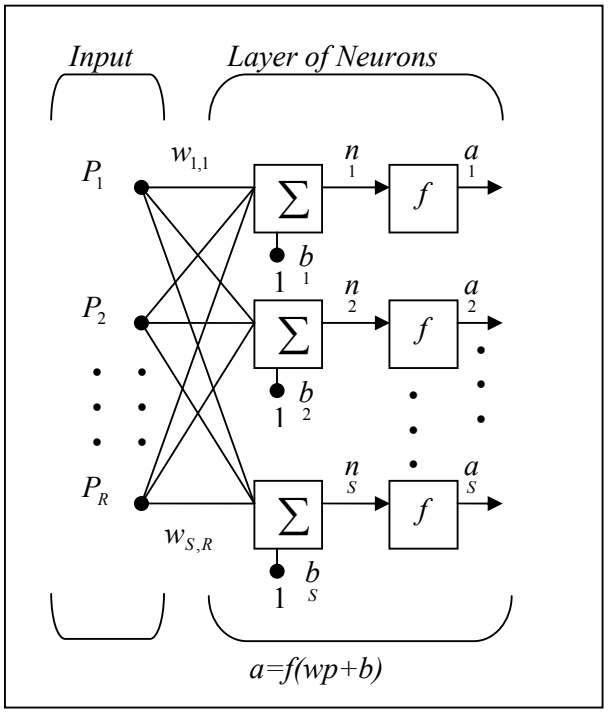

Figure 3: A Layer of Neurons

[19] defines learning rule as a procedure for modifying the weights and biases of a network, as known as training algorithm. The learning rule is applied to train the network to perform some particular task. Learning rules may be broadly categorized as supervised learning and unsupervised learning. In supervised learning, the learning rule is provided with a set of examples (training set) which contain many pairs of

inputs and target outputs. As the inputs are applied to the network, the network outputs are compared to the targets. The learning rule is used to adjust the weights and biases of the network in order to move the network outputs closer to the targets. In unsupervised learning, the weights and biases are modified in response to network inputs only. There are no target outputs available. Most of these algorithms perform clustering operations. They categorize the input patterns into a finite number of classes. This is especially useful in such application as vector quantization.

2.1.1.2 Introduction of Radial Basis Neuron (RBF)

[33] defines RBF as means for interpolation in a stream of data as it has built into a distance criterion with respect to centre. Figure 4 shows a radial basic function network with R inputs. The net input to the *radbas* transfer function is the vector distance between its weight vector W and the input vector P, multiplied by the bias b. The radial basis function has a maximum of 1 when its input is 0. As the distance between W and P decreases, the output increases. Thus, a radial basis neuron acts as a detector that produces 1 whenever the input P is identical to its weight vector P.

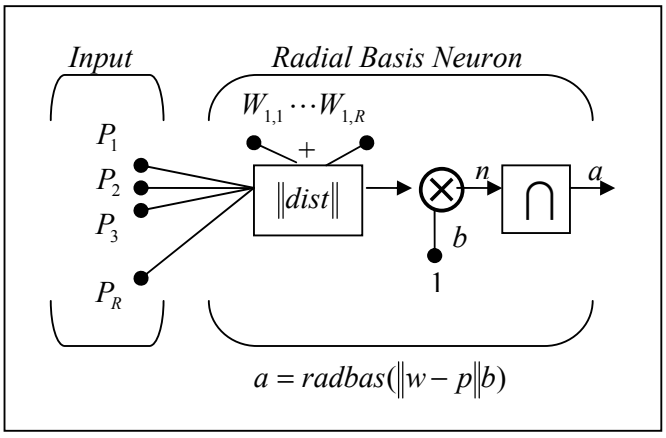

Figure 4: Radial Basic Function Network

2.1.2 Introduction of Fuzzy Logic

The point of fuzzy logic is to map an input space to an output space, and the primary mechanism for doing this is a list of if-then statements called rules. All rules are evaluated in parallel, and the order of the rules is unimportant. Fuzzy inference is a method that interprets the values in the input vector and, based on same set of rules, assigns values to the output vector [19].

A *membership function* (MF) is a curve that defines how each point in the input space is mapped to a membership value (or degree of membership) between 0 and 1. The input space is sometimes referred to as the *universe of discourse*, a fancy name for a simple concept [19]. For example, we have a Body Mass Index (BMI) analysis system. The universe of discourse is all potential BMI, say 0 to 50. The word "*overweight*" would correspond to a curve that defines the degree to which any person is overweight. We may say that all people with BMI larger than 25 are officially considered overweight, as shown as Figure 5.

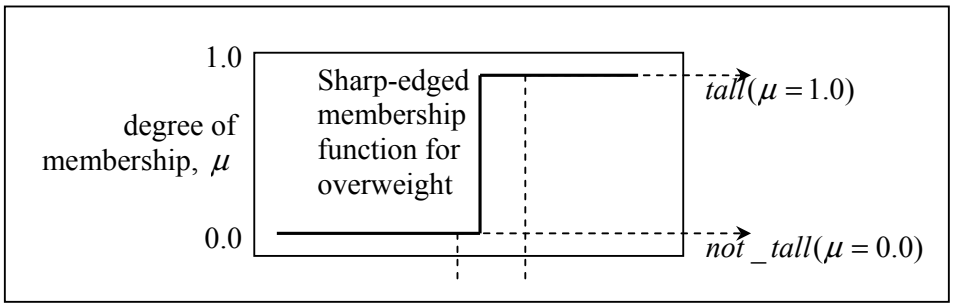

Figure 5: Membership function of overweight problem

However, such a distinction is clearly absurd. This is because it is unreasonable to call one person overweight and one underweight while their BMI only differs by a little, say 0.1. We may define the set of overweight people with a smoothly varying curve that passes from not-overweight to overweight as shown as Figure 6. The curve is known as a membership function and is often given the designation of μ. The output-axis is a number of known as the membership value between 0 and 1. This curve defines the transition from not overweight to overweight. For example, person with μ of 9.5 is more overweight than person with μ of 3.0.

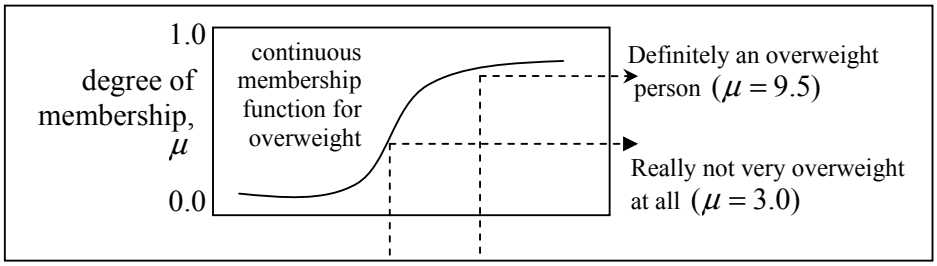

Figure 6: Continuous membership function for overweight problem

We now know what is fuzzy about fuzzy logic, but what about the logic? Let us say that we have fuzzy values of 1 (completely true) and 0 (completely false) for two set *A* and *B*; and apply standard logical operations.

A	B	A And B
0	0	**0**
0	1	**0**
1	0	**0**
1	1	**1**

A	B	A Or B
0	0	**0**
0	1	**1**
1	0	**1**
1	1	**1**

A	Not A
0	**1**
1	**0**

Table 1: Truth Table

Now, consider the above truth table. When we say *A AND B*, it actually performs *min* operation, where *A* and *B* are limited to the range (0,1), by using function *min(A,B)*. Using the same reasoning, we can replace *OR* operation with the *max* function. Finally, the operation *NOT A* becomes equivalent to the operation *1-A*. This is applied to fuzzy set in the same way. The truth table can be further converted to plot of two fuzzy sets applied together to create one fuzzy set, as shown in Figure 7 below. The upper part of the figure displays plots corresponding to the two-valued truth tables above, while the lower part of the figure displays how the operations work over a continuously varying range of truth values *A* and *B* according to the fuzzy operations we have defined.

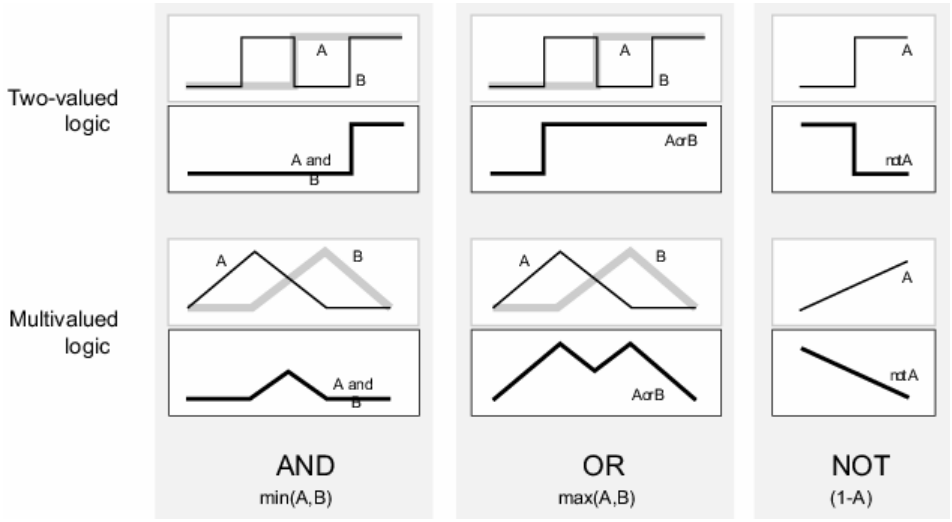

Figure 7: True Table for Fuzzy sets

(directly extracted from MATLAB Help [19]*)*

Given these three functions, we can resolve any construction using fuzzy sets and the fuzzy logical operation *AND*, *OR* and *NOT*. In short, there are three steps in fuzzy system, first is to fuzzify inputs based on the rules, subsequently applying logical operators on the fuzzified inputs, lastly apply implication operator to shape the output fuzzy set.

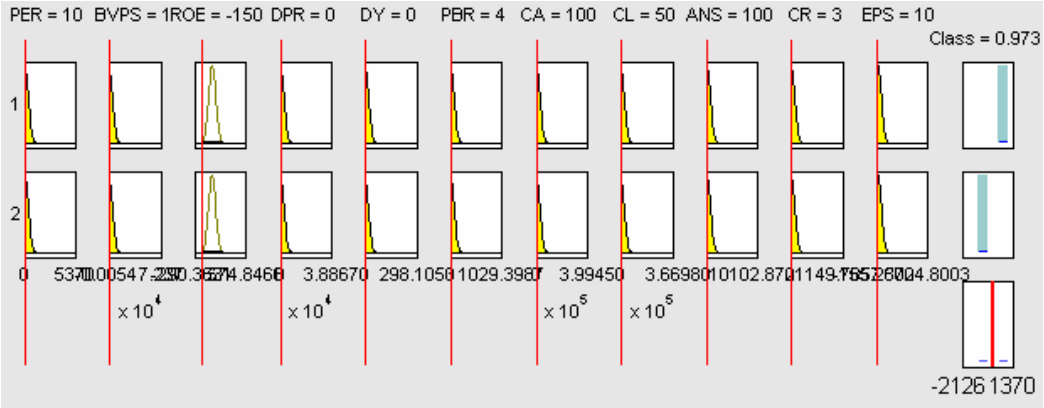

Figure 8: Fuzzy Inference System

Fuzzy inference is the process of formulating the mapping from a given input to an output using fuzzy logic. We present the fuzzy inference diagram of our equities picking problem in Figure 8. There are two rules with 11 attributes for each role. Every attributes will have a member function to fuzzify the input of the attributes. For

this example, *AND* operator is used on the fuzzified inputs for each rule. The fuzzy output of the rule is shown in the first right column; we named this column as "Class" here. The implication operator is used for the rules outputs to shape the fuzzy output set.

For example, the ANFIS system [17], which is an instance of the more generic form of Takagi-Sugeno-Kang (TSK) model [27] replaces the fuzzy sets in the implication with a first order polynomial equation of the input variables. Generally, a *r*-input one-output ANFIS system consists of rules in the following form: -

R_i: IF (x_1 is A_{i1}) and…and (x_r is A_{ir})

THEN $y = f_i(x_1, x_2, \ldots, x_r) = b_{i0} + b_{i1}x_1 + \ldots + b_{ir}x_r$

Where:

- x is the input vector,
- A_i is the fuzzy membership function,
- f_i is a first order polynomial function, and
- b_{ij} j=0,1,…,r are real-valued parameters.

The fuzzy inference performed by ANFIS is an interpolation of all the relevant rules based on the physical location of the input data in the fuzzy subspaces. The predicted output of the model is generally given by the following equation: -

$$y = \frac{\sum_{i=1}^{L} \alpha_i f_i(x_1, x_2, \ldots, x_r)}{\sum_{i=1}^{L} \alpha_i} = \frac{\sum_{i=1}^{L} \alpha_i (b_{i0} + b_{i1}x_1 + \ldots + b_{ir}x_r)}{\sum_{i=1}^{L} \alpha_i}$$

where: α_i is the matching degree of rule R_i, which is computed by considering the product of all the relevant membership functions of the rule.

2.2 Soft Computing in Financial Market

Investment and financial trading problems are usually divided into two disciplines, which are fundamental analysis and technical analysis. Fundamental analysis techniques provide a framework for examining underlying forces which affect the price of an investment; on the other hand, technical analysis techniques analyze the past trading data, which includes prices, volume, open interest, etc. and believing these are reflecting the behavior of market participants [28]. Generally, fundamental analysis is preferred for long-term investment whereas technical analysis is for short-term trading. However, they also complement each other for better trading decisions in the research on trading expert system, for example, the work of [4].

The common soft computing techniques which are applied in both analyses are time series prediction, pattern recognition and classification, as well as optimization. Time series technique forecast future data points using historical data sets, for example, studying at the historical daily closing price in order to predict tomorrow closing price. Pattern recognition and classification try to classify observations into classes, for example, classifying securities into "winner" and "loser" classes. Optimization involves solving problems where patterns in the data are not known, for example, determining the optimal point to enter the securities market [28].

As compared to fundamental analysis, technical analysis is more common technique in the research of solving financial investment problems with soft computing. Nevertheless, the focus of this dissertation is to apply soft computing with fundamental analysis in equities picking on DJIA (Dow Jones Industrial Average Index) equities.

2.3 Challenges

This problem is not as easy as it appears. There are well-known challenges for equities picking with soft computing, such as: -

- Selection of additional features to improve the performance, as suggested by [21] and [10]. Intuitively, the more the features, the more accurate the neural networks performance are. However, the nature of financial market is noisy and stochastic. The stock market itself is not only driven by fundamental data, but also by human psychological factors and market principles. Because of this, the system may suffer the curse-of-dimensionality issue. Hence, the fundamental rule is to select most suitable features but not trying to cover as many features as possible. We will also present the time complexity comparison of the under studied soft computing models.
- Poor predictability accuracy. Due to the non-deterministic nature of the financial market, artificial neural network models may not be able to out-perform significantly but slightly to the logit model [10]. With such, we will present not only the accuracy as one of the performance matrices, but also the appreciation of the picked equities, as [30].
- Data availability [21]. It is highly impossible to obtain all the data that impact the stocks price movements. We need to maximize the accuracy and equities appreciation based on the limited data available.

There are many other challenges, such as trading rules to simulate real life trading system to include trading cost, time management, etc. All and all, the fundamental objective is to build a reliable decision support system to replace expert knowledge in financial world.

Chapter 3: Problems Definition

Our main interests in this dissertation are to find out which soft computing models among the three models is the best for equities picking problems with fundamental analyses, and to maximize the total next-year appreciation of picked equities.

Chapter 3.1 presents the chosen attributes which are used as inputs for neural network models to predict next-year appreciation of equities. The next-year appreciation of equities is defined as: -

$$Next\,Year\,Appreciation,\% = \left(\frac{P_1, Next\,Year\,Market\,\mathrm{Pr}ice}{P_0, Current\,Year\,Market\,\mathrm{Pr}ice} - 1 \right) \times 100\%$$

We classify the equities as either "winner" or "loser" based on the next-year appreciation. The detail of data processing and classification problems will be explained in *Chapter 4.1.2*. *Chapter 3.2* presents the three soft computing models under-studied. *Chapter 3.3* explains the Confusion Matrix which will be used to pick valuable stocks.

3.1 Features Selection

The history of fundamental analysis may be traced back to the work done by Benjamin Graham who is acknowledged as the father of modern security analysis. The work of [12] suggested that it is possible and cheap to have positive risk-adjusted rates of return with Benjamin Graham's common stock selection rules. This indicates the existence of relationship between the returns of the equities and their fundamental attributes, such as price-to-earning ratio, capitalization and size of the firms. This finding spun off much research work, such as [6], [12], [5], [22], [32], [23], [15], [3], [1], [14] as mentioned in [30] and [31]. Their work further support the ten attributes which was proposed by Graham in his first book, "Security Analysis" in 1934 [7], to screen undervalued equities. As according to [30], Graham published a list of ten

attributes of an undervalued stock, which could be used by investors seeking excess return. These ten attributes were: -

- Earnings-to-price yield double the AAA bond yield
- P/E four-tenths highest average P/E in most recent 5 years
- Dividend yield two-thirds the AAA bond yield
- Price two-thirds tangible book value per share
- Price two thirds net current asset value
- Total debt less than tangible book value
- Current ratio greater than or equal to 2
- Total debt less than or equal to net quick liquidation value
- Earnings doubled in most recent 10 years
- No more than two declines in earnings of 5 percent or more in the past 10 years

It was noted that few companies could meet all 10 criteria. Besides Graham's ten attributes, Aby further developed another four fundamental rules for equities screening [1], namely: single valued P/E (P/E < 10), market price < book value, established track record of return on shareholder equity (ROE > 12%) and dividends paid out less than 25% of earnings. It was found that when the four criteria are used to screen stocks, quality investments seem to. As such, [30] chooses the combination of Graham's rules and Aby's rules to form the attributes of soft-computing models to identify high potential equities, which are: -

Attributes	**Description**
P/E ratio	Price to earning ratio
BVPS	Book value per share
ROE	Return on equities
DPR	Dividend payout ratio
DY	Dividend yield (annual dividend per share over price per share)
PBR	Price to book ratio
CA	Total current assets
GD	Gross debt
ANS	Weighted average number of shares
CR	Current ratio (current asset over current liability)
EPS	Earning per share

Table 2: Identified features for Soft-computing models

As research work has been done on using the above identified attributes for screening undervalued equities, these eleven attributes will be used for our soft computing models.

The above completes the attributes selection for our soft computing models.

3.2 Soft-computing Models

Figure 9 presents a survey of soft computing approaches which are under studied in our dissertation. The three criteria of considerations are:

1. the representational topology
2. the structure learning approaches (e.g. clustering approaches)
3. the parameter learning approaches

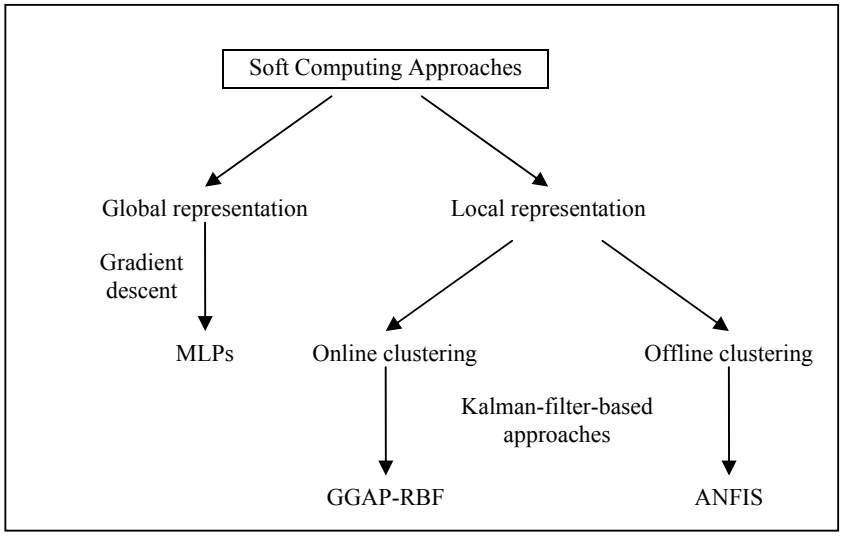

Figure 9: Taxonomy of Soft Computing Approaches under studied

The representational topology defines how the input-output relationships of the soft computing approaches are represented. The global representation describes that the output relationship is represented by a globally optimized function of the input-spaces, in this manner, the input-spaces are usually not pre-clustered. The examples of these are MLPs or any kind of higher-order polynomial function that mapped the input-spaces to an output-space. On the other hand, the local representation pre-clustered

the input spaces into different locally subspaces, and the parameter identification or learning to map the input-output relationship is performed in the local region. Examples of these are the RBF and fuzzy logic systems.

The advantage of global representation is that these approaches are less prone to the curse-of-dimensionality because usually the space complexities are pre-defined. However, these methods are easily suffered from trapping in local minimum region that affects the accuracy of learning. In contrast, the local representation usually involves some clustering mechanism, and dependant on the effectiveness of clustering, sometimes the space complexities are increased exponentially and easily become intractable. However, the advantages of the local representation approaches are, they are less susceptible to trapping in local minima region because the region has been limited to smaller subspaces. Furthermore, it is also possible to interpret the results with respect to these local subspaces.

There are two classes of clustering approaches, which are the online and offline clustering approaches. The online clustering method gradually adds on the nodes while training, therefore it is suitable for systems that are constantly changing. However, the problem is also harder to analyze, and the addition or deletion of a cluster is usually problem-dependent. Carelessly assigning cluster to a system might result in catastrophic results, for example, the system may run out of memory during computation or take forever of training. On the other hand, offline clustering presumes that the system under study is static, or semi-static, which means that the systems are not constantly changing. With such, the system can choose some samples of data for pre-clustering during the structure learning process, and once that is done, the structure of the network will not be going on for further changes; that is to say, no new nodes will be added. The advantage of this is obvious; the computational time and space complexities are easier to manage. The disadvantage of offline clustering is, if the system underwent drastic changes over sometime, this approach might be myopic in learning. When this happens, the designer will have to manually restart the clustering process and retrain the system.

Parameter learning can be divided into online and offline learning (e.g. least-squares method) as well. In our dissertation, we only consider the online learning. Gradient-

descent methods are usually used in MLPs and those which have a global representation. This may be slow and take a number of training epochs before it converges, however this is simple and robust. Kalman-filter-based approaches [17], or usually called recursive-least-squares methods, are usually used in approaches with local representation. It is a fact that they converge within few training iterations. However, the computational complexities per training epoch are much higher than gradient-descent and is prone to the initial conditions and noisy data.

In this dissertation, we will consider the following soft-computing models: -

1. Multi-level Perceptrons (MLP) feed-forward neural network, which is global representation with gradient-descent with momentum learning
2. General Growing and Pruning Radial Basis Function (GGAP-RBF), which is local representation model with online clustering with an extended Kalman filter-based parameter learning approach [11]
3. Adaptive Neuro-Fuzzy Inference Systems (ANFIS) model, which is a fuzzy system with offline clustering (subtractive clustering) with the online Kalman filter-based approach [17]

In short, MLPs and RBFs networks are the two most commonly-used types of feedforward network. For the method in which hidden units combine values coming from preceding layers in the network: MLPs use inner products, while RBFs use Euclidean distance [20]. As explained, RBFs do not scale well with the dimensionality of the inputs features but have more local structure than the MLPs such that outliers and other "issues" in one region of the input will not affect the others [16]**.** ANFIS differ from the MLPs and RBFs fundamentally as ANFIS use the fuzzy if-then rules to map a given inputs to an output. Strictly speaking, ANFIS is a sub-area of Fuzzy Logic instead of Neural Networks. The main advantage is that it is possible to extract information about the rules used to make decisions.

We shall demonstrate the properties of these approaches based on the training complexities and results of the equities selection problem in subsequent chapters.

3.3 Introduction of Confusion Matrix

In learning of extremely imbalanced data, the classifier that predicts every case as the majority class can earn very high level of accuracy, which is not desirable. Thus, the overall classification accuracy is often not an appropriate measure for the performance.

Confusion Matrix is introduced and used in [10]. A confusion matrix contains information about actual and predicted classification done by a classification system, a shown as the following table: -

	Predicted "Class 1"	**Predicted "Class 2"**
Actual "Class 1"/(+1)	A	b
Actual "Class 2"/(-1)	C	d

Table 3: Confusion Matrix

Table 3 contains information about the number of actual and predicted instances which classified by the classification system, which are the three soft-computing models in our context. The entries of Table 3 have the following meaning in our context of study: -

- a: the number of correct predictions that the instances are "Class 1"
- b: the number of incorrect predictions that the instances are "Class 1"
- c: the number of incorrect predictions that the instances are "Class 2"
- d: the number of correct predictions that the instances are "Class 2"

Several performance metrics can be derived from Table 6 as shown as the following: -

- True Positive rate (TP or Acc^+): the proportion of "Class 1" cases that were correctly identified, as calculated as: $TP = \frac{a}{a+b}$
- False Negative rate (FN): the proportion of "Class 1" cases that were incorrectly classified as "Class 2", as calculated as: $FN = \frac{b}{a+b}$

- False Positive rate (FP): the proportion of "Class 2" cases that were incorrectly classified as "Class 1", as calculated as: $FP = \frac{c}{c+d}$
- True Negative rate (TN or Acc^-): the proportion of "Class 2 " cases that were correctly identified, as calculated as: $TN = \frac{d}{c+d}$

Several more useful performance metrics are derived from the confusion matrix, such as accuracy, precision, G-mean, etc. which are shown as the following: -

- Traditional Accuracy, Acc is calculated as $\frac{a+d}{a+b+c+d}$
- Precision, p is calculated as $\frac{a}{a+c}$
- Recall, r is calculated as $\frac{a}{a+b} = Acc^+$
- Geometric mean, G-mean is calculated as $\sqrt{Acc^+ \times Acc^-}$

[18] pointed out that geometric mean – G-mean of accuracies measured can be applied on imbalanced data to have better performance measurement of their problems. [26] discussed the relation of Acc^+ and Acc^- as some problems can increase the accuracy of Acc^+ at the cost of the accuracy on the negative samples, Acc^-. The relation of the two quantities can be captured by what is called the ROC (Relative Operating Characteristics) curve, which will be discussed in detail at the latter part of this study as we are going to apply this on our experiments.

Chapter 4: Experiments Results

4.1 Preparation

The simulation set-up involves data collection, data processing and networks training. The data is collected through Bloomberg system. MATLAB neural-network toolbox and fuzzy-logic toolbox are used for simulation. The simulation environment is: Intel Pentium M, 1.50 GHz, and 760 MB of RAM with running OS – Microsoft Windows XP Professional 2002, SP2.

4.1.1 Data Collection

The market under study is DJIA (Dow Jones Industrial Average). There are a total of 1630 equities have been extracted with a period of ten years, from 1995 to 2004. We include all equities, which include stocks that have been de-listed in order to avoid bias. For the required features but not available, the value of zero will be assigned. We will remove those entries that have missed more than half of the required features, in order to reduce the possible noise to the benchmark.

This set of data forms the sample of our soft computing models. In neural network methodology, the samples are often subdivided into "training", "validation", and "test" sets [20]. Refer to the [8], the definitions are given as such: -

- Training set: A set of examples used for learning that is to fit the parameters (e.g. weights) of the classifier.
- Validation set: A set of examples used to tune the parameters (e.g. architecture, not weights) of a classifier, for example to choose the number of hidden units in neural network.
- Test set: A set of examples used only to assess the performance (generalization) of a fully-specified classifier.

Why do we want to separate the sample into three distinctly separated set? Basically we want the trained networks to be able to perform on new data, that is to generalize,

with such the training set can not be used again to test the generalization performance in order to avoid bias.

We will have two settings for our problem, which are shown in the following tables: -

Setting	Training set	Validation set	Test set
1	1995 – 2002 (8 years)	N/A	2003 – 2004 (2 years)
2	1995 – 2002 (8 years)	2003 (1 year)	2004 (1 year)

Table 4: Setting 1 and Setting 2 of the data arrangement

Setting 1 will be used for Experiment I, which is for the comparisons of the accuracies and appreciation among the three soft computing models. We follows [30] 80:20 rules, which is using the first 80% (eight years) of the data set to predict the known results for the last 20% (two years) of the data set. Setting 2 will be used for Experiment II, which is for picking the most valuable equities by choosing the best cut-off-point for the soft computing models as well as by choosing the most valuable equities based on the significant of the output values. That is the reason of having validation set here, which is to choose the best cut-off-point, such that the appreciation of the selected equities can be maximized.

4.1.2 Data Processing

The classification problem of equities selection is defined as the following. "Class 1" is defined as any stock which appreciates in share price in value equal or more than 80% within one year, otherwise is classified as "Class 2". This is in-line with [30] and [31] with the exception that we are using 80% cut-of-point instead of 100% to separate the data set into two classes. The reasons are that "Class 1" data can be increased by almost 50% if we lower the cut-of-point to 80%, and it is also highly desirable even if we have 80% of share price appreciation instead of 100%. The nature of collected data set is imbalanced. Next, we will be discussing on our methods to handle imbalanced data.

Imbalance data essentially means at least one of the classes constitutes only a very small minority of the data [9]. And, the interest usually leans towards correct classification of the "rare" class (which we will refer to as the "Class 1" in our

context). Refer to [9], there are two common approaches to handle imbalanced data. One is by assigning high cost to misclassification of the minority class and trying to minimize the overall cost, which is called, cost sensitive learning. Another is to use a sampling technique, which is either down-sampling the majority class or over-sampling the minority class, or both. Down-sampling means reducing the size of the samples and over-sampling means blow up the samples by self-copying. We will choose the latter as down-sampling may result in loss of information.

As discussed, there are in total 1,630 securities extracted from DJIA with the period of 1995 to 2004. To be specific, every row contains eleven attributes and a known class, which is either "Class 1" or "Class 2". The information of the "class" forms the output of our soft computing models. The training set (eight years) consists of 10,243 input rows. Out of these 10,243 inputs, there are 9,224 inputs are classified as "Class 2" and only contain 1,019 "Class 1" inputs. This is an imbalanced data; "Class 2" dominates the data set but our interest is on identifying the minority class – "Class 1". Over-sampling technique is applied on "Class 1"; it blows up "Class 1" from 1,019 input rows to 9,224 input rows and the data now is balanced by having half as "Class 1" and half as "Class 2. It is not necessary to apply over-sampling technique on validation set and test set, the soft-computing models is only trained with training set but not the rest.

The following table summarizes the designs of the processed data.

Setting	**Training set**	**Validation set**	**Test set**
1	1995 – 2002 (8 years) Original: 10,243 inputs Over-sampling to:18,448 inputs	N/A	2003 – 2004 (2 years) Original: 2,422 inputs No over-sampling is done
2	1995 – 2002 (8 years) Original: 10,243 inputs Over-sampling to:18,448 inputs	2003 (1 year) Original: 1,448 inputs No over-sampling is done	2004 (1 year) Original: 974 inputs No over-sampling is done

Table 5: Summary on Setting 1 and Setting 2

4.1.3 Networks Training

The soft computing models under study are Multi-level Perceptrons (MLP) feed-forward neural network, Adaptive Neuro-Fuzzy Inference Systems (ANFIS) and General Growing and Pruning Radial Basis Function (GGAP-RBF).

4.1.3.1 Multi-level Perceptrons (MLP) feed-forward neural network

MLP is configured with the number of hidden neurons being two-times of the input layer, which is twenty-two neurons (eleven attributes times two). The training algorithm is gradient descent with momentum and adaptive learning rate. Both the hidden layer neurons and output layers neurons have tangent sigmoid activation functions, which have the output values between -1 and +1. This MLP configuration can be visualized with Matlab Neural Network Graphical User Interface (GUI), as shown in Figure 10. Please refer to *Chapter 2.1.1.1* for the syntax and semantics of this MLP structure.

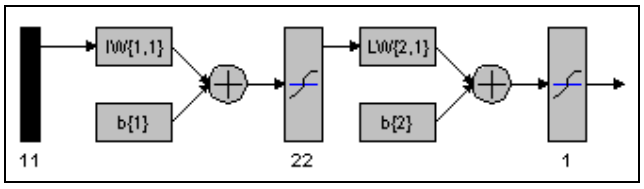

Figure 10: MLP structure

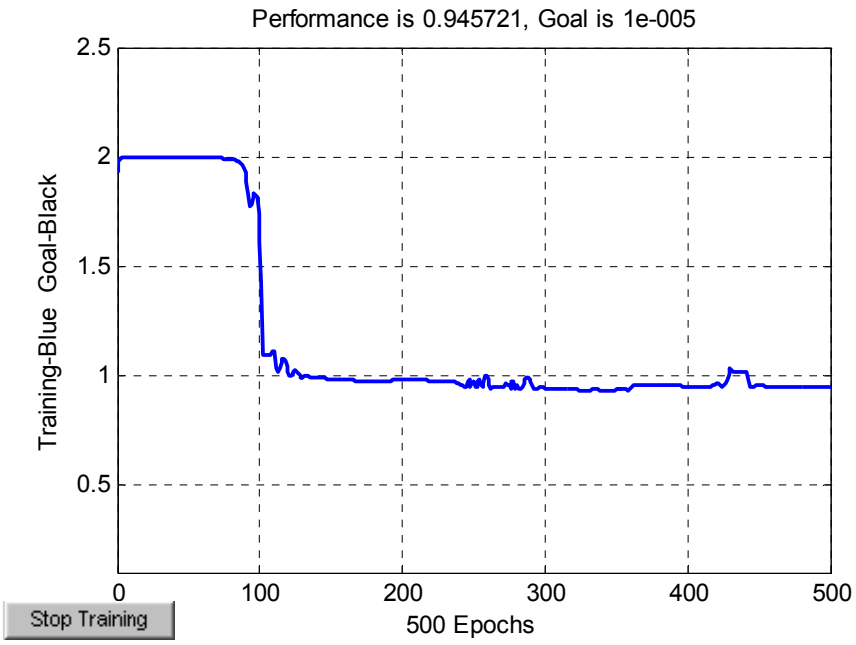

Figure 11: MLP Training

Figure 11 shows the MLP training mean squared error (MSE) against training epochs. The initial MSE is about 2 and it is stabilized at MSE of 0.945721. According to Figure 11, the actual epochs required to reach the possible minimal training error is actually less than 150. The CPU computational time for training with MLP with the above configuration is 188.45 seconds[1]. Therefore, the chosen maximum training epochs for MLP are 500.

[1] Simulation environment: CPU: 1.5GHz; Memory: 768MB; OS: Windows XP.

4.1.3.2 Adaptive Neuro-Fuzzy Inference Systems (ANFIS)

ANFIS is configured using subtractive clustering with a radius of 0.20. And, it is trained for 10 epochs. The trained ANFIS model has two rule nodes, each nodes is represented as a locally-defined linear functions. The CPU computational time for training with ANFIS with the above configuration is 396.85 seconds[2].

We can visualize the ANFIS structure with Matlab Anfis Editor *anfisedit*. We present the ANFIS structure in Figure 12, the developed rules and the membership function after training is shown in Figure 13 and Figure 14 respectively.

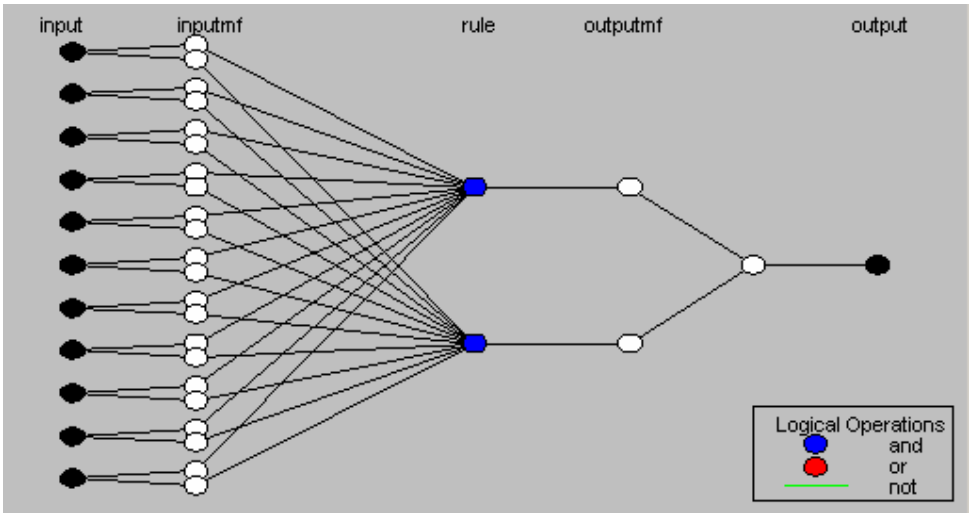

Figure 12: ANFIS structure

Figure 12 presents the ANFIS structure. The inputs comprise eleven attributes that are chosen in *Chapter 3.1*. Please refer to *Chapter 2.1.2* for the syntax and semantics of ANFIS structure.

We have two rules for the ANFIS system. The rules govern the fuzzy inference process from inputs to outputs, as explained in Section *2.1.2 Introduction of Fuzzy Logic.*

[2] Simulation environment: CPU: 1.5GHz; Memory: 768MB; OS: Windows XP.

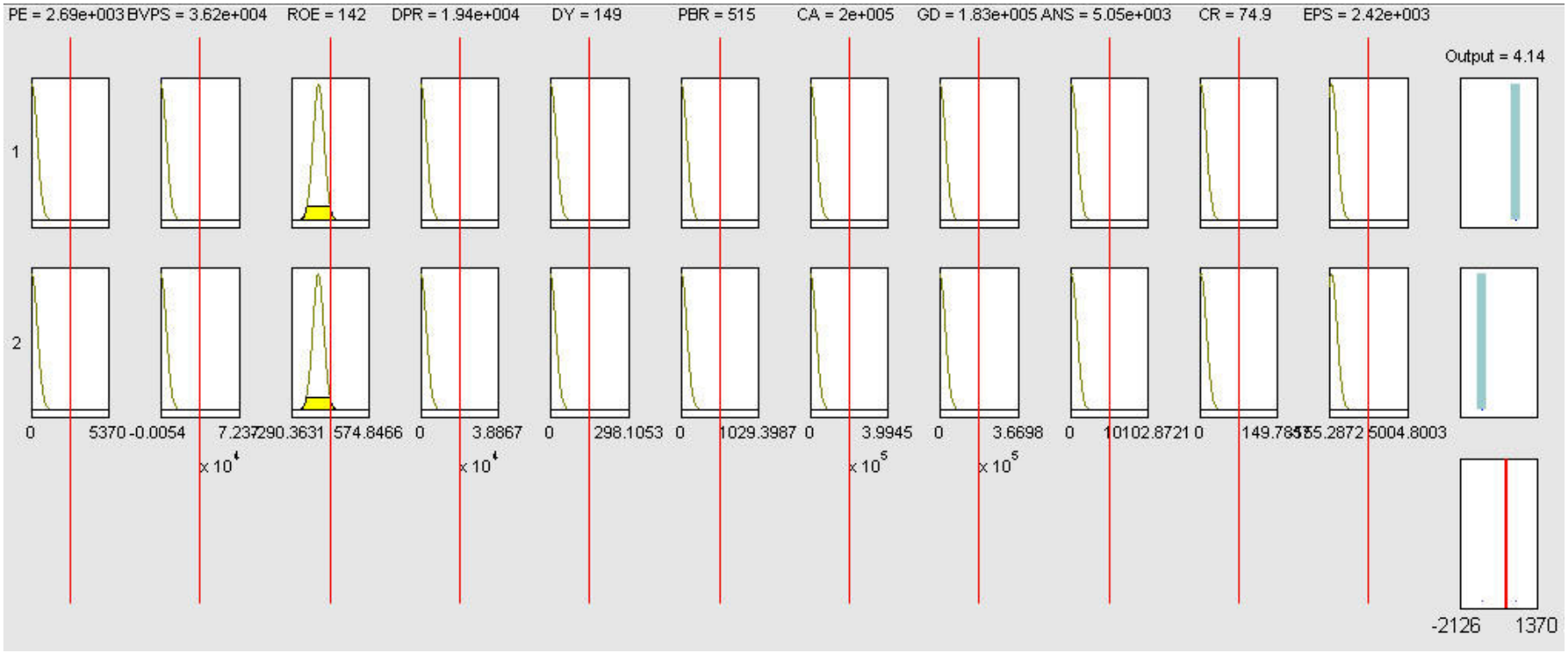

Figure 13: The two rules of ANFIS system, for the fuzzy inference process from inputs to outputs. Each row of plots corresponds to one rule and each column of plots corresponds to either an input variable (yellow) or an output variable (blue).

After 10 epochs of training, the final ANFIS which has the minimum checking error will be chosen. The new membership functions for the eleven attributes are shown as below: -

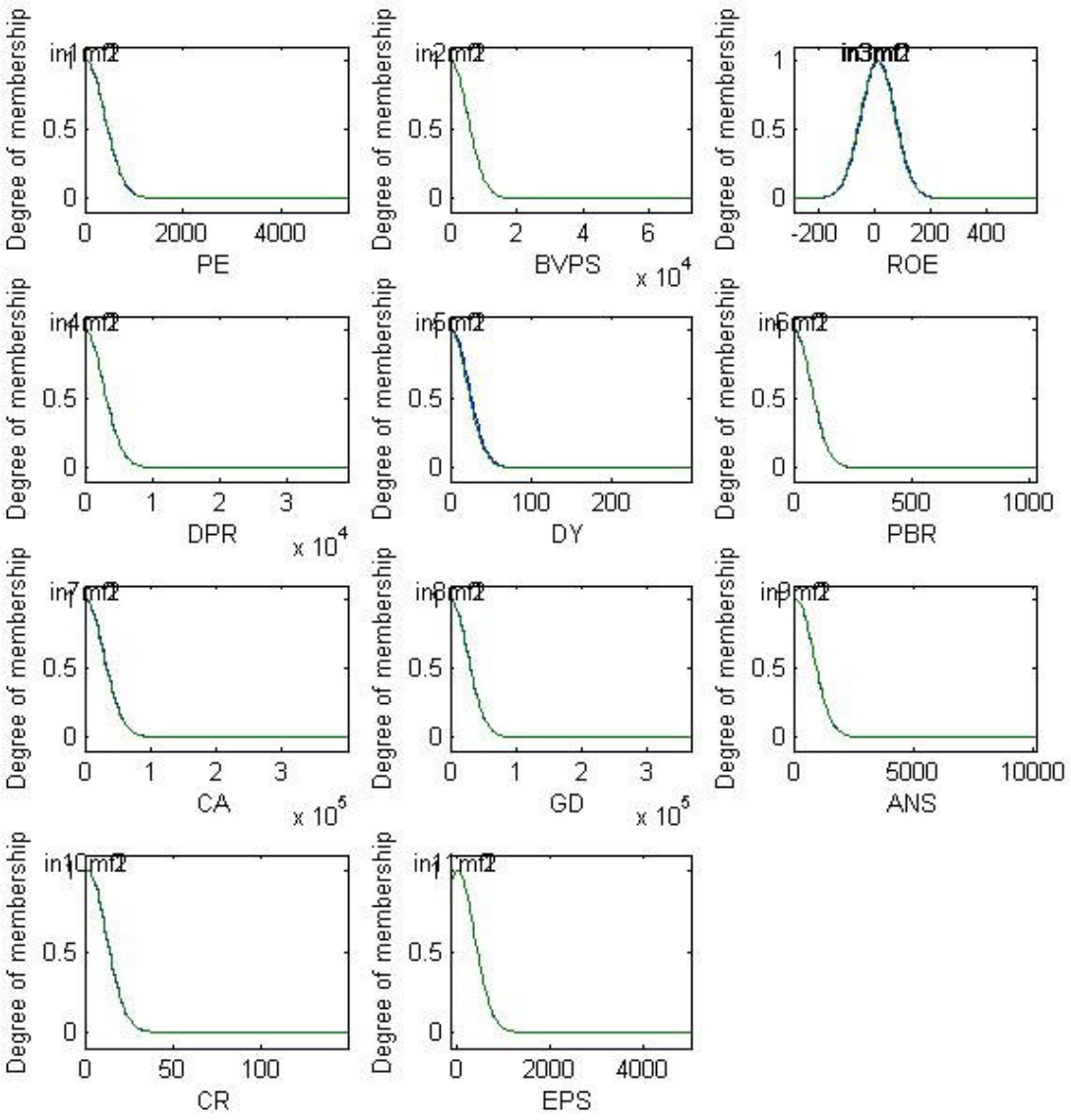

Figure 14: Membership functions for ANFIS attributes after training.
(Horizontal-axis shows the degree of membership)

The root-mean squared error for the training data has been shown as below: -

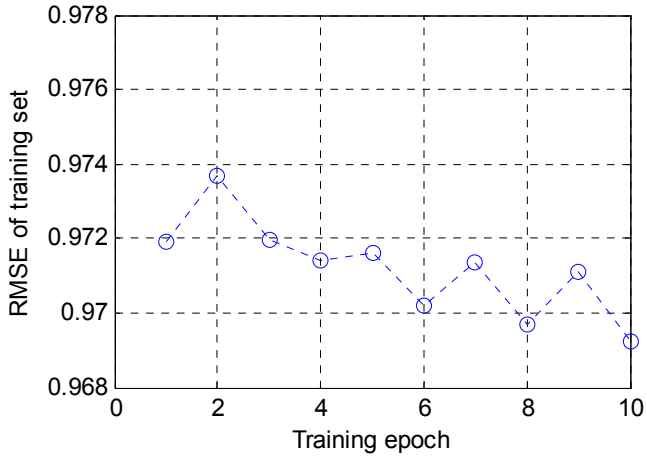

Figure 15: RMSE of ANFIS training

The oscillations after epoch 4 are due to the heuristic rules to increase and reduce step size in ANFIS training algorithm [17].

The training algorithms among these three models, namely MLP, ANFIS and GGAP-RBF, are different in nature. The training error representations between Figure 11 and Figure 15 have different meaning that can not be directly comparable with each others. As an example, the training of ANFIS is affected by the increment and decrement of step size in Figure 15; and the training of MLP is affected by the momentum setting. Figure 11 and Figure 15 provides general idea on how training error curves are. Furthermore, the performance of MLP, ANFIS, and GGAP-RBF are compared from various perspectives in the subsequent sections. Therefore, the training curves comparison for these three models is not a necessity.

4.1.3.3 General Growing and Pruning Radial Basis Function (GGAP-RBF)

GGAP-RBF which has been proposed by [11] in 2005, will be used to study our problem. We applied the provided MATLAB source codes by [11] for training. The CPU computational time for training with GGAP-RBF is 360.7 minutes (21,642

seconds)[3]. The time complexity for GGAP-RBF is obviously too high as compared to MLP and ANFIS, which spent 188.45 seconds and 396.85[4] seconds respectively.

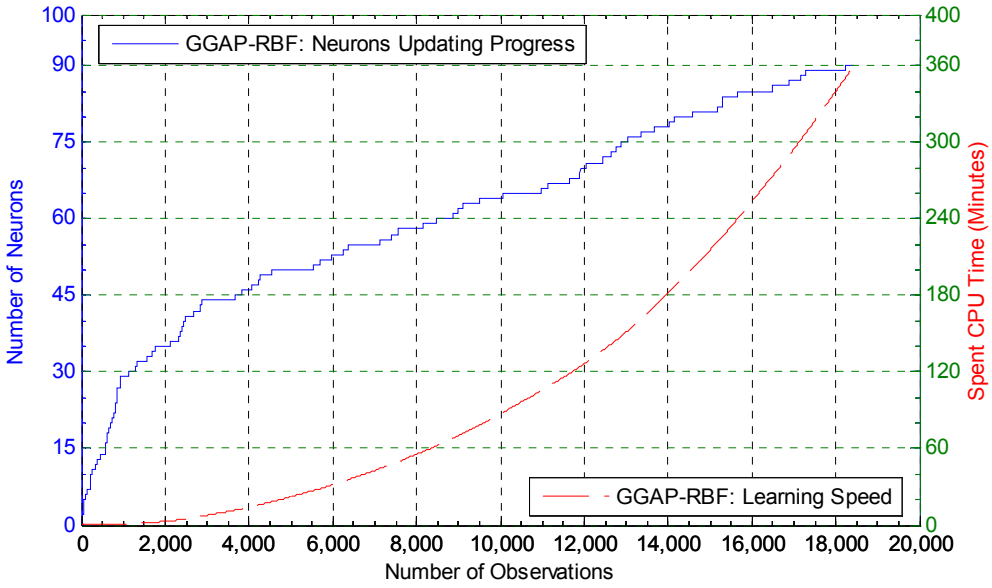

Figure 16: Learning speed and neuron updating progress for GGAP-RBF

To understand this, we plot the learning speed and neuron updating progress against the number of inputs as shown as the figure above. For this problem, the [11] proposed GGAP-RBF [11] obtains a total of 90 neurons after the six-hour training. The time complexity is exponential. The more the neurons are added, the slower the algorithm work. As you can see, the first-half of the training set, which comprises about 9,224 input rows, occupied the first hour of the total training CPU time. On the other hand, the second-half of the training set occupied about five times more than first-half of the training data on the spent CPU time on training. This shows that GGAP-RBF does not scale well with the numbers of inputs although there are usually large numbers of instances for financial problems.

The above is summarized as below: -

Soft-computing models	Computational Time (for training)	Descriptions
MLP	188.45 seconds	Training algorithm: Gradient descent with momentum;

[3] Simulation environment: CPU: 1.5GHz; Memory: 768MB; OS: Windows XP.
[4] Simulation environment: CPU: 1.5GHz; Memory: 768MB; OS: Windows XP.

		Hidden layers: 22 neurons; 500 epochs
ANFIS	396.85 seconds	Subtractive clustering[5]; 10 epochs
GGAP-RBF	21,642.00 seconds	After training: 90 neurons have been added.

Table 6: Comparisons of Computational Time

4.2 Experiment I: Performance Validation

After detail explanations of the three soft-computing models under studied, the results of their training will be presented, subsequently followed by discussion on how it is related to equities appreciation. Essentially, out-of-samples data (with known classification outputs) will be applied as new set of inputs to the trained networks and the predicted outputs of the trained networks will be compared to the known desire outputs to compare the performance in accuracy or in appreciation among the three soft-computing models. As mentioned earlier, data Setting 1 will be applied for Experiment I.

Setting	Training set	Test set
1	1995 – 2002 (8 years) Original: 10,243 inputs Over-sampling to:18,448 inputs	2003 – 2004 (2 years) Original: 2,422 inputs No over-sampling is done

Table 7: Summary of Setting 1 for Experiment I

4.2.1 Results in Accuracy

Training results and Testing results will be presented in the following sections.

4.2.1.1 Training Results

As discussed, we have in total of 18,448 input rows (also known as samples, instances or observations) which comprise 9,224 of "Class 1" data and 9,224 of "Class 2" data, after over-sampling technique has been applied on "Class 1". This forms the input data for the soft-computing models. We can apply the same input data, which we used for training, to the trained models (also known as networks), to obtain the recall rate.

[5] Subtractive Clustering is a fast, one-pass algorithm for estimating the number of clusters and the cluster centers in a set of data. [19]

Recall is the process of putting input data into a trained network and receiving the output; subsequently compare the output with the desire output.

We plotted the 18,448 training input instances with regards of its predicted output for the three soft computing models under studied, as shown as Figure 17, Figure 18 and Figure 19. Let us look at each of them in details.

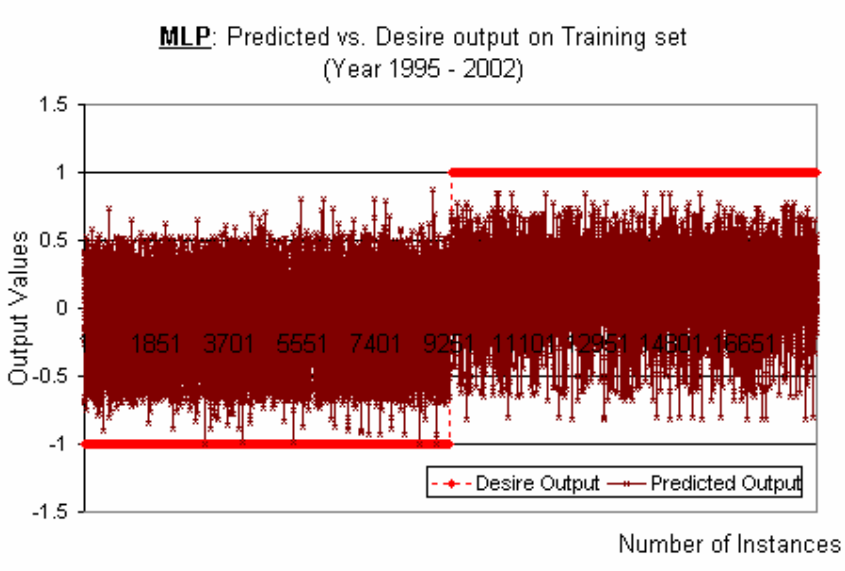

Figure 17: MLP: Training results (training set for the year 1995-2002)

Refer to Figure 17, the desire outputs of total 18,448 training input instances have been sorted in order to form the step function-liked pattern. The desire outputs value is either "Class 1" or "Class 2", which is numerically represented as -1 and +1 respectively. And, there are half of the "Class 1" data (as shown as +1) and half of the "Class 2" data (as shown as -1). That means the data is sorted in such a way that the "Class 1" instances are grouped to form step-up (+1) pattern and "Class 2" instances are grouped to form step-down (-1) pattern. We re-apply the training input instances to the trained networks, and obtain the predicted outputs. The predicted outputs' values are fuzzy and the range of the values are dependent of the soft-computing models, for example the output values of our MLP is in the range between -1 and +1, as the transfer function being used is tangent sigmoid, which has minimum output value of -1 and maximum of +1.

We expect that the predicted output values should tend to have higher values for instances which have desire outputs of +1 (right-hand-side of the step-function-liked the plot), and vice-versa. The training algorithms can be concluded to be not useful if there is no such relationship.

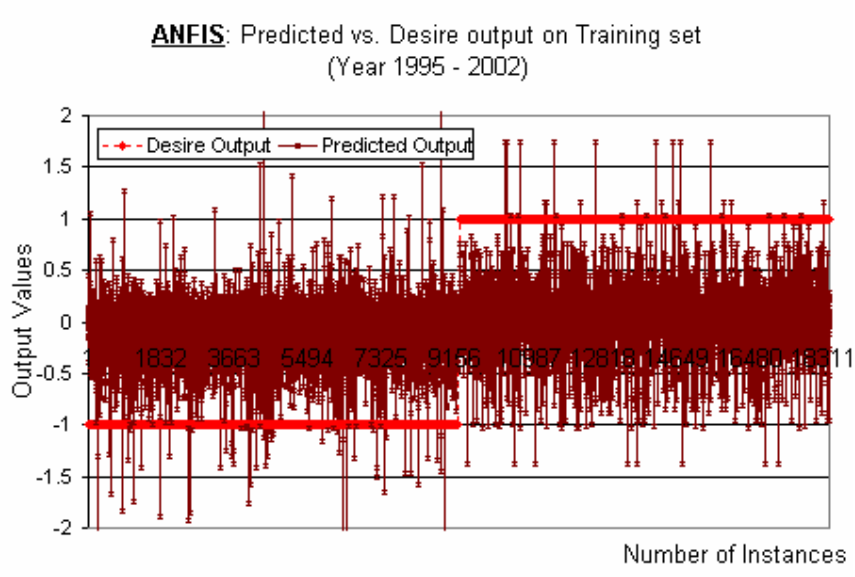

Figure 18: ANFIS: Training results (training set for the year 1995-2002)

The outputs of ANFIS are by linear function and the output values may be out of the range of -1 and +1, as shown in Figure 18. There is observable slightly upward trend for the predicted output values from the range between -0.6 and +0.25 to the range between -0.5 and +0.5. If we only consider the predicted output values which are above +0.5 (use your hand to cover at the +0.5 y-axis horizontally), the majority of the points are on the right-side of the step function, which reflects that majority of the predicted output having values equal or greater than +0.5, are having desired output of "Class 1"(+1). This shows that ANFIS has meaningful output for applying its trained network on training set, similarly to MLP.

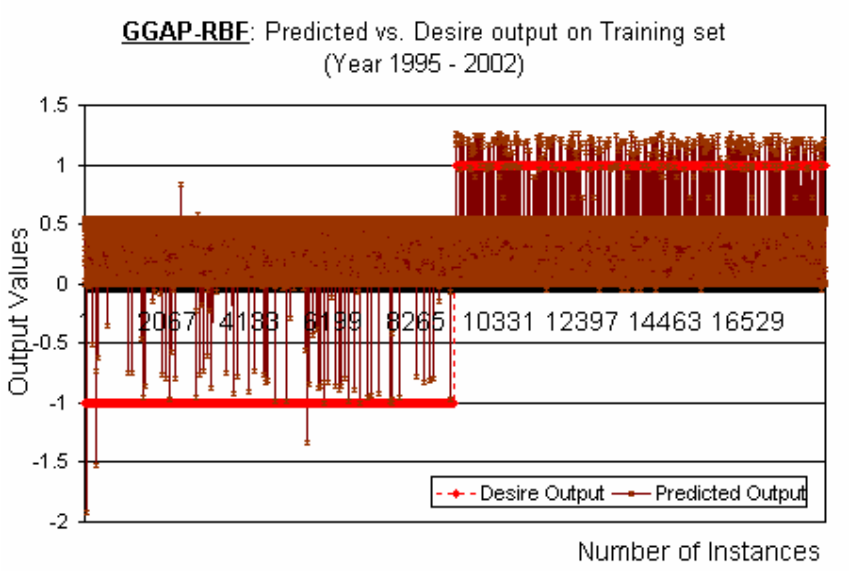

Figure 19: GGAP-RBF: Training results (training set for the year 1995-2002)

GGAP-RBF performs differently to MLP and ANFIS. Let's look at the left-side of the step function, which contains all the "Class 2" instances. Though majority of the instances having predicted output values in the range of 0 and +0.5, but there is almost zero instances having predicted output values greater than +0.5. Similar pattern can be found for all "Class 1" instances as well (right-side of the step function). What does this imply? Let us define the rules as such: First rule, for all predicted output values equal or greater than +0.6, mark them as "Class 1". Second rule, for all predicted output values equal or lower than -0.1, mark them as "Class 2". And, simple ignore those instances with predicted output values in between +0.6 and -0.1. As you can see from Figure 19, majority of the instances have predicted output values in the range of +0.6 and -0.1, and thus will be ignored. However, the identified "Class 1" and "Class 2" instances, based on the defined two rules, are having almost 100% of accuracy. This shows that GGAP-RBF has the most desirable outcomes for recall rate.

Refer to Figure 17, Figure 18 and Figure 19, we see that there is such positive relationship between the predicted outputs and desire outputs for all three models. We see that there is a shift up-trend for the predicted outputs from "Class 2" to "Class 1" for all three models. This shows that the chances to accurately classify the data are more than 50%. (The probability to correctly flip a coin is 50-50, hence we can only

conclude the existence forecasting ability only if the chances for success is more than 50%).

4.2.1.2 Test Results

Good recall rate is meaningless as the models are trained and tested with the same set of instances which results in high level of bias. What we are interested in is not recall rate, but test rate. Test rate applies out-of-sample data as the inputs to the trained network and compare the output to their known desire outputs. Out-of-sample data refers to the samples which are not used in training the models.

The test set of Setting 1 comprises last two years of data (total of 2,422 input rows). Here, the test set will be applied to form the inputs of the trained models. The same data arrangement as the previous section is applied, as shown as the Figure 20, Figure 21 and Figure 22.

The test set is an imbalanced data. The "Class 1" (+1) is minority data which is only at about 10% of the total test set. Thus, the step-liked function has a late step-up.

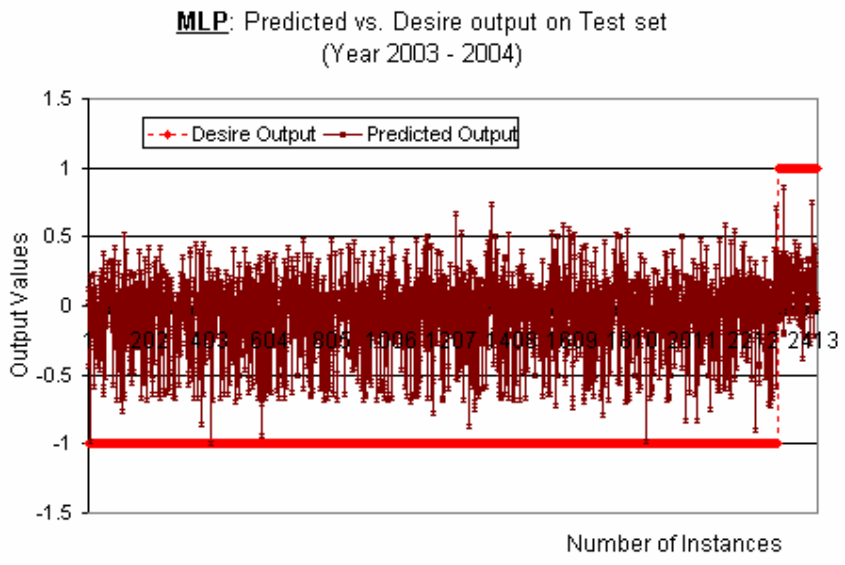

Figure 20: MLP: Test results (test set for the year 2003-2004)

Refer to the Figure 20, we see that for MLP model, there is slightly positive correlation between the predicted output values with the desire output values. Majority of the "Class 1" instances have predicted output values of equal or greater than zero. By observation, similar pattern can be identified in ANFIS model as well, as shown as the next figure. In the "Class 1" instances (which is on the right-hand side of the step-function that contains group of +1 desire output values), there is a chunk of predicted output, which have values fall into the range of 0 and +0.5. This chunk of data is slightly having higher values than the predicted output of those "Class 2" instances, and shows that there is positive correlation between predicted outputs and desire outputs.

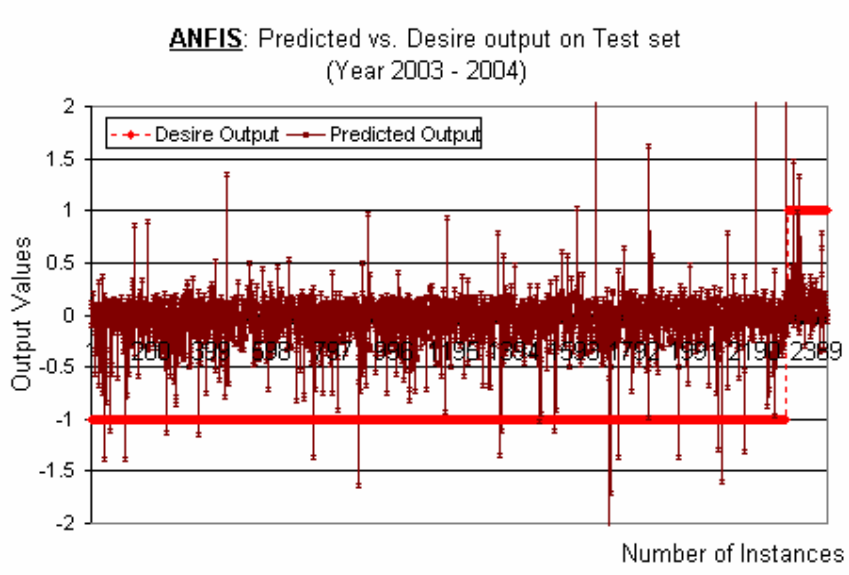

Figure 21: ANFIS: Test results (test set for the year 2003-2004)

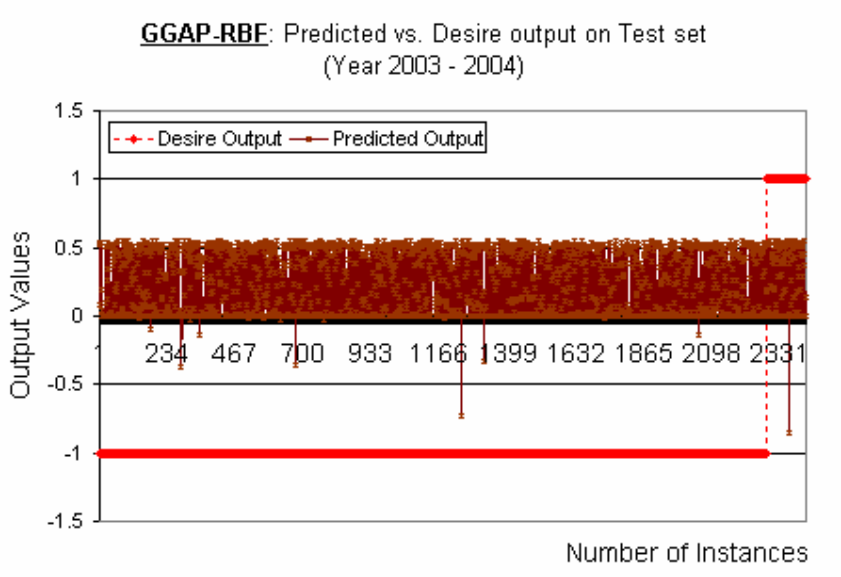

Figure 22: GGAP-RBF: Test results (test set for the year 2003-2004)

The test results for GGAP-RBF is different than MLP and ANFIS, as shown in the Figure 22. Similar to its training results, majority of the predicted output values fall into the range of 0 and +0.5. There are only a few instances which have values larger or equal to +0.6 and smaller or equal to -0.1. There is no distinct cut off values based on the predicted output to classify the training set. This shows that the GGAP-RBF has no ability on prediction in training set! It is no better than tossing a coin.

After studying the results on training set (recall) and test set (generalize), we found out that MLP and ANFIS performs similar in terms of accuracy of predictions while GGAP-RBF performs excellently for recalling its trained models with training set but extremely poor for generalization. The reasons may be due to one or more of the mentioned challenges of applying soft computing on financial problems: noisy and non-deterministic nature of the data.

If we simply apply 0 as cut-off-point for the output values to classify the equities into "Class 1" or "Class 2" for MLP and ANFIS, and we also simply apply 0.25 as cut-off-point for GGAP-RBF, then we can show the summary of the accuracies results as below: -

Accuracies	Recall Rate (Training set)	Generalize Rate (Test set)
MLP	62.787%	51.363%
ANFIS	62.538%	51.775%
GGAP-RBF	54.51%	57.345%

Table 8: Summary of Accuracies Results (Recall & Generalize rate)

(Choose intuitively: MLP and ANFIS: Cut-off-point of Zero; GGAP-RBF: Cut-off-point of +0.25)

We calculated the accuracies of both training set and test set, as shown as Table 8. Accuracy is defined as the number of correctly predicted equities over total number of predictions. As the test set is highly imbalanced, the generalize rate is highly bias to "Class 2". Thus, the accuracies shown on test set are not reflecting the reality, e.g. having all predicted outputs values as "Class 2" will give almost 90% of accuracies. Training set has been over-sampled such that "Class 1" and "Class 2" data in training are equal. As a result, we will not be interested in the Generalize Rate but focusing on Recall Rate in Table 8. The Recall Rates of MLP and ANFIS are comparable while GGAP-RBF gives the lowest Recall Rate.

4.2.2 Results in Appreciation

The major concern for equities picking problems is to maximize the returns of the picked equities. The accuracy analysis discussed is not enough as accuracy comparison is not the main concern here although they are inter-related, that is to say, higher accuracy may or may not directly related to higher returns in equities share price appreciation. For example, accuracy analysis does not tell that whether higher predicted output values imply higher appreciation and vise-versa. Hence, we will present the analysis of the predicted output values of the soft-computing models against the known equities next-year share price appreciation (in terms of percentage, %), on both the training set and test set.

4.2.2.1 Training Results

The analysis of the predicted output values of the soft-computing models against the known equities next-year share price appreciation (in terms of percentage, %), on the training set as shown as the following: -

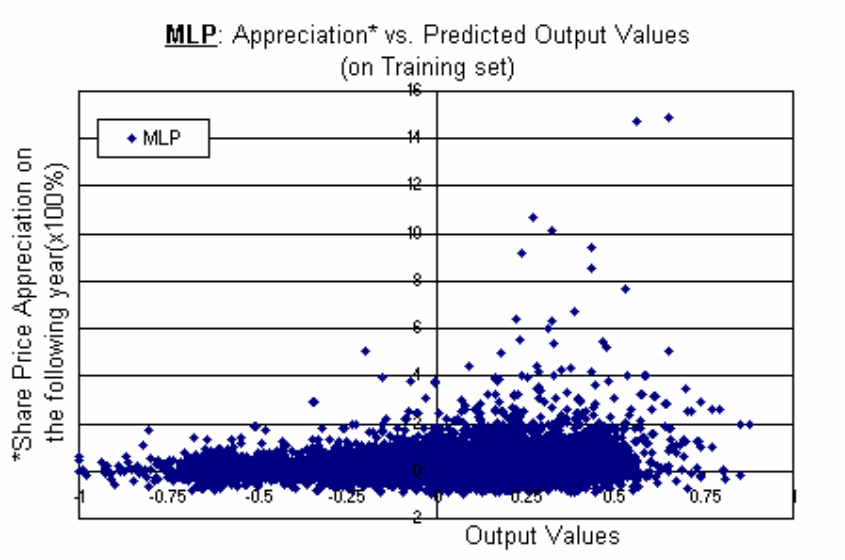

Figure 23: MLP: Actual appreciation vs. NN prediction (training set for the year 1995-2002)

As mentioned earlier, MLP trained network has predicted output values (neural network output values) in the range between -1 and +1, hence the x-axis is in the range of -1 and +1. The plot, as shown as Figure 23, skews towards +1, which reflects the good positive correlation with appreciation of equities value in the following year. We hereby expect that higher predicted output values shall reflect higher appreciation returns.

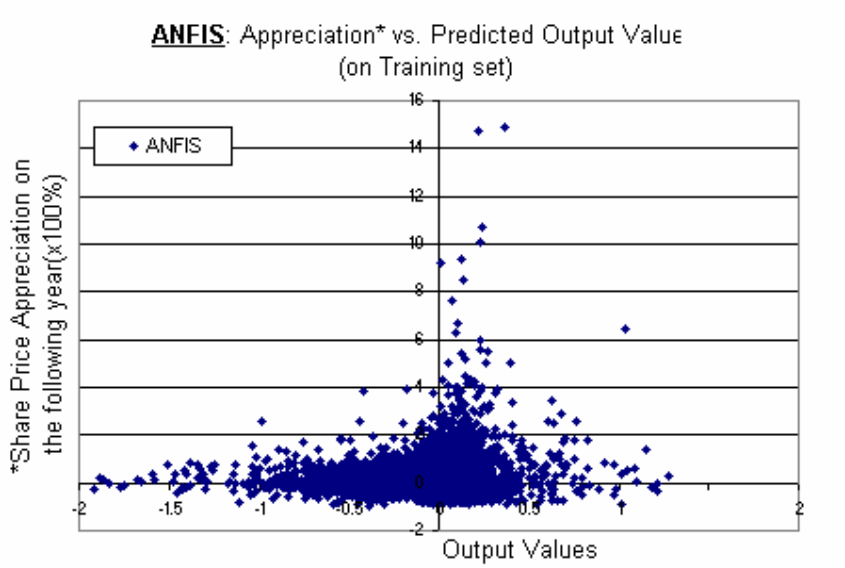

Figure 24: ANFIS: Actual appreciation vs. NN prediction (training set for the year 1995-2002)

Similarly, ANFIS model also show good positive correlation with the appreciation of the equities price in the following year. In contrast to MLP and ANFIS models, GGAP-RBF model gives a different scatter chart. It does not form a clear skewing curve, but we see some correlation relationship between the equities appreciation and the predicted output values around the x-axis values of -1 and +1 respectively, as shown as Figure 25.

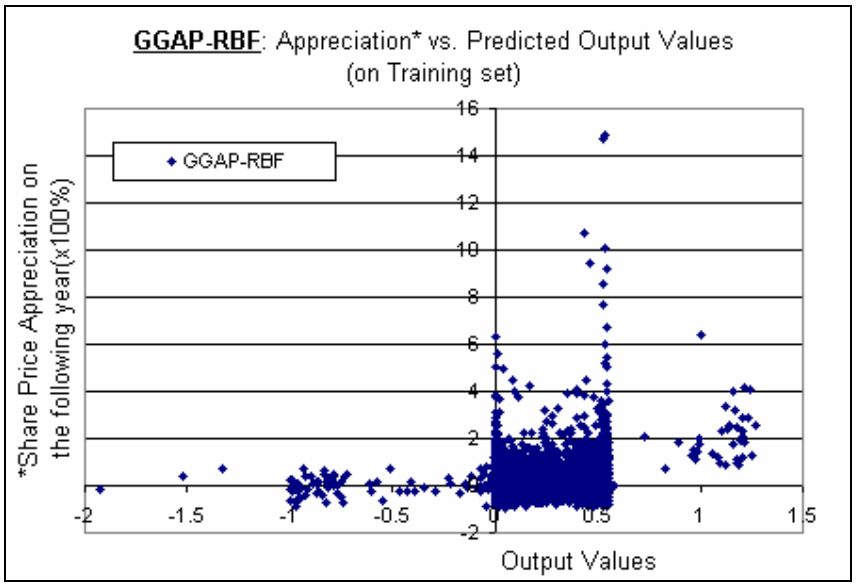

Figure 25: GGAP-RBF: Actual appreciation vs. NN prediction (training set for the year 1995-2002)

The explanation of the results presented in this section will be illustrated with numerical analysis in *Chapter 4.2.2.3*.

4.2.2.2 Test Results

By observation, similar pattern of correlation can be identified in test set as well for MLP and ANFIS models, as shown as Figure 26 and Figure 27. However, it appears that GGAP-RBF again has the lowest correlation or close to +0 correlations among the appreciation (Y-axis) and the predicted output values (X-axis), as shown as Figure 28.

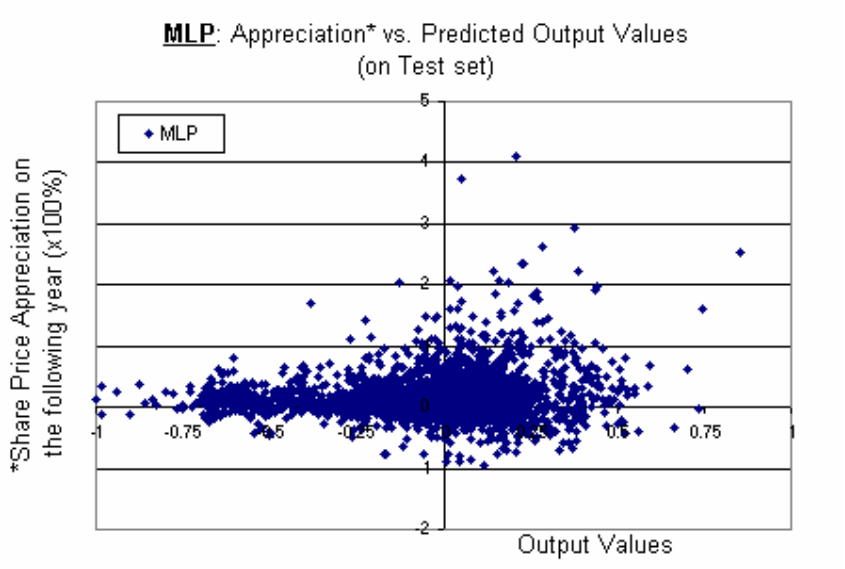

Figure 26: MLP: Actual appreciation vs. NN prediction (test set for the year 2003-2004)

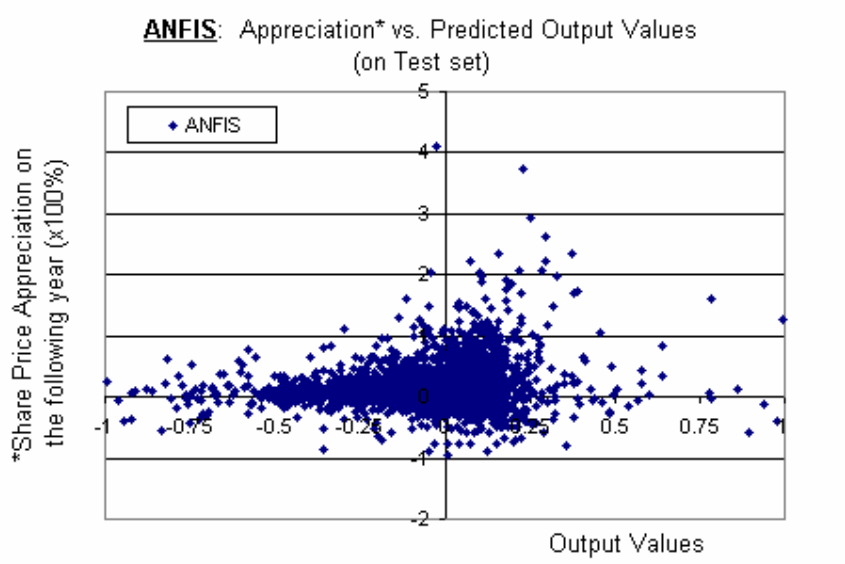

Figure 27: ANFIS: Actual appreciation vs. NN prediction (test set for the year 2003-2004)

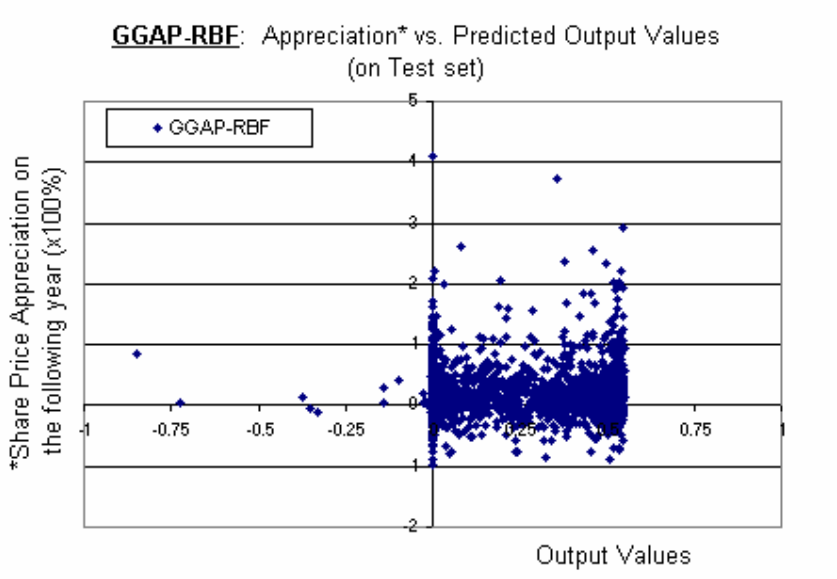

Figure 28: GGAP-RBF: Actual appreciation vs. NN prediction (test set for the year 2003-2004)

The explanation of the results presented in this section will be illustrated with numerical analysis in *Chapter 4.2.2.3*.

4.2.2.3 Results: Correlation

A quantitative approach to analyze the correlation between appreciation and predicted output values is using the Pearson correlation coefficient, which is defined in the following equation: -

$$r = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2 \sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

where: r, the correlation coefficient may take any value between -1.0 and +1.0. The larger the value is, the higher the correlation is between two data vectors.

The correlation between the percentage of appreciation of the equities share price and the predicted output values, has been shown in the following table: -

Correlation	MLP	ANFIS	GGAP-RBF
Training set	0.312327151	0.231391027	0.184214955
Test set	0.129631143	0.152538366	0.056896817

Table 9: Correlation (Actual appreciation vs. NN prediction)

It is expected that GGAP-RBF has the lowest correlation between its predicted output values and the appreciation of the equities share price and it has close to +0, which is +0.0569 of such correlation on test set. Again this demonstrates that GGAP-RBF performs poorly for equities prediction problems, which belong to class of stochastic prediction problems.

The correlation values are all pretty low for all three models. However, it is meaningful for the comparison of the three models. A more useful mechanism to pick valuable equities will be approached in *Chapter 4.3*.

4.3 Experiment II: Proposed Approaches for Equities Picking

We have discussed the experiments results on the accuracies of predictions and the correlation analysis of the appreciation of the picked equities by using data arrangement – Setting 1. The accuracies, however, does not lead to money. The main interest is to pick the most valuable equities to be invested in such that we can receive high return for next-year equities appreciation.

In this section, two approaches for equities picking will be proposed and discussed. Firstly, Relative Operating Characteristics (ROC) curve [25] will be applied to identify the cut-off-point systematically for the three trained soft-computing models to filter out the invaluable equities. (ROC curve will be explained in *Chapter 4.3.2*) Secondly, the top ten equities, which have greater neural network predicted output values, are chosen from each model and simply pick these equities.

Now, with the same soft-computing models which has been developed and trained earlier, we are going to apply data arrangement – Setting 2's validation set and test set, as shown as Table 10, for configuring the trained network and test respectively.

Setting	Training set	Validation set	Test set
2	1995 – 2002 (8 years) Original: 10,243 inputs Over-sampling to:18,448 inputs	2003 (1 year) Original: 1,448 inputs No over-sampling is done	2004 (1 year) Original: 974 inputs No over-sampling is done

Table 10: Summary of Setting 2 for Experiment II[6]

4.3.1 Networks Configuration

As discussed earlier, the predicted output values of trained soft-computing models is positively correlated with the appreciation of the equities in one year period. How do we make decisions on which equities to be signaled for trading? That is to say, how do we set a cut-off-point whereby all equities with predicted output value equal or greater than the cut-off-point are to be classified as "Class 1", otherwise "Class 2"?

[6] Sampling technique is discussed in Section *4.1.2 Data Processing*.

The output values of neural network models fall into the range of -1 to +1 (as explained in *Chapter 4.2*), therefore we systematically apply all possible cut-off-points from -1 to +1 in a delta step of 0.01 to classify the prediction into either "Class 1" or "Class 2". Subsequently, we compute the average appreciation for the equities which have been predicted as "Class 1" (or, "winner"). The results of these experiments are presented in Figure 29, Figure 30 and Figure 31. For all the three figures, x-axis represents the cut-of-point. Y-axis of upper plot represents the average appreciation of the predicted "Class 1" equities whereas the y-axis of bottom plot represents the total number of equities has been predicted as "Class 1" with the respective cut-off-point. For example, if we have ten equities are picked or predicted as "Class 1" (or, "winner") equities with the cut-off-point of zero, we will compute the average appreciation of those picked equities' by summing all the ten equities' actual next-year appreciation and averaging them by ten. This average appreciation is shown in upper plot. The highlighted points in the figures are further explained in subsequent paragraphs.

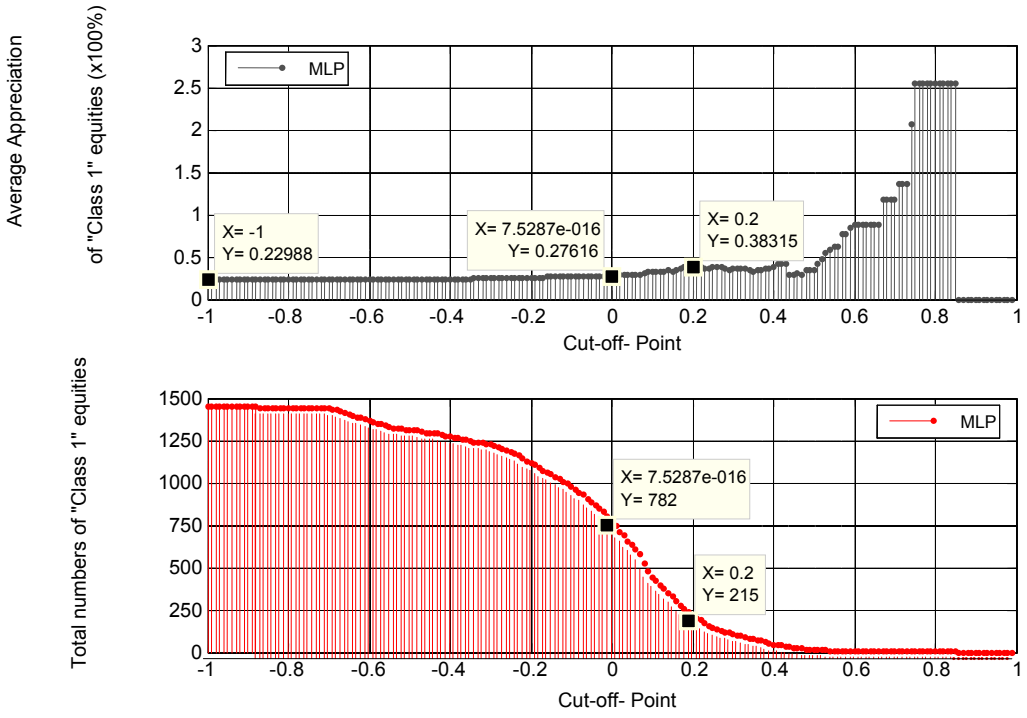

Figure 29: MLP: Appreciation vs. Cut-off-point (validation set for the year 2004)
(Average appreciation is calculated from selected "Class 1" equities)

Figure 29 demonstrates the impact of chosen cut-off-point over the average appreciation return from the picked equities. This experiment is done on validation set with trained MLP model. Let's pick the cut-off-point of zero. At the zero cut-off-point, there are in total of 782 equities have been signaled as "Class 1" equities. And the average appreciation of 782 equities is 27.616%, which is higher than the average appreciation of all the equities (22.99%). If 0.2 cut-off-point has been chosen, the total number of equities being classified as "Class 1" are further reduced to 215 from 782. The average appreciation of the 215 equities has been improved to 38.31% from 27.61%, which is much higher than the average appreciation of all the equities, 22.99%.

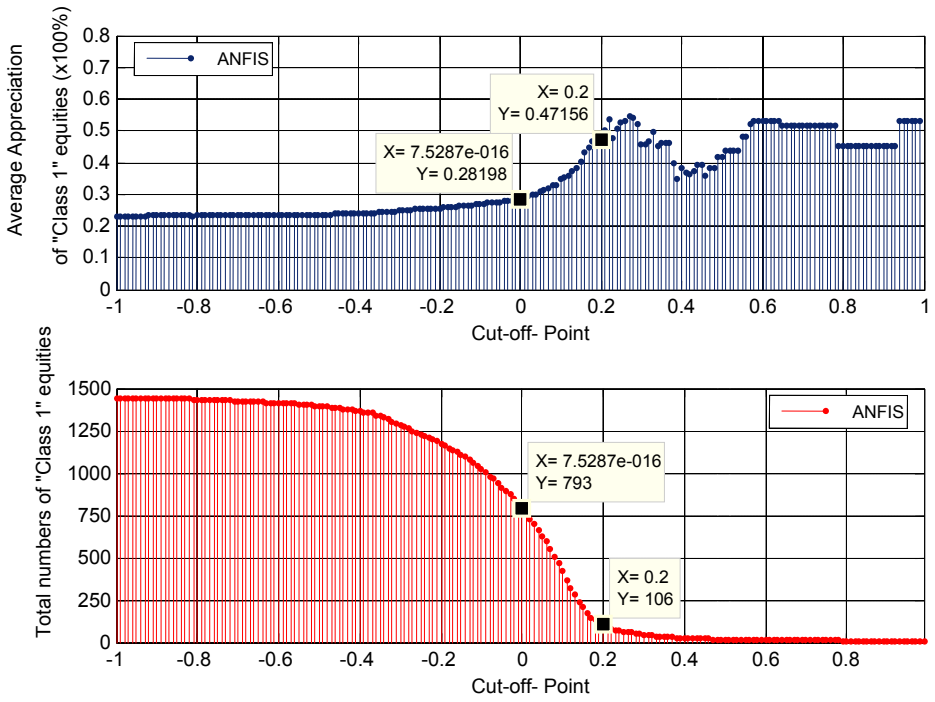

Figure 30: ANFIS: Appreciation vs. Cut-off-point (validation set for the year 2004)
(Average appreciation is calculated from selected "Class 1" equities)

Figure 30 demonstrates the impact of chosen cut-off-point over the average appreciation return from the picked equities. This experiment is done on validation set with trained ANFIS model. As shown as Figure 30, the performance of ANFIS trained model is more desirable than MLP on validation set. At the zero cut-off-point,

there are in total 793 equities that have been classified as “Class 1” equities. And the average appreciation of the 793 equities is 28.198%, which is slightly higher than MLP in both the number of picked equities and the average appreciation. If the cut-off-point has been chosen at 0.2, the total numbers of equities being classified as “Class 1” are further reduced to 106 from 215 as compared to MLP. The average appreciation of the 106 equities has been improved to 47.156% from 38.31% as compared to MLP. Furthermore, there are sudden average appreciation dips at around cut-off-point +0.4 and +0.8, this further supports that the stock picking problem is a stochastic and non-deterministic problem that are highly challenging.

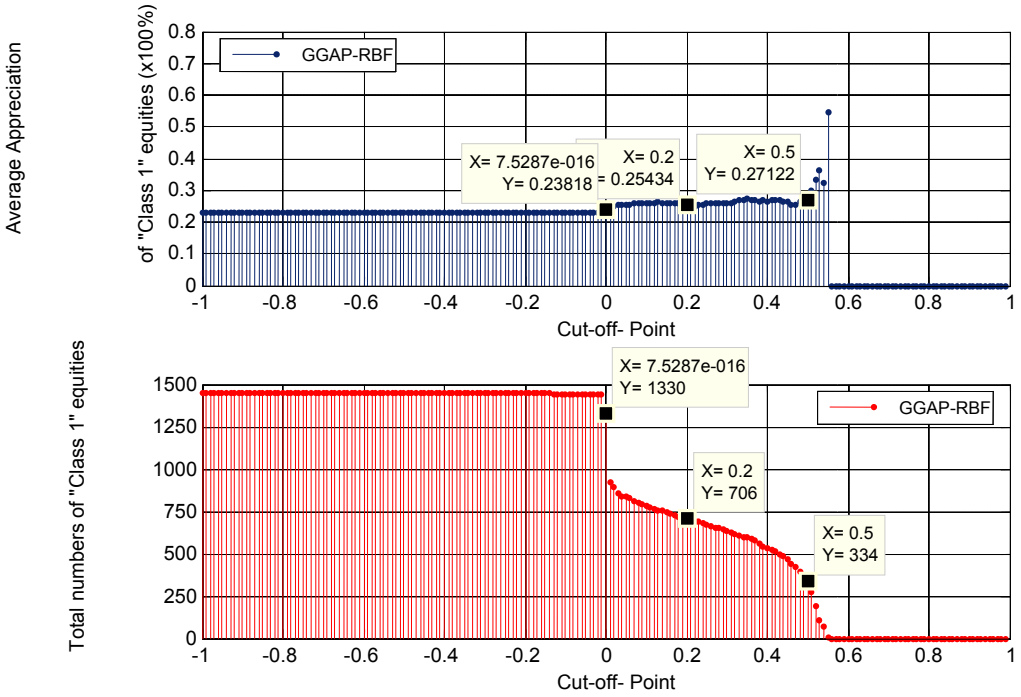

Figure 31: GGAP-RBF: Appreciation vs. Cut-off-point (validation set for the year 2004)
(Average appreciation is calculated from selected “Class 1” equities)

Figure 31 demonstrates the impact of chosen cut-off-point over the average appreciation return from the picked equities. This experiment is done on validation set with trained GGAP-RBF model. As shown as Figure 31, the cut-off-point of zero is meaningless as it signaled almost all the equities, which is 1330 out of 1448, and the average appreciation is about the average of total equities available. At cut-off-point

0.2, the total number of equities signaled has been half reduced to 706 (with average appreciation of 27.12%), which is comparable to cut-off-point zero for MLP and ANFIS. As illustrated in Figure 25, there are none or near-to-zero data that have been clustered in the range of more than +0.6. This explains that there is a dip to zero beyond value of +0.6 in the upper plot of Figure 31. This supports a future research direction of GGAP-RBF to fine-tune the clustering process for handling of similar stochastic data, which is beyond the scope of this research.

Chosen cut-of-point	Neural Network Model	Average Appreciation of the predicted "Class 1" equities	Number of equities predicted as "Class 1"
0	MLP	27.616%	782
0	ANFIS	28.198%	793
0	GGAP-RBF	23.818%	1330
+0.2	MLP	38.315%	215
+0.2	ANFIS	47.156%	106
+0.2	GGAP-RBF	25.434%	706
+0.5	GGAP-RBF	27.122%	334

Table 11: Impacts of the chosen cut-of-points over the average appreciation of predicted "Class 1" (or, "winner") equities

We summarize the findings in Figure 29, Figure 30 and Figure 31 into Table 11. Based on the findings, GGAP-RBF model is best work for higher value of cut-off-point. For example, it has the average appreciation of +27,122% with just 334 equities being picked, at the cut-of-point level of +0.5. On the other hand, both MLP and ANFIS give average appreciation of as high as 38.315% and 47.156% respectively with cut-off-point of +0.2. This concludes that we can choose higher value of cut-off-point for GGAP-RBF and lower value cut-of-points for MLP and ANFIS.

4.3.2 Networks Comparison with ROC curve

As explained in *Chapter 3.3*, True Positive rate is the proportion of "Class 1" cases that were correctly identified. True Negative rate is the proportion of "Class 2" cases that were incorrectly classified as "Class 1". Theoretically, as the value of cut-off-point increases, the rate of True Positive increases as well. The average appreciation of picked equities increases and the total number of signaled equities drops. There is always a trade off between True Positive rate and True Negative rate. To maximize the output performance of the trained soft-computing models, we are interested to maximize the True Positive rate, at the same time, minimize the True Negative rate. This is the methodology of our study to configure the trained soft-computing models.

As discussed in *Chapter 4.3.1*, different cut off values will lead to different group of picked equities and generate different average appreciation. As our problem involves in imbalanced data analysis, we argue that confusion matrix is the best to justify its performance [9]. Based on the confusion matrix, we suggest applying ROC curve. ROC curve has two variables from confusion matrix, which are *True Positive* rate (TP) and *True Negative* rate (TN). We always want to maximize True Positive rate and minimize True Negative rate for optimal performance. As a result, the interceptor point of True Positive rate and True Negative rate will be the best optimal cut-off-point for our problem.

Based on experiment II, we further present the True Positive rate and True Negative rate of the three neural network models. With the same methodology as *Chapter 4.3.1*, we systematically apply all possible cut-off-points from -1 to +1 in a delta step of 0.01 to classify the prediction into either "Class 1" or "Class 2". We then calculate the True Positive rate and True Negative rate by comparing the predicted classification with the actual classification.

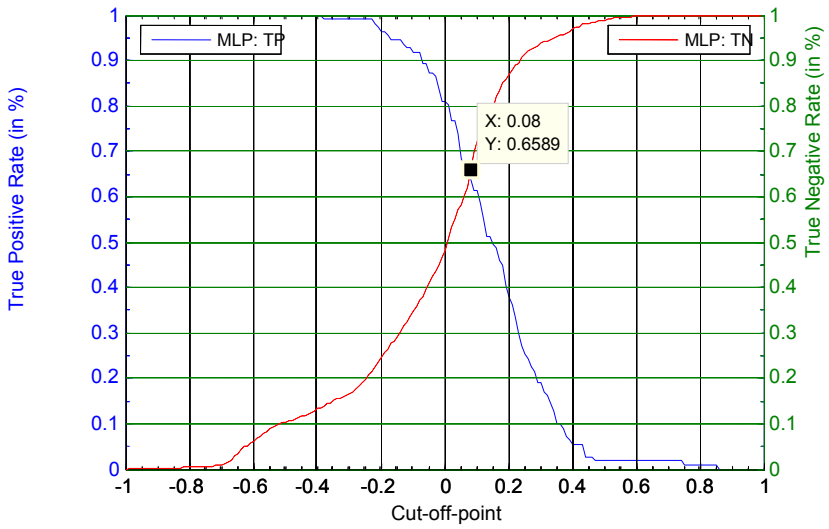

Figure 32: MLP: True Positive Rate vs. True Negative Rate

Figure 32 shows the True Positive rate and True Negative rate of the output of trained MLP on validation set. The horizontal axis shows the chosen cut-off-point for the output values of trained network to predict the data as either “Class 1” or “Class 2”. The two vertical axes show the True Positive rate and True Negative rate respectively. Thus, the cut-off-point at the intersection point of these two curves is the optimal cut off value for best configuration. The best cut off value for MLP is 0.08.

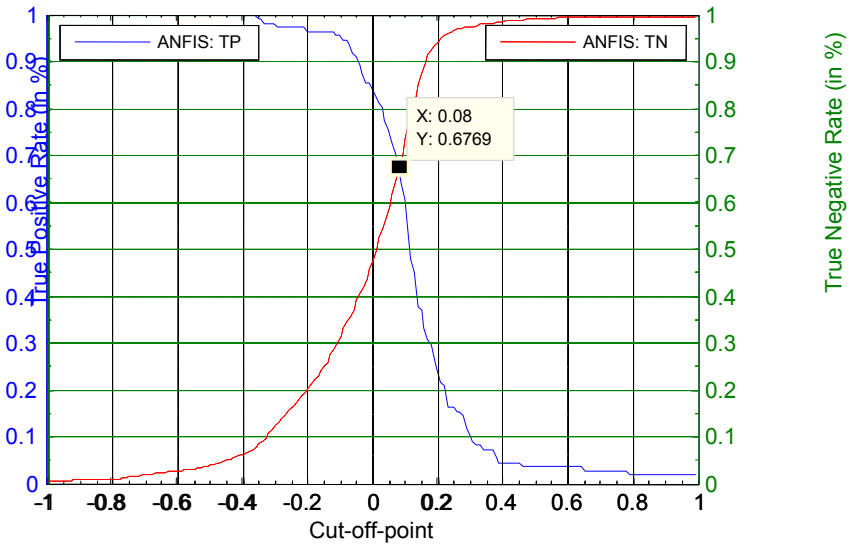

Figure 33: ANFIS: True Positive Rate vs. True Negative Rate

Figure 33 shows the True Positive rate and True Negative rate of the output of trained ANFIS on validation set. Similarly to MLP, the best cut off value for ANFIS is 0.08.

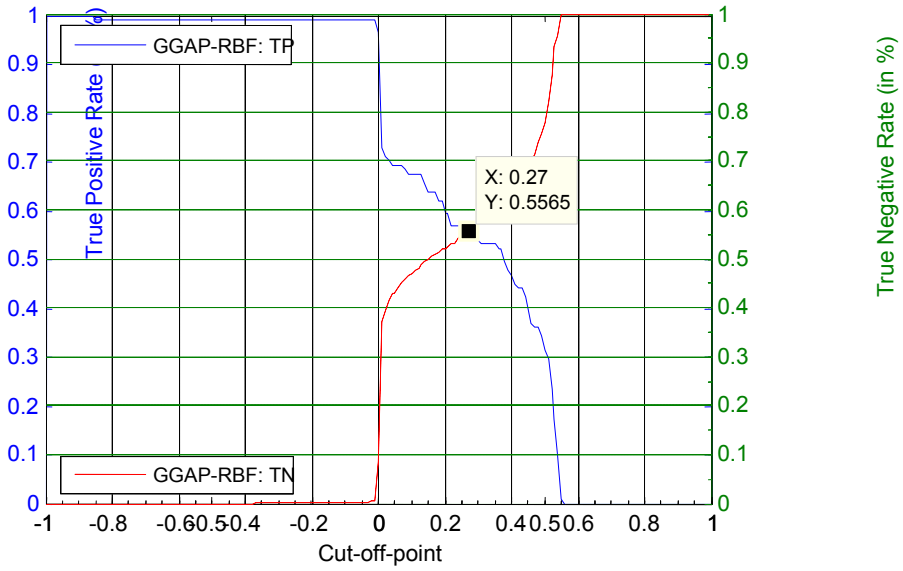

Figure 34: GGAP-RBF: True Positive Rate vs. True Negative Rate

Figure 34 shows the True Positive rate and True Negative rate of the output of trained GGAP-RBF on validation set. The best cut off value for GGAP-RBF is 0.27.

Based on the above findings, we choose the cut off values of +0.08, +0.08 and +0.27 for MLP, ANFIS and GGAP-RBF respectively. And, their numerical results for validation set are shown in Table 13, Table 14 and Table 15.

The results are presented in a customized confusion matrix table format as shown in Table 12. The summary results of each model are presented in the caption of below each table.

TP instances	FN instances	(TP+FN) instances	True Positive Rate	False Negative Rate
FP instances	TN instances	(FP+TN) instances	False Positive Rate	True Negative Rate
(TP+FP) instances	(FN+TN) instances	(TP+FP+FN+TN) instances	**Accuracy Rate***	**Precision Rate***

Table 12: Format of Customized Confusion Matrix
(*Accuracy Rate and Precision Rate is explained in *Chapter 3.3*)

71	40	111	63.964 %	36.036%
456	881	1337	34.106 %	65.894%
527	921	1448	**65.746%**	**13.472%**

Table 13: Customized Confusion Matrix for MLP (Validation set)

*(**Cut- off-point = 0.08**. Total signaled equities = **527**. **Average appreciation = 30.35%.** The average return of All 1,448 Training set equities = **22.99%**)*

76	35	111	68.468%	31.532%
432	905	1337	32.311%	67.689%
508	940	1448	**67.749**	**14.961%**

Table 14: Customized Confusion Matrix for ANFIS (Validation set)

*(**Cut-off-point = 0.08**. Total signaled equities = **508**. **Average appreciation = 32.63%.** The average return of All 1,448 Training set equities = **22.99%**)*

62	49	111	55.856%	44.144%
593	744	1337	44.353%	55.647%
655	793	1448	**55.663%**	**9.4656%**

Table 15: Customized Confusion Matrix for GGAP-RBF (Validation set)

*(**Cut-off-point = 0.27**. Total signaled equities = **655**. **Average appreciation = 25.75%.** The average return of All 1,448 Training set equities = **22.99%**)*

Refer to Table 13, Table 14 and Table 15, all the average appreciation of the models out-perform the average appreciation of the market (22.99%). GGAP-RBF has least accuracies, the most number of equities being selected, a total of 655 equities are picked, and with the least average appreciation (25.75%).

Now, we have configured our trained soft-computing models with the best cut off values to obtain the optimal performance. The cut off values are 0.08, 0.08 and 0.27 for MLP, ANFIS and GGAP-RBF respectively. These settings will be applied on testing set to validate and compare the performance of each model. Before that, let's compare the ROC (Relative Operating Characteristics) curves of three soft-computing models.

The relative operating characteristics (ROC) curve is a highly flexible method for representing the quality of dichotomous, categorical, continuous, and probabilistic forecasts [24]. The ROC is a representation of the skill of a forecast system in which the hit rate and the false-alarm rate are compared [25]. True Positive rate and False Positive rate are explained in *Chapter 3.3*.We illustrate the relationship between True Positive rate and False Positive rate with the following Figure 35: -

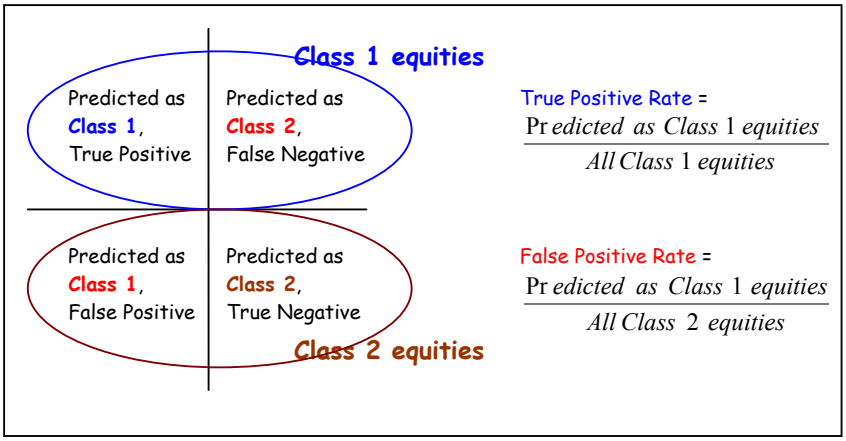

Figure 35: The relationship between True Positive rate and False Positive rate

Equities that have neural network predicted value higher than the cut off value, we classify them as Class 1 (or, "winner"), otherwise Class 2 (or, "loser"). If we apply a very high cut off value, none of the equities will be classified as Class 1 and all equities will be classified as Class 2. Refer to Figure 35, if all the equities are predicted or classified as Class 2 equities, and we predict 0 equities as Class 1 equities, then we will have both True Positive rate and False Positive rate of 0. On the other hand, if we apply a very low cut off value, result in all the equities are classified as Class 1. Then, we will have both True Positive rate and False Positive rate of 1. As we classify more and more equities as Class 1, either True Positive rate or False Positive rate, or both increase. Figure 36 illustrates the resulting True Positive rate and False Positive rate pairs when moving from high cut off values to low cut off values.

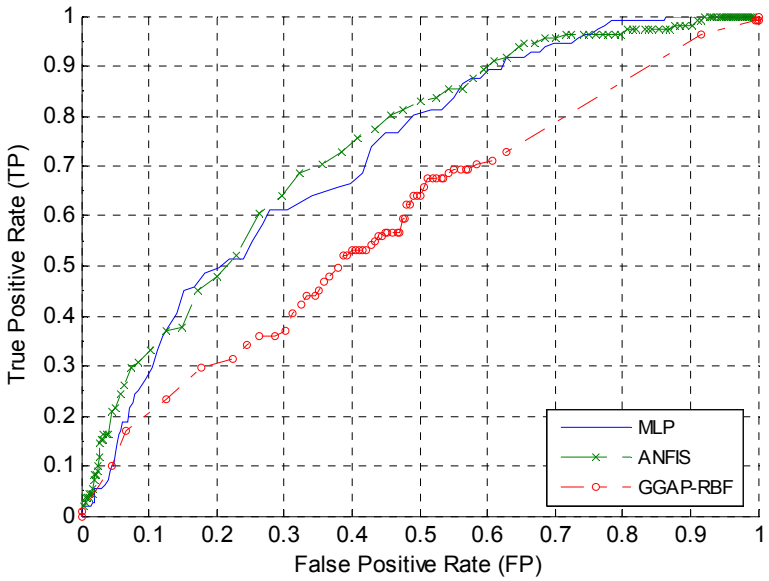

Figure 36: ROC (Relative Operating Characteristics) curves for MLP, ANFIS & GGAP-RBF

In general, for good forecast systems, the ROC curve bends toward the top left, where the True Positive rates are larger than False Positive rates. Where the curve lies close to the diagonal, the forecast system does not provide any useful information. If the curve lies below the line, the system performs negatively for the prediction problem [25].

Figure 36 is ROC curves of three soft-computing models applying on validation set. The horizontal axis of ROC curve measures error rate on "Class 2" instances while the left vertical axis measures accuracy on "Class 1" instances. The curve shows to what extent accuracy on "Class 1" instances drops with reduced error rate on "Class 2" instances. The larger the area below the ROC curve, the higher the classification potential of the algorithm. Obviously, GGAP-RBF has the lowest area below its ROC curve; MLP and ANFIS appear to be comparable.

4.3.3 Test Results

In Setting 2, we have test set of 974 instances, with known output. We will use this set of data as input of the trained soft-computing models, and apply the preconfigured cut off values for the prediction outputs to classify the instances to either "Class 1" or "Class 2". The results are presented in a customized confusion matrix table format as shown in Table 16. The summary results of each model are presented in the caption of below each table.

TP instances	FN instances	(TP+FN) instances	True Positive Rate	False Negative Rate
FP instances	TN instances	(FP+TN) instances	False Positive Rate	True Negative Rate
(TP+FP) instances	(FN+TN) instances	(TP+FP+FN+TN) instances	**Accuracy Rate***	**Precision Rate***

Table 16: Format of Customized Confusion Matrix
(*Accuracy Rate and Precision Rate is explained in *Chapter 3.3*)

12	10	22	54.545%	45.455%
269	683	952	28.256%	71.744%
281	693	974	**71.355%**	**4.2705%**

Table 17: Customized Confusion Matrix for MLP (Test set)
*(**Cut- off-point = 0.08**. Total signaled equities = **281**. **Average appreciation = 13%.** The average return of All 974 Test set equities = **11.22%**)*

11	11	22	50%	50%
234	718	952	24.58%	75.42%
245	729	974	**74.846%**	**4.4898%**

Table 18: Customized Confusion Matrix for ANFIS (Test set)
*(**Cut-off-point = 0.08**. Total signaled equities = **245**. **Average appreciation = 14.93%.** The average return of All 974 Test set equities = **11.22%**)*

9	13	22	40.909%	59.091%
360	592	952	37.815%	62.185%
369	605	974	**61.704%**	**2.439%**

Table 19: Customized Confusion Matrix for GGAP-RBF (Test set)
*(**Cut-off-point = 0.27**. Total signaled equities = **369**. **Average appreciation = 11.15%**. The average return of All 974 Test set equities = **11.22%**)*

Refer to Table 17, Table 18 and Table 19, ANFIS model has the highest Precision Rate and average appreciation of the signaled stocks. On the other hand, GGAP-RBF has demonstrated its low ineffectiveness in picking valuable equities, with the best cut-off-point 0.27. MLP model beats the average return 11.22% by 1.78% whereas ANFIS model beats the average appreciation with 3.71%.

4.3.4 Pick the Best Ten

The above experiment assumes that we have unlimited resources. With such assumption, we can trade as many equities as possible. What if we want to focus on a certain number of equities only, say top 10 equities? We have early demonstrated that there indeed has positive correlation between the outputs of trained models and the appreciation value. Based on this finding, we select the equities based on the strength of the signals from the neural network models. The signals essentially refer to the neural network output values. Figure 37 illustrates the impacts on the average appreciation based on the number of equities that have strongest signals, and are those equities that are being picked.

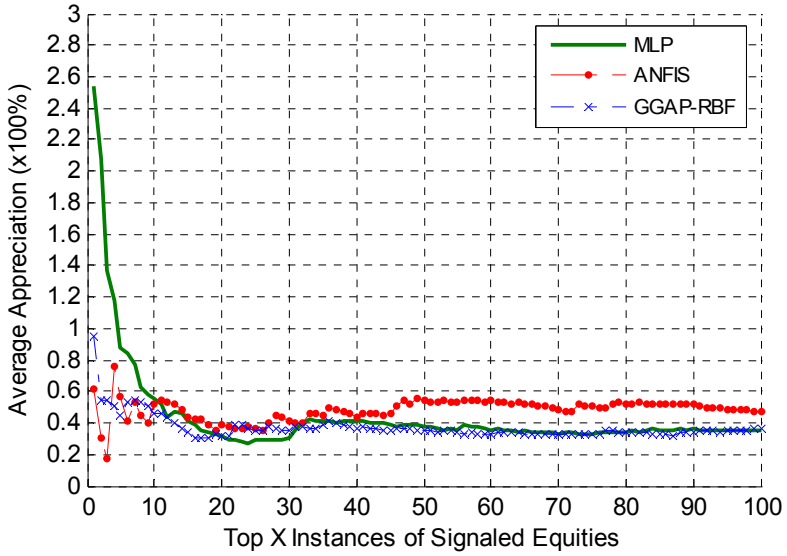

Figure 37: Average Appreciation of selected numbers of Equities with highest output values (validation set for the year 2003) (x-axis represents selected *x* numbers of equities)
(This is based on the strength of predicted output values. The average return of all 1,448 Validation set equities = **22.99%**)

Refer to Figure 37, intuitively we can choose the top 10 of the signaled equities as the average appreciation is about 40% to 60% for all three soft-computing models, which is about doubling the average market appreciation, 22.99%. We apply this on test set subsequently, which is illustrated in Figure 38.

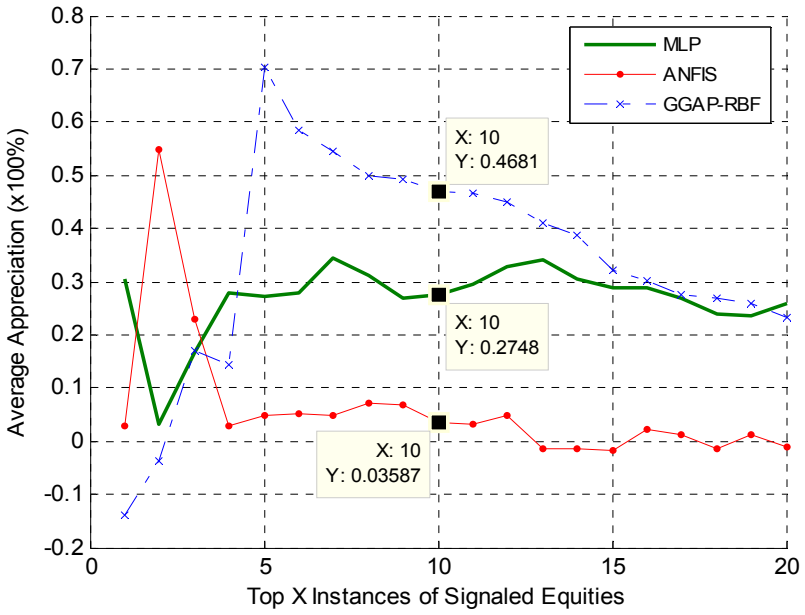

Figure 38: Average Appreciation of selected numbers of Equities with highest output values (test set for the year 2004) (x-axis represents selected *x* numbers of equities)
(This is based on the strength of predicted output values. The average return of all 974 test set equities = **11.22%**)

Let us assume that we want to trade on the top ten equities that have been signaled on trained soft-computing models, out of the 974 instances of test set. As shown in Figure 28, MLP produces 27.48% of appreciation of it top 10 equities; ANFIS produces 3.587% of appreciation and GGAP-RBF produces 46.81% of appreciation in one year period. If we combine the results of all the three models, the average appreciation of the 30 signaled equities will be 25.959%, which is much higher than the average market appreciation of 11.22% by 14.73%.

GGAP-RBF has picked several counters that are giving negative return (or, lost in appreciation). However, GGAP-RBF is capable of giving us the highest appreciation if we only focus on top ten equities. We have an equity which has next-year appreciation as high as 300% in this test set. This special counter is not picked by both MLP and ANFIS in their top tens, but only GGAP-RBF. This results in the good average appreciation of GGAP-RBF. Therefore, we could not conclude that GGAP-RBF is the best in this experiment on test set as this is an outlier case.

Chapter 5: Conclusions and Future Work

5.1 Conclusions

This study shows that GGAP-RBF has huge time complexity as compared to MLP and ANFIS, which are 360.7 minutes as compared to 188.45 seconds and 396.85 seconds [Table 6]. Moreover, GGAP-RBF does not out-perform MLP and ANFIS in Recall Rate, where MLP and ANFIS have accuracies of 62.787% and 62.538% respectively; GGAP-RBF has the Recall Rate of 54.51% [Table 8].

The study shows that there is positive relationship between outputs of the trained networks with the equities appreciation, which may result in earnings of investment. This is materialized as the computation of correlation of the outputs of the trained networks against the equities appreciation. GGAP-RBF has the lowest correlation values for both training set and test set, 0.184 and 0.056 respectively; whereas MLP (Training set: 0.312; Test set: 0.129) and ANFIS (Training set: 0.231; Test set: 0.152) both have comparable correlation values [Table 9].

A systematic equities selection approach based on ROC curve is proposed. The interceptor point of TP (True Positive rate) and TN (True Negative rate) is used for the cut-off-point to obtain optimal equities appreciation. With this configuration, GGAP-RBF has cut-off-point of +0.27, average appreciation of 25.75% and 11.15% for validation set and test set, where validation set has market average appreciation of 22.99% while test set has market average appreciation of 11.22%. MLP and ANFIS both have cut-off-point of +0.08. The average appreciation of validation set for MLP and ANFIS are 30.35% and 32.63% respectively; while the average appreciation of test set are 13% and 14.93% respectively. This shows that they consistently out-perform the average market appreciation.

Based on the studies of correlation, we pick top ten equities with the most significant output values for each soft-computing model. The average appreciation of the top 30 equities picked by all three models is 25.95% on test set, which is significantly higher than market average appreciation.

We conclude that the soft-computing models show relatively positive results on equities picking. The output values of trained models are having meanings on the next-year appreciation of the equities.

5.2 Future Work

This study shows that there is positive correlation between the neural-network prediction and the next-year appreciation of equities. Besides this, the strengths of the prediction values generally are linked to top equities performer. However, this work is only done on DJIA equities, for equities from 1995 to 2004. It is advised to do experiments on more years of data and different markets to study their impacts and whether the results obtained in this dissertation is applicable.

As the objective of this dissertation is to understand, to apply selected soft-computing models on solving the equities picking problems, some theoretical study point to the facts that system with online clustering generally lead to higher time complexity if it is not carefully designed, as in the case of GGAP-RBF. However, we have not meticulously studied why GGAP-RBF performed poorly in this equities picking problem. We can further study the developed rules of ANFIS, and more experiments on different configurations of neural networks.

Features sensitivity analysis can be performed to understand the significance of each feature. Most of the time, we can reduce the eleven features to a lesser numbers. Moreover, Logit regression analysis can be applied in our developed environment to compare the results as Logit approach is still very popular in financial market. We can use Weka [13] software to achieve this.

Lastly, we can further develop a trading system to simulate real-life trading activities based on the work of this dissertation. To do this, we need to include some trading rules, such as transaction cost, limited fund and transaction timing.

Bibliography

[1] Aby Carroll D., Briscoe Nat R., Elliott R. Stephen, Bacadayan A., "Value Stocks: A Look at Benchmark Fundamentals and Company Priorities", Journal of Deferred Compensation, Fall 2001, Vol. 7 Issue 1, p20, 11p)

[2] About Prof. Lotfi A. Zadeh. [Online] http://www.cs.berkeley.edu/~zadeh/acprco.html (current April 8, 2006).

[3] Angelos Kanas, "Neural Network Linear Forecasts for Stock Returns", International Journal of Finance & Economics, Vol. 6, Issue 3, Jul. 2001, pp. 245-254.

[4] Baba, N., Inoue N., Asakawa H., "Utilizing of Neural Networks and GAs for Constructing Reliable Decision Support Systems to Deal Stocks", Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks, IJCNN 2000, 111-116 vol. 5.

[5] Banz, R.W., "The Relationship between Return and Market Value of Common Stocks", Journal of Financial Economics, Vol. 9, Issue 1, pp. 3-18.

[6] Basu S., "Investment Performance of Common Stocks in Relation to Their Price-Earnings Ratios: A Test of the Efficient Market Hypothesis", The Journal of Finance, Vol. 32, No. 3. (Jun, 1977), pp. 663-682.

[7] Benjamin Graham "Security Analysis", ISBN: 0071448209, McGraw-Hill Publisher, 3rd Edition, December 10, 2004.

[8] B. D. Ripley, "Pattern Recognition and Neural Networks", ISBN: 0521460867, Cambridge University Press, 1996.

[9] Chen C., Liaw, A., and Breiman, L. "Using random forest to learn unbalanced data," Technical Report 666, Statistics Department, University of California at Berkeley, 2004. http://www.stat.berkeley.edu/users/chenchao/666.pdf

[10] Falas T., A. Charitou, and C. Charalambous, "The Application of Artificial Neural Networks in the Prediction of Earnings", IEEE, Orlando, 1994.

[11] G.-B. Huang, P. Saratchandran and N. Sundararajan, "A Generalized Growing And Pruning RBF (GGAP-RBF) Neural Network for Function Approximation," IEEE Transactions on Neural Networks, vol. 16, no. 1, pp. 57--67, 2005.

[12] Henry R. Oppenheimer, Gary G. Schlarbaum, "Investing with Ben Graham: An Ex Ante Test of the Efficient Markets Hypothesis", The Journal of Financial and Quantitative Analysis, Vol. 16, No. 3 (September, 1981), 341-360.

[13] Ian H. Witten and Eibe Frank (2005) "Data Mining: Practical machine learning tools and techniques", 2nd Edition, Morgan Kaufmann, San Francisco, 2005.

[14] Janet Lowe, "Benjamin Graham on Value Investing: Lessons from the Dean of Wall Street", ISBN: 0140255346, Penguin Books Publisher, 1995.

[15] Joseph D. Piotroski, "Value Investing: The Use of Historical Financial Statement Information to Separate Winners from Losers", Journal of Accounting Research, Vol. 38, Supplement: Studies on Accounting Information and the Economics of the Firm (2000), pp. 1-41.

[16] José C. Principe, Neil R. Euliano, W. Curt Lefebvre "Neural and Adaptive Systems: Fundamentals through Simulations" ISBN: 0471351679, Wiley Publisher, 1999

[17] J.-S.R. Jang, "ANFIS: Adaptive-Network-Based Fuzzy Inference Systems", IEEE Trans. Systems, Man & Cybernetics 23 (1993) 665—685.

[18] Kubat, M. & Matwin, S. (1997). "Addressing the curse of imbalanced data sets: One-sided sampling", In Proceedings of the 14th International conference on Machine Learning, pp. 179-186, Morgan Kaufmann

[19] MATLAB Help: Neural Network Toolbox and Fuzzy Logic Toolbox

[20] Neural Network FAQ: [Online] ftp://ftp.sas.com/pub/neural/FAQ.html (current April 8, 2006).

[21] Quah T. S., Bobby Srinivasan, "Utilizing Neural Network in Stock Pickings", The 2000 International Conference on Artificial Intelligence, Las Vegas, U.S.A., 26-29 June 2000, pp. 941-946.

[22] Reinganum, Marc R., "Abnormal Returns in Small Firm Portfolios", Financial Analysts Journal, Vol. 37, Issue 2, pp. 52-56.

[23] Richard Frankel, Charles M. C. Lee, "Accounting Valuation, Market Expectation, and Cross-sectional Stock Returns", Vol. 25, Issue 3, 30 June 1998, pp. 283-319.

[24] Simon J. Mason and Nicholas E. Graham, "Conditional Probabilities, Relative Operating Characteristics, and Relative Operating Levels", International Research Institute for Climate Prediction, Scripps Institution of Oceanography, University of California, San Diego, La Jolla, California, 19 April 1999.

[25] Swets, J. A., 1974, "The relative operating characteristics in psychology", Science, 182, 990-1000.

[26] Swets J. A., "Measuring the Accuracy of Diagnostic Systems", Science, Vol. 240, No. 4857, Jun 3 1998, pp. 1285-1293.

[27] T. Takagi, M. Sugeno, 1985, "Fuzzy identification of systems and its application to modeling and control", IEEE Trans. on Systems, Man and Cybernetics, 15(1):116-132.

[28] Vanstone B., Tan C., "A Survey of the Application of Soft Computing to Investment and Financial Trading", 8th Australian & New Zealand Intelligent Information Systems Conference (ANZIIS 2003), Sydney, 2003.

[29] Vanstone B., Finnie G., Tan C., "Enhancing Security Selection in the Australian Stockmarket using Fundamental Analysis and Neural Networks", Proceedings of the 8th IASTED International conference on Artificial Intelligence and Soft Computing (ASC 2004), Marbella, Spain, 1-3 September 2004.

[30] Vanstone B., Finnie G., Tan C., "Applying Fundamental Analysis and Neural Networks in the Australian Stockmarket", Proceedings of the International conference on Artificial Intelligence in Science and Technology (AISAT 2004), Hobart, Tasmania, 21-25 November 2004.

[31] Vanstone B.,Finnie G., Tan C., "Evaluating the Application of Neural Networks and Fundamental Analysis in the Australian Stockmarket", IASTED International Conference on Computational Intelligence, CI 2005, Calgary, AB, Canada, ACTA Press.

[32] Werner F. M. De Bondt, Richard H. Thaler, "Further Evidence on Investor Overreaction and Stock Market Seasonality", The Journal of Finance, Vol. 42, No. 3, Papers and Proceedings of the Forty-Fifth Annual Meeting of the American Finance Association, New Orleans, Louisiana, December 28-30, 1986. (Jul., 1987), pp. 557-581.

[33] Wikipedia: RBF. [Online] http://en.wikipedia.org/wiki/RBF (current April 8, 2006).