VOT 74080

DESIGN AND DEVELOPMENT OF INTELLIGENT KNOWLEDGE DISCOVERY SYSTEM FOR STOCK EXCHANGE DATABASE

DR. MOHD NOOR MD SAP
DR. HARIHODIN SELAMAT
DR. SITI MARIYAM HJ. SHAMSUDDIN
RASHID HAFEEZ KHOKHAR
ZAMZARINA BT CHE MAT @ MOHD SHUKOR
ABDUL MAJID AWAN

RESEARCH VOTE NO:
VOT 74080

Faculty of Computer Science and Information Systems
Universiti Teknologi Malaysia

2005

ABSTRACT

The stock market is a complex, nonstationary, chaotic and non-linear dynamical system. Most of the existing methods suffer from drawbacks like long training times required, often hard to understand results, and inaccurate predictions. This study focuses on data mining approach for stock market prediction. The aim is to discover unknown patterns, new rules and hidden knowledge from large databases of stock index that are potentially useful and ultimately understandable for making crucial decisions related to stock market. The prototype knowledge discovery system developed in this research can produce accurate and effective information in order to facilitate economic activities. The developed prototype consists of mainly two parts: i) based on Fuzzy decision tree (FDT); and ii) based on support vector regression (SVR). In predictive FDT, aim is to combine the symbolic decision trees with approximate reasoning offered by fuzzy representation. In fuzzy reasoning method, the weights are assigned to each proposition in the antecedent part and the Certainty Factor (CF) is computed for the consequent part of each Fuzzy Production Rule (FPR). Then for stock market prediction significant weighted fuzzy production rules (WFPRs) are extracted. The predictive FDTs are tested using three data sets including Kuala Lumpur Stock Exchange (KLSE), New York Stock Exchange (NYSE) and London Stock Exchange (LSE). The results of predictive FDT method are favorably compared with those of other random walk models like Autoregression Moving Average (ARMA) and Autoregression Integrated Moving Average (ARIMA). The SVR prediction system is based on support vector machine (SVM) approach. Weighted kernel based clustering method with neighborhood constraints is incorporated in this system for getting improved prediction results. The SVM based method gives better results than backpropagation neural networks. SVM offers the advantages including: i) there is a smaller number of free parameters; ii) SVM forecasts better as it offers better generalization; iii) training SVM is faster. In essence, both the subsystems (FDT and SVR based) developed in this project are complementary to each other. As the fuzzy decision tree based system gives easily interpretable results, we mainly use it to classify past and present data records. Whereas we use the stronger aspect of the SVR based approach for prediction of future trend of the stock market, and get improved results.

ABSTRAK

Pasaran saham merupakan sistem yang dinamik tak-linear, kompleks, tidak pegun dan rencam. Kebanyakan kaedah peramalan mempunyai pelbagai kelemahan seperti masa latihan yang lama, hasil peramalan yang kurang tepat dan sukar difahami. Penyelidikan ini menjurus kepada mengaplikasikan kaedah perlombongan data terhadap pangkalan data saham yang bersaiz besar. Matlamatnya untuk mengenalpasti pelbagai corak yang masih tidak diketahui, peraturan yang baru dan pengetahuan yang tersembunyi di mana ia berkemungkinan berguna dan bermanfaat serta dapat menjana hasil peramalan yang lebih tepat. Prototaip bagi Sistem Penemuan Pengetahuan telah dibangunkan yang mampu menghasilkan maklumat yang tepat dan berkesan untuk peramalan saham. Prototaip ini mempunyai dua bahagian penting iaitu berasaskan pohon keputusan kabur (FDT) dan berasaskan regresi sokongan vektor (SVR). Peramal FDT berperanan menyatukan pohon keputusan simbolik dengan taakulan anggaran yang terdapat pada perwakilan kabur. Dalam taakulan kabur, pemberat akan diumpukkan pada setiap ungkapan di bahagian hadapan peraturan dan faktor ketetapan (CF) akan dikira bagi bahagian akibat untuk setiap peraturan pengeluaran kabur (FPR). Dalam peramalan saham pula, pemberat peraturan pengeluaran kabur nyata (WFPRs) yang penting akan diekstrak. Peramal FDT diuji dengan tiga set data pasaran saham iaitu Kuala Lumpur, New York dan London. Keputusan menunjukkan peramal FDT adalah lebih baik berbanding dengan model bergerak rawak seperti purata bergerak autoregresif (ARAMA) dan purata bergerak autoregresif (ARIMA). Peramal SVR pula adalah berasaskan mesin sokongan vektor (SVM). Pengelompokan berasaskan pemberat inti dengan kekangan jiran digabungkan dalam subsistem ini bagi memperolehi keputusan yang lebih baik. Hasil menunjukkan SVM adalah lebih baik berbanding dengan kaedah rangkaian neural propagasi ke belakang. Kebaikan SVM ialah ia mempunyai bilangan parameter bebas yang sedikit, menawarkan generalisasi yang banyak dan masa latihan yang singkat. Kedua-dua subsistem (berasaskan FDT dan SVR) yang dibangunkan adalah saling melengkapi di antara satu sama lain. Memandangkan peramal FDT dapat menjana hasil yang mudah diterjemah maka ia digunakan bagi mengkelaskan rekod terdahulu dan semasa. Manakala peramal SVR pula digunakan bagi menjangkakan arah aliran pasaran saham pada masa depan dan menjana hasil yang lebih baik.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ABBREVATIONS

P_O	-	Open price
P_C	-	Close price
V	-	Volume
C_{P_O}	-	Change in open $=\{Low, Average, High\}$,
C_{P_C}	-	Change in close $=\{Low, Average, High\}$,
C_V	-	Change in volume $=\{Low, Average, High\}$,
$Entr_i^{(k)}$	-	Minimum classification information-entropy for each attribute (k)
$Ambig_i^{(k)}$	-	Minimum classification ambiguity for each attribute (k)
$Th = \lambda_\circ$	-	Threshold value
$CF = \mu$	-	Certainty factor
W	-	Weight for single and multiple antecedent
Gw	-	Weighted average
AG_w	-	Aggregated weighted average
$x_i^{(j)}$	-	Data point
c_j	-	Cluster centre
k	-	Linguistic terms
S	-	Nonleaf node
∇	-	Symmetrical difference of A and A'
GR	-	Gain ratio
A	-	Antecedent (pattern)
A'	-	An observation (fact) of A

C	-	Consequent of a rule
C'	-	Result of consequent
$\mu_A(x)$	-	Degree of membership x by linguistic term A
FDT	-	Fuzzy Decision Tree
WFPRs	-	Weighted Fuzzy Production Rules
FPRs	-	Fuzzy Production Rules
KDD	-	Knowledge Discovery in Databases
AARS	-	Approximate Analogical Reasoning Schema
SMP	-	System Marginal Price
FT	-	Function T
DS	-	Degree of Subsethood
IC	-	Inclusion Cardinality
EC	-	Equality and Cardinality
PR	-	Predictive Reasoning
KLSE	-	Kuala Lumpur Stock Exchange
NYSE	-	New York Stock Exchange
LSE	-	London Stock Exchange
CF	-	Certainty Factor
DM	-	Degree of Measure
SVM	-	Support Vector Machine
SVR	-	Support Vector Regression
PCD	-	Prediction of change in direction

CHAPTER 1

INTRODUCTION

1.1 Overview

With the increase of economic globalization and evolution of information technology, financial time series data are being generated and accumulated at an unprecedented pace. As a result, there has been a critical need for automated approaches to effective and efficient utilization of massive amount of financial data to support companies and individuals in strategic planning and decision-making for investment. Therefore, in pursuit of developing an intelligent knowledge discovery system for stock market application, some problems are discussed, in this chapter, that investors face during crucial part of their decision-making process for the selection of real time stock market to invest in. These problems are highlighted by analyzing the existing data mining and statistical techniques for stock market prediction. Among the variety of existing techniques for stock market prediction, support vector machine (SVM), linear and non-linear models, neural networks, association rules, and classification approaches have been found to be good techniques for stock market predictions. However, most of these techniques still have some serious drawbacks to handle time series stock market data. Financial time series forecasting is one of the most challenging applications of modern time series forecasting. This is because financial time series are inherently noisy, nonstationary and deterministically chaotic (Deboeck, 1994; Yaser, 1996; Huang,

et al., 2005; Cao & Tay, 2003; Kim, 2003). The related problems are mentioned in the following sections.

1.2 Problem Background

The stock market is a rather complicated system, and good predictions for its developments are the key to successful trading. Traders must predict stock price movements in order to sell at top range and to buy at bottom range. As stock trading is a very risky business (Torben and Lund, 1997), it is necessary to evaluate the risks and benefits before entering into any trading. The key to realize high profits in stock trading is to determine the suitable trading time when the risk of trading should be minimum. Many attempts have been made for meaningful prediction of stock market by using data mining and statistical techniques like Time Series (Agrawal *et al.*, 1995a; Breidt, 2002), Support Vector Machine (Cao, 2003; Huang *et al*., 2005, Kim, 2003, Cao and Tay, 2003, Yang *et al*., 2002, 2004), Neural Networks (Wang *et al*., 2003; Raymond, 2004), Linear and Non-linear models (Weiss, 2000; Chinn *et al*., 2001), Association Rules (Ke and Yu 2000; Sarjon and Sap, 2002a) and Classification approaches (Agarwal *et al*., 2001; Han, J and Pei, 2000). However, these techniques for predicting the trend of stock market real time data are yet to be achieved as good classifiers (model). The following subsections present some general problem background regarding these techniques for predicting stock market trends, and prompt for the need of designing and developing a knowledge discovery system for this purpose.

1.2.1 Time Series

Investors are facing with an enormous amount of stocks in the market. The meaningful prediction of time series data is one of the fast growing research areas in the field of financial engineering. Investors in the market want to maximize their stock return by buying or selling their investments at an appropriate time. Since stock market

data are highly time-variant and are normally in a nonlinear pattern, predicting the future trend (i.e., rise, decrease, or remain steady) of a stock is a challenging problem. Agrawal *et al*. (1995a; 1995b) introduced different similarity queries on time series data to find the behaviors of selling or buying patterns of different stocks. In queries of this type, approximate rather than exact matching is required. Recently, some researchers used data mining techniques for attempting to index, cluster, classify and mine association rules from increasingly massive sources of time series data (Savnik *et al*., 2002; Yimin and Dit-Yan, 2004).

For example, Keogh and Pazzani (1998) introduce a new scalable time series classification algorithm. Das *et al*. (1998) attempt to show how association rules can be learned from time series. Han *et al*. (1999) investigate time series databases for periodic segments and partial periodic patterns using data mining methods. All these algorithms that operate on time series data need to compute the similarity between patterns by using Euclidean distance, and the Euclidean distance is sensitive to the absolute offsets of time sequences.

Savnik *et al*. (2002) present an algorithm for matching sequences with the set of time series. The matching algorithm proposed by them is not enough to be effectively used for other domains including stock market data, sensor data in engineering environments, and medical measurements. In another attempt, the model-based clustering method (Yimin and Dit-Yan, 2004) can incorporate prior knowledge more naturally in finding the correct number of clusters. However, the clustering performance of model-based clustering method degrades significantly when the underlying clusters are very close to each other. Another problem is that it is not designed for modeling the differences in trend of the time series.

1.2.2 Linear and Non-linear Statistical Models

The Autoregressive Conditional Heteroskedasticity (ARCH) class of models (Chinn *et al*., 2001; Pierre and Sébastien, 2004) have become a core part of empirical finance. ARCH is a nonlinear stochastic process, where the variance is time-varying, and a function of the past variance. ARCH processes have frequency distributions which have high peaks at the mean and fat-tails, much like fractal distributions. The issue of forecasting with ARCH models has been discussed by Koenker and Zhao, (1996). However, the procedures developed are restricted to a limited class of ARCH models, and often do not take account parameter of uncertainty, and often have questionable finite sample properties. Another statistical model is Auto Regressive Integrated Moving Average (ARIMA). ARIMA models have been already applied to forecast commodity prices (Weiss, 2000; Chinn *et al*., 2001). However, since the ARIMA models are linear and most real world applications involve non-linear problems, this introduces a limitation in the accuracy of the predictions generated (Ferreira *et al*., 2004).

1.2.3 Neural Networks

Artificial Neural Networks (ANN) techniques that have been widely used for load forecasting are now used for price prediction (Nicolaisen, 2000; Hippert, 2001). Using neural networks to model and predict stock market returns has been the subject of recent empirical and theoretical investigations by academics and practitioners alike (Wang *et al*., 2003; Raymond, 2004). Wang *et al*. (2003) investigate whether trading volume can significantly improve the forecasting performance of neural networks, or whether neural networks can adequately model such nonlinearity. Such types of neural networks cannot handle the nonlinearity between stock return and trading volume. Raymond (2004) introduce a feasible and efficient solution (iJADE Stock Advisor) for automatic intelligent agent-based stock prediction. However, the model appeared highly sensitive to the training parameters and also these methods have various number of

hidden neurons. As tested in the experiments, no significant improvements appear, it may be due to not finding the optimal architecture and available training methods.

Some of recent studies, however, showed that ANN had some limitations in learning the patterns because stock market data has tremendous noise and complex dimensionality (Kim, 2003). Neural network training requires nonlinear optimization and bears the danger of getting stuck at local minima (Kim, 2003; Huang *et al*., 2005; Cao and Tay 2003). Back-propagation (BP) neural network, the most popular neural network model, however, suffers from difficulty in selecting a large number of controlling parameters which include relevant input variables, hidden layer size, learning rate, momentum term (Kim, 2003).

1.2.4 Support Vector Machine

Recently, support vector machine (SVM), a novel neural network algorithm, was developed by Vapnik and his colleagues (Vapnik, 1998). SVM is a very specific type of learning algorithms characterized by the capacity control of the decision function, the use of the kernel functions and the sparsity of the solution (Vapnik, 1995, 1998; Cristianini & Taylor, 2000). Many traditional neural network models had implemented the empirical risk minimization principle, whereas SVM implements the structural risk minimization principle. The former seeks to minimize the mis-classification error or deviation from correct solution of the training data but the latter searches to minimize an upper bound of generalization error. In addition, the solution of SVM may be global optimum while other neural network models may tend to fall into a local optimal solution. Thus, over-fitting is unlikely to occur with SVM.

Recently, due to the advantage of the generalization power with a unique and global optimal solution, the Support Vector Machine (SVM) has attracted the interest of researchers and has been applied in many applications, e.g., pattern recognition (Burges, 1998), and function approximation (Vapnik *et al*., 1997). Its regression model, the

Support Vector Regression (SVR), has also been successfully applied in the time series prediction (Mukherjee *et al*., (1997), especially in the financial time series forecasting (Cao, 2003; Huang *et al*., 2005, Kim, 2003, Cao and Tay, 2003, Yang *et al*., 2002, 2004). This model, using the ε-insensitive loss function, can control the sparsity of the solution and reduce the effect of some unimportant data points. Extending this loss function to a general ε-insensitive loss function with adaptive margins has shown to be effective in the prediction of the stock market (Yang *et al*., 2002, 2004). In financial data, due to the embedded noise, one must set a suitable margin in order to obtain a good prediction (Yang *et al*., 2002). Yang *et al*., (2002) has extended the standard SVR with adaptive margin and classified into four cases. The model proposed by them requires the adaptation of the margin width and the degree of asymmetry.

In modeling the financial time series, one key problem is its high noise, or the effect of some data points, called outliers, which differ greatly from others. Learning observations with outliers without awareness may lead to fitting those unwanted data and may corrupt the approximation function. This will result in the loss of generalization performance in the test phase. Hence, detecting and removing the outliers are very important. Specific techniques, e.g., a robust SVR network (Chuang *et al*., 2002) and a weighted Least Squares SVM (Suykens *et al*., 2001) have been proposed to enhance the robust capability of SVR. Some researchers tend to include novel factors in the learning process (Ince & Trafalis, 2004; Kim, 2003, Cao and Tay, 2003).

1.2.5 Association Rules

Association rule mining uncovers interesting correlation patterns among a large set of data items by showing attribute-value conditions that occur together frequently. It has been applied successfully in a wide range of business prediction problems (Ke *et al*., 2000; Sarjon and Sap, 2002). Pasquier *et al.* (1999) proposed an algorithm, called a-close, for finding frequent closed itemsets and their support in a stock market database. However, a-close method is costly when mining long patterns or with low minimum

support threshold in large databases like stock market and also cannot generate association rules at higher levels. In another attempt, typical algorithms for discovering frequent itemsets in stock market by using association rules operate in a bottom-up, breadth-first search direction (Lin and Kedem, 2002). The computation starts from frequent 1-itemsets (the minimum length frequent itemsets) and continues until all maximal (length) frequent itemsets are found. During the execution, every frequent itemset is explicitly considered. Such algorithms perform well when all maximal frequent itemsets are short. However, performance drastically depreciates when some of the maximal frequent itemsets are long.

1.2.6 Classification

Among various approaches for discovering different kinds of knowledge from large databases, classification has been recognized as an important problem in data mining (Agarwal *et al*., 2001; Huang and Hsu, 2002). Classification is one kind of data mining technique to identify essential features of different classes based on a set of training data and then classify unseen instances into the appropriate classes. Many popular classification techniques, Apriori-like approach (Agarwal *et al*., 2001; Ma, 2004), FP-growth (Han and Pei, 2000; Xu *et al*., 2002), Naive Bayes' Classifier (Huang and Hsu, 2002), Decision Theoretic Model (Elovici and Braha, 2003), Information Network (Last and Maimon, 2004) have been used as data mining problems. These approaches are still not suitable where the real databases contain all records. In such cases, huge space is required to serve the mining, and large applications need more scalability.

Mostly the trading practice adopted by financial analysts relies on accurate classification prediction of the price levels of financial instruments. However, some recent studies have suggested that trading strategies guided by forecasts on the direction of the change in price level are more effective and may generate higher profits. Wu and Zhang (1997) investigate the predictability of the direction of change in the future spot

exchange rate. In another study, Aggarwal and Demaskey (1997) find that the classification performance of cross-hedging improves significantly if rates can be predicted. O'Connor *et al*., (1997) conduct a laboratory-based experiment and conclude that individuals show different tendencies and behaviors for upward and downward series. The findings in these studies are reasonable because accurate point estimation, as judged by its deviation from the actual observation, may not be a good predictor of the direction of change in the instruments price level.

1.3 Problem Statement

In recent years, there have been a growing number of studies looking at the direction or trend of movements of stock market indices with various kinds of statistical and data mining techniques (such as Mark *et al*., 2000; Elovici and Braha, 2003; Zhang and Zhou, 2004). However, as discussed in the problem background, these techniques have some limitations to handle time series and huge stock market data. Also, none of these studies provide a comparative evaluation of different classification techniques regarding their ability to predict the sign of the index return. Among the variety of data mining techniques, classification has been found to scale well, run fast, and produce highly interpretable results. Therefore, there is a critical need for developing an intelligent knowledge discovery system to predict the stock market behavior.

In this project, one of the main focus is on the development of classification data mining model to identify better decision making classifiers for stock market prediction. To achieve this, decision tree with fuzzy sets has been used to build classification models for predicting classes of unseen records. The predictive FDT algorithm and similarity-based fuzzy reasoning method are used to answer the following research questions:

- How classification data mining techniques are able to uncover hidden patterns and predict future trends and behaviors in financial markets?

- How the trader's expectations can be fulfilled by handling uncertainty factors including political situation, oil price, overall world situation, local stock markets, etc.?
- How the optimal predictive FDT can be constructed that has better comprehensibility (number of rules), less complexity (number of nodes) and better learning accuracy?
- How similarity-based fuzzy reasoning method can improve the learning accuracy of generated weighted fuzzy production rules for the prediction of stock market data?

Moreover, we also explored how support vector machine can be applied for financial forecasting? How the prediction performance of the forecasting system can be improved by preprocessing the data in order to handle noise and outliers in the data? How the technical indicators, like volatility, relative strength index, moving average convergence divergence, relative difference in percentage, etc., affect the prediction performance?

1.4 Objectives of the Project

The objectives of this project are to build an accurate model by using computational intelligence techniques, like classification, fuzzy sets and support vector machine, in order to make the methods accessible, user-friendly, and prepared for the broader population of economists, analysts, and financial professionals. The main objectives of this project are as follows:

- To develop and enhance data mining algorithms for extracting the patterns of knowledge from stock market database.
- To use the developed algorithms for designing and building an intelligent knowledge-discovery system for stock exchange environment.

- To disseminate and promote data mining technology in the financial and stock exchange industries.

1.5 Scope of Project

This project focuses on the enhancement of data mining algorithms in financial application. In data mining algorithms, classification model has been developed to extract the weighted fuzzy production rules for time series stock market prediction. In addition, a support vector machine based model is also implemented for the improvement of prediction results. The scope of this project covers the following points:

- Reviews and comparisons of the existing data mining methods for predicting future trends and behaviors in financial markets.
- Analyze and evaluate the performance of the predictive FDT algorithm by using data from three stock exchanges, i.e. KLSE, LSE, and NYSE.
- Test and analyze the predictive FDT algorithms on the chaotic and complex nature of intraday stock market data from 100 different associate stocks.
- Analyze and evaluate the results of predictive FDT and similarity-based fuzzy reasoning method for prediction of real time stock market.
- Test and analyze the results of support vector regression by incorporating some preprocessing and technical indicators
- Analyze and evaluate the results of SVR with backpropagation neural networks

In essence, both the subsystems (FDT and SVR based) developed in this project are complementary to each other. As the fuzzy decision tree based system gives easily interpretable results, we mainly use it to classify past and present data records. Whereas we use the stronger aspect of the SVR based approach for prediction of future trend of the stock market.

1.6 Project Contributions

In this project, the data mining and artificial intelligence techniques are used to build classification model for stock market prediction. The knowledge discovery system developed in this project can produce accurate and effective information in order to facilitate economic activities. The system consists of mainly two parts: i) based on Fuzzy decision tree (FDT); and ii) based on support vector regression (SVR). In the FDT method: first we construct predictive fuzzy decision tree (FDT) with expressive power of fuzzy reasoning method, and then; for stock market prediction significant weighted fuzzy production rules (WFPRs) are extracted from predictive FDT. The SVR Prediction System is based on Support Vector machines approach. Weighted kernel based clustering method with neighborhood constraints is incorporated in this system for getting improved prediction results. Looking at the forecasting results presented by the two methods, i.e. FDT & SVR methods, the trader can confidently decide whether to buy a stock.

Some points of major contributions of this project are described as follows:

- The enhancement of classification based data mining algorithms to predict trends and behaviors in financial market.
- The construction of predictive fuzzy decision tree for classification, that provides the reasonable performance of the parameters like comprehensibility (number of rules), complexity (number of nodes), and predictive accuracy.
- Careful assignment of some uncertainty factors including oil price, local stock market, natural disasters, etc., for the classification of stock market index.
- In predictive reasoning method, the conventional fuzzy production rules are enhanced by assigning a weighting factor to each proposition in the

antecedent parts to have various degree of relative importance with respect to the same consequent.

- Enhancement of prediction results of SVR by incorporating our developed clustering algorithm–weighted kernel kmeans with neighborhood constraints–able to handle noise and outliers. Because, in the financial time series, the recent data points could provide more important information than the distant data points, in the weighted clustering algorithm, the weights are assigned accordingly.
- Enhancement of prediction performance by considering the technical indicators, like volatility, relative strength index, moving average convergence divergence, relative difference in percentage, etc.

1.7 Project Report Organization

The general research background is given in this chapter. Next chapter presents a brief overview of previous works in data mining and statistical techniques for financial application. In chapter 3, after formulating the problem, we discuss the other important steps like how to clean the data, engineer the data to be maximally useful with the problem at hand, engineer a mining algorithm, run the mining algorithm and explain the results to the expert. In chapter 4, predictive fuzzy decision tree is constructed. In this chapter, the first three important steps to be implemented before applying the predictive FDT algorithm are discussed. In the first step, k-means clustering method is used to find the center for every fuzzy set. Then, the degree of membership is calculated by applying triangular membership function. Finally, the predictive FDT is constructed by using fuzzy decision tree algorithm. In chapter 5, similarity-based fuzzy reasoning method is used for mining weighted fuzzy production rules. In this chapter, the experiments and results are presented that are produced from predictive FDT and similarity-based fuzzy reasoning method. The accuracy of extracted WFPRs are also compared with the standard methods of stock market prediction. In chapter 6, we use support vector machine for financial forecasting. In this chapter, we use weighted kernel k-means for

improving the prediction performance. Finally in chapter 7, the conclusions on predictive fuzzy decision tree, extracted weighted fuzzy production rules and support vector machine approach have been given.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

In pursuit of developing an intelligent knowledge discovery system, statistical and data mining techniques are addressed and some important issues related to predicting future trends and behaviors in financial markets are discussed in this chapter. Some difficulties with existing classification techniques are also discussed to build accurate and efficient classifiers for data mining of time series stock market data. Three most popular fuzzy decision trees are discussed to underline how fuzzy decision trees are more suitable than the existing stock selection techniques in the perspective of knowledge discovery in databases. In this regard, three popular similarity-based fuzzy reasoning methods are addressed to mine significant WFPRs. Moreover, it is also discussed how support vector machine can be useful for stock market prediction.

2.2 Stock Exchange Mechanism

From the perspective of market design, it is important to understand how to concretely implement such a trading system. Consider, for instance, a trader submitting an order to buy a given vector of assets (Giovanni, 2003), the mechanism of the buying-selling procedure can be simply illustrated by looking at the following example

(Okamoto *et al*., 1995). A person X decides to invest in securities. Therefore, he asks a broker from a Stock Exchange Company to search for the best investment possible for a long-term period. The optimization goal is to obtain the good with the best relation between bid and offer.

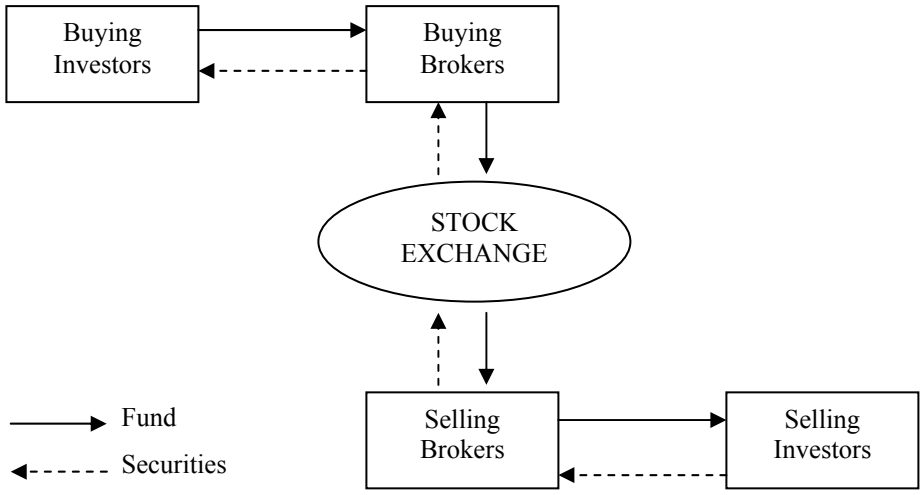


Figure 2.1: Typical trading processes.

On the other hand, a different person, Y, wants to have quickly some liquidity, in order to purchase some expensive goods. Therefore, he decides to sell some shares of a company, and contacts his broker and encharges him with the job. Both brokers access the electronic market data system and acquire data about the quotes on the market.

Thereafter, both brokers forward their data to the customers. If person X and Y decide to buy and sell respectively the same good G (here: shares) in a certain amount A, at the current market price, the respective brokers either transmit the orders directly to the trading post, through a system called the "Super Dot" system, or they have to follow the hierarchical order of sending their orders to the floor of the Stock Exchange (in this example considered the same). Wherefrom they are sent to each brokerage firm's own clerks and given to the firm's brokers, who take it to the trading post of the Stock Exchange Floor. At the post, the specialist in that company assures fare and orderly transactions (NYSE Net).

The floor brokers on the trading post try to get the best price for their customers, and the brokers representing persons X and Y agree finally on a price. The transaction is entered electronically, and persons X and Y are informed. Through the computer, the transaction is reported within minutes and appears on tickertapes across the country and around the world.

The transaction takes place electronically, crediting X's brokerage firm and debiting Y's. X settles its account within 3 business days, by paying for the amount A and a commission to his broker. Y settles its account within 3 business days, by collecting the money from the broker minus the commission.

The Stock Exchange is, in short terms, an auction market, where stocks are bought and sold at prices determined by the bids and offers of investors. These are represented on the trading floor by floor professionals, who use their skill, judgment and experience in order to obtain the best possible prices for their customers. The whole process is supported and made easier by the extensive usage of advanced technology. The stock exchange is the place where securities–negotiable investment claims against assets and their periodical return–are bought and sold (Fritz, 1940).

2.2.1 Stock Market Returns

Stock market has long been considered a high return investment field (Jing *et al.*, 1997). Due to the fact that stock markets are affected by many highly interrelated economic, political and even psychological factors, it is very difficult to forecast the movement of stock market. Kuala Lumpur Stock Exchange (KLSE) is popular among the researchers because KLSE is one of the largest markets in the emerging economies in terms of capitalization. There are several journals describing the prediction of KLSE index by using neural network (Jing *et al*., 1997). Therefore, it is not surprising that it gains the interest of investor to predict the market.

Because the stock price patterns have non-linear time elasticity and differences in names and time spans, it is difficult to find these patterns by statistical models or simple rule-based approaches (Tetsuji *et al.*, 1992). Several researchers claim that the stock market and other complex systems exhibit chaos, a nonlinear deterministic process which only appears random because it can not be easily expressed (Raman, 1997).

Many researches have been carried out in order to provide financial analysts with a clear direction with which they can predict movements of stocks in a certain direction. The ability of the network to keep updating it with historical information will allow it to predict the data even more accurately and thus aim to increase the profitability for the investors and the users. Some aspects that wish to be achieved by current researchers include (Asim *et al.*, 2005):

- To aid the users to make valuable decisions by using intelligent computational techniques.
- Help the investors to generate and maximize their profits out of the investments they make in the stock market.
- Reduce the risk involved when investing in stocks.
- Support the users in an efficient portfolio management

Stock market is a type of time series forecasting where it takes an existing series of data and forecast the future data values. Data of daily stock price can be treated as a time series. If we have the daily prices of a certain period, they can be fed to a network which predicts the future price of the first day following the end of the known period. We repeat this process iteratively to predict the price for the future working days (Saad *et al*., 1996). In recent decades, an increasing focus has been put on this time series forecasting field especially on stock market prediction.

There are several reasons for trying to predict stock market prices. Stock market investment has become an important means of individual finance (Teh, 2005).

Apparently, it is significant for investors to estimate the stock price and then select the trading opportunities accurately in advance, which will bring high return to stockholders (Jing *et al.*, 1997). Consequently, investors are seeing the need for flexible forecasting models.

2.3 Stock Market Prediction Techniques

Normally, techniques on stock market prediction can be divided into two main categories. They are traditional techniques and intelligent techniques. The traditional techniques can be further categorized into technical analysis and fundamental analysis. Nowadays, intelligent techniques have become the main focus of the current researches and have been widely applied in stock market prediction.

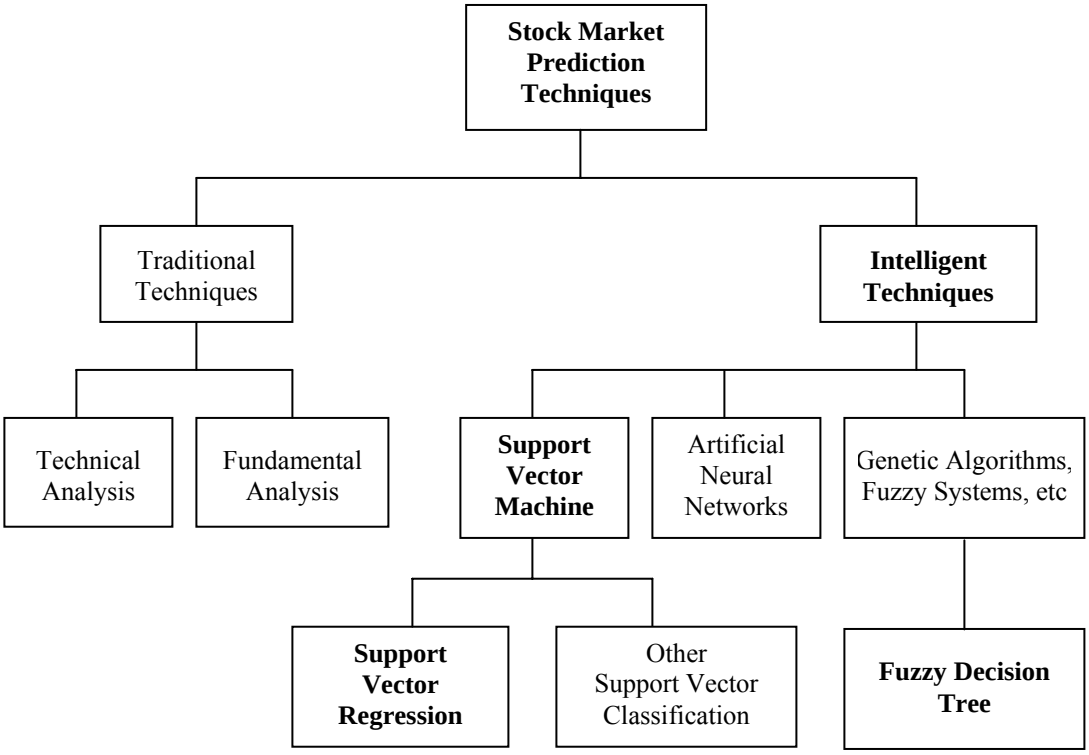


Figure 2.2: Stock market prediction techniques.

The intelligent techniques consist of ANN, GA, Fuzzy Systems, SVM and others. These intelligent techniques are the superior prediction techniques that are widely applied by many individuals, including researchers, investment professionals and average investors. Figure 2.2 shows the techniques that have been applied in stock market prediction field (Teh, 2005).

2.3.1 Traditional Approaches

In the past, the major forecasting methods used in the financial area are either technical analysis or fundamental analysis before the introduction of intelligent techniques.

In a typical free market, trading is stimulated by the market moving prices up to tempt the sellers or moving prices low enough to attract the buyers. This price fluctuation thus behaves as a goad for promoting opportunities. The market as a result tries to establish an equilibrium between buying and selling forces. This dynamic mechanism of trading has inspired traders to predict the future trends of the market (Jang *et al.*, 1991).

Technical analysis has become popular over the past several years, as more and more people believe that the historical performance of a stock is a strong indication of future performance. The use of past performance should not come as a big surprise. People using fundamental analysis have always looked at the past performance by comparing fiscal data from previous quarters and years to determine future growth (Investopedia, 2006).

Table 2.1: Stock market analysis.

Analysis Types	Descriptions
Fundamental analysis	• Forecasting is based on macro economic data such as exports and imports, money supply, interest rates, foreign exchange rates, inflationary rates and unemployment figures, etc. • Basic financial status of companies is also taken into consideration.
Technical Analysis	• Predictions are made by exploiting implications hidden in past trading activities by analyzing patterns and trends shown on the price and volume charts. • Based on the rationale that history will repeat itself and that the correlation between price and volume reveals market behavior. • Does not take any external influential factors, like breaking news, into consideration. • It is a method of evaluating securities by analyzing statistics generated by market activity, past prices, and volume. • Does not attempt to measure a security's intrinsic value. • Looks for patterns and indicators on stock charts that will determine a stock's future performance (Investopedia, 2006).

The difference lies in the technical analyst's belief that securities move with very predictable trends and patterns. These trends continue until something happens to change the trend, and until this change occurs, price levels are predictable. Therefore, intelligent techniques, especially SVM, fuzzy systems, etc., that are motivated for pattern recognition have gained interest among the investors.

2.3.2 Prediction

Today the field of financial market research seems to be at the exciting stage of "crisis"–past results are being questioned, and new solutions are being proposed. In recent years, analysts have suggested that time series models better predict the price of the stock as compared to the traditional approaches (Asim *et al*., 2005). The forecasting of financial time series relies on the discovery of strong empirical regularities in observations of the system. Because these regularities are often masked by noise and time series often have nonlinear and non-stationary behavior it has been suggested that some financial time series are very difficult to predict.

According to Fayyad *et al.* (1996), prediction, or also known as regression, is a learning function that maps a data item to a real-valued prediction variable. Regression is the study of relationships among variables, a principal purpose of which is to predict, or estimate the value of one variable from known or assumed values of other variables related to it. Once the pattern is established, we can interpret and integrate it with other data. Besides that, prediction is also associated with the discovery and use of patterns in large volumes of time series data in which prediction of future values can be made (Liu *et al.*, 2001).

Time series prediction is a field that has attracted many researchers in the past and also in the present. Much of the researches conducted in time series prediction are stemmed from the pervasiveness of time series in daily life such as stock and foreign exchange trends, weather changes and required inventory levels. Several time-series prediction techniques have been applied in various domains such as linear stochastic auto-regressive moving-average models (ARMA), artificial neural network (ANN) and K-nearest-neighbor (K-NN) (Toth *et al.*, 2000), and others. Below are some common techniques that are being applied by researchers.

Table 2.2: Techniques applied in time-series prediction.

Techniques	Description
Smoothing	Smoothing techniques are used to reduce irregularities (random fluctuations) in time series data. They provide a clearer view of the true underlying behavior of the series.
Exponential Smoothing	Exponential smoothing is a smoothing technique used to reduce irregularities (random fluctuations) in time series data, thus providing a clearer view of the true underlying behavior of the series. It also provides an effective means of predicting future values of the time series (forecasting).
Moving Average Smoothing	A moving average is a form of average which has been adjusted to allow for seasonal or cyclical components of a time series. Moving average smoothing is a smoothing technique used to make the long term trends of a time series clearer.
Running Medians Smoothing	Running medians smoothing is a technique analogous to that used for moving averages. The purpose of the technique is the same, to make a trend clearer by reducing the effects of other fluctuations.
Differencing	Differencing is a popular and effective method of removing trend from a time series. This provides a clearer view of the true underlying behavior of the series.
Autocorrelation	Autocorrelation is the correlation (relationship) between members of a time series of observations, such as weekly share prices or interest rates, and the same values at a fixed time interval later.
Extrapolation	Extrapolation is used when the value of a variable is estimated at times which have not yet been observed. This estimate may be reasonably reliable for short times into the future, but for longer times, the estimate is liable to become less accurate.

The classic (1970s) time series analysis approach uses a Box Jenkins algorithm to transform the data from a stationary process into Auto Regression and Moving Average components, the so-called ARMA. The method allows estimation of actual pulses or inclusion of forecast pulses that do not fit the ARMA methodology (Box and Jenkins, 1976).

Artificial neural network (ANN) is a non-linear, data-driven approach that depends on the available data to be "learned", without any a priori hypothesis about the kind of relationship, which makes it suitable for complex data. However, neural network depends on network structure and complexity of samples, which cause over fitting and low generalization (Lin and Lin, 2003; Kim and Valdes, 2003; Zhu *et al.*, 2002). This approach relies on building layers of nodes, each connected to a preceding layer through weights. By adding more and more layers, and incorporating memory delay lines so that previous inputs can be stored, the output of the network can come to resemble the time series (Junaida, 2005).

Support Vector machine is a new machine learning algorithm that have received much attention among researchers due to its principle of structural minimization risk which have greater generalization and proved success in time series prediction (Johan, 2001). SVM works by finding a hyperplane that separates the training set in a feature space induced by a kernel function used as the inner product in the algorithm (Cristianini and Taylor, 2000). The successes of SVM in time series prediction are evident from several researches; stock price forecasting (Bao *et al.*, 2004), traffic speed prediction (Vanajakshi and Rilett, 2004) and travel time series prediction (Wu *et al.*, 2004).

2.4 Time Series Forecasting

According to Engineering Statistic Handbook, time series can be defined as an ordered sequence of values of a variable at equally spaced time intervals. Some distinctive properties of time series include:

- continuous vs. discrete
- univariate vs. multivariate
- evenly sampled vs. unevenly sampled
- periodic vs. aperiodic
- stationary vs. nonstationary
- short vs. long

These properties, as well as the sampling interval and temporal overlap of multiple series, must be considered in selecting a dataset for analysis.

According to (Bowerman & O'Connell, 1979), forecasting is the act of making prediction of future events and conditions while time series data is used in making predictions. In recent years, much literature on time series prediction have been utilizing numerous time series data with prediction methods that include exchange rate data, stock market index, air pollutant data, electricity load data and traffic speed data (Toth *et al.*, 2000; Kamruzzaman *et al.*, 2003; Lu *et al.*, 2002; Chen *et al.*, 2004; Vanajakshi & Rilett, 2004).

In general, there are two main goals of time series analysis and prediction:

- Identifying the nature of the phenomenon represented by the sequence of observations in the past and by the sequence of other system variables (supporting attributes) in case of the multivariate time series
- Forecasting – predicting future values of the phenomenon.

Both of these goals require that the pattern of observed time series data is identified and more or less formally described. Once the pattern is established, it can be interpreted and integrated with other data. As in most other analyses, in time series analysis it is assumed that the data consists of a systematic pattern (usually a set of identifiable components) and random noise (error) that usually makes the pattern difficult to identify (Jan *et al.*, 2003). Forecasting in time series is a common problem.

Different approaches have been investigated in time series prediction over the years (Weigend, 1993). The methods can be generally divided into global and local models as given in Table 2.3.

Table 2.3: Global and local model in time series forecasting.

Models	Descriptions
Global model	• Only one model is used to characterize the phenomenon under examination. • Generally better results with stationary time series. Stationary series are series that do not change with time (Jan *et al.*, 2003).
Local model	• Dividing the data set into smaller sets, each being modeled with a simple model.

Using a statistical approach, the integrated autoregressive moving average (ARIMA) methodology have been developed (Box *et al.*, 1994). The methodology is useful for fitting a class of linear time series models. Statisticians in a number of ways have addressed the restriction of linearity in the Box-Jenkins approach. Robust versions of various ARIMA models have been developed. More recently, machine learning techniques (mainly neural networks) have been studied as an alternative to these nonlinear model-driven approaches. The process of constructing the relationships between the inputs and output variables is addressed by certain general-purpose 'learning' algorithms (Jan *et al.*, 2003).

Time series data are often examined with the purpose of discovering the historical patterns that can be exploited in the preparation of a forecast. In order to identify the patterns embedded in the data, it is convenient to consider time series as consisting of four components which are trend, cycle, seasonal variations and irregular fluctuations (Bowerman & O'Connell, 1979). Here are some descriptions regarding these components:

- Trend - referred to the upward or downward movement that characterizes a time series over a period of time.
- Cycle - refers to recurring fluctuations around trend levels which can last from 2 to 10 years or even longer measured from the peak to peak.
- Seasonal variations - defined as periodic patterns in a time series that are completed within the period of a calendar year and are then repeated on a yearly basis.

In forecasting, it is important to take into account the different components of time series since no forecasting technique best suits to every time series components (Bowerman & O'Connell, 1979). The analysis of time series, which exhibits seasonal variation, depends on whether it is appropriate to measure the seasonal effect or eliminate seasonality. In a study by Chen *et al.* (2004) in load forecasting, they found out that seasonality helps to improve prediction performance due to the capability of an SVM model to identify the patterns embedded in the electricity load.

Before discussing into detail on stock market prediction, it is better to understand about the useful forecasting methods. Table 2.4 enlists some commonly used forecasting methods.

For using any method for forecasting, one must use a performance measure to assess the quality of the method. Mean Absolute Deviation (MAD), and Variance are the most useful measures. However, MAD does not lend itself to further use for making inferences, but that the standard error does. For the error analysis purposes, variance is preferred since variances of independent (uncorrelated) errors are additive. MAD is not additive. Nevertheless, there are no forecasting methods that best suit to every time series components (Bowerman & O'Connell, 1979).

Table 2.4 Forecasting methods (Hossein, 1996).

Methods	Descriptions
Multiple Regression Analysis	Used when two or more independent factors are involved-widely used for intermediate term forecasting. Used to assess which factors to include and which to exclude. Can be used to develop alternate models with different factors.
Nonlinear Regression	Does not assume a linear relationship between variables-frequently used when time is the independent variable.
Trend Analysis	Uses linear and nonlinear regression with time as the explanatory variable-used where pattern over time.
Decomposition Analysis	Used to identify several patterns that appear simultaneously in a time series-time consuming each time it is used-also used to deseasonalize a series.
Moving Average Analysis	Simple Moving Averages-forecasts future values based on a weighted average of past values-easy to update.
Weighted Moving Averages	Very powerful and economical. They are widely used where repeated forecasts required-uses methods like sum-of-the-digits and trend adjustment methods.
Adaptive Filtering	A type of moving average which includes a method of learning from past errors-can respond to changes in the relative importance of trend, seasonal, and random factors.
Exponential Smoothing	A moving average form of time series forecasting-efficient to use with seasonal patterns- easy to adjust for past errors-easy to prepare follow-on forecasts-ideal for situations where many forecasts must be prepared-several different forms are used depending on presence of trend or cyclical variations.
Hodrick-Prescott Filter	This is a smoothing mechanism used to obtain a long term trend component in a time series. It is a way to decompose a given series into stationary and nonstationary components in such a way that there sum of squares of the series from the nonstationary component is minimum with a penalty on changes to the derivatives of the nonstationary component.
Modeling and Simulation	Model describes situation through series of equations-allows testing of impact of changes in various factors-substantially more time-consuming to construct-generally requires user programming or purchase of packages such as SIMSCRIPT. Can be very powerful in developing and testing strategies otherwise non-evident.
Probabilistic Models	Use Monte Carlo simulation techniques to deal with uncertainty-gives a range of possible outcomes for each set of events.
Forecasting error	All forecasting models have either an implicit or explicit error structure, where error is defined as the difference between model prediction and the "true" value.

Time series data are of growing importance in many new database applications, such as data mining. A time series is a sequence of real numbers, each number representing a value at a time point. For example, the sequence could represent stock or commodity prices, sales, exchange rates, weather data, biomedical measurements, etc. Different similarity queries on time-series have been introduced (Agrawal *et al*., 1995a, 1995b). For example, a user may want to find stocks that behave in approximately the same way (or approximately the opposite way) for hedging, or to find products that had similar selling patterns during the last year or years when the temperature patterns in two regions of the world were similar. In queries of this type, approximate rather than exact matching is required. However mining different queries from huge time-series data is one of the important issues for researchers. In useful data mining techniques like classification and clustering, to handle time-series data is one of the stimulating research issues.

Recently there has been an explosion of interest in time series data mining, with researchers attempting to index, cluster, classify and mine association rules from increasingly massive sources of data (Han *et al*., 1999; Savnik *et al*., 2002; Yimin & Dit-Yan, 2004). For example, Keogh and Pazzani (1998) introduce a new scalable time series classification algorithm. Das *et al*. (1998) attempt to show how association rules can be learnt from time series. It uses a sliding window to discretize the time series and then cluster these subsequences using a suitable measure of patterns. From these patterns they can find rules relating patterns in a time series to other patterns in that series, or patterns in one series to patterns in another series. Han *et al*. (1999) investigate time series databases for periodic segments and partial periodic patterns using data mining methods. Their techniques are aimed at discovering temporal patterns.

All these algorithms that operate on time series data need to compute the similarity between patterns. Similarity search based these algorithms are based on the minimum distance in large time series databases. For this purpose, these algorithms focus on similarity matching and retrieval of time series by using Euclidean distance.

However, the Euclidean distance is sensitive to the absolute offsets of time sequences, so two time sequences that have similar shapes but with different vertical positions may be classified as dissimilar. And, also these methods are very sensitive to the computational algorithm with regard to the window's width and the distance.

Savnik *et al*. (2002) present an algorithm for matching sequences with the set of time series. The algorithm can be employed for locating the starting time points of the undated sequences of values, and for locating the pattern templates which are common to the set of sequences. The algorithm was successfully used for the problems of dating and pattern recognition in tree-ring time series. However, the matching algorithm is not enough to be effectively used for other domains including stock market data, sensor data in engineering environments, and medical measurements. The model-based clustering method (Yimin and Dit-Yan, 2004) can incorporate prior knowledge more naturally in finding the correct number of clusters. Also, for time series data, they provide a principled approach for handling the problem of modeling and clustering time series of different lengths. However, the clustering performances of model-based clustering method degrade significantly when the underlying clusters are very close to each other. Another problem is that it is not designed for modeling the differences in trend of the time series.

2.5 Stock Market Prediction

In stock market, shares of stocks are bought and sold in an organized way for people to buy and sell stocks and corporations to raise money. Predicting stock market has been a research topic in the field of financial engineering for many years. Several recent researches presented encouraging results on stock market prediction using statistical and data mining techniques.

The digital revolution has made digitized information easy to capture and fairly inexpensive to store (Fayyad *et al*., 1996a; Han and Kamber, 2001). With the

development of computer hardware and software and the rapid computerization of business, huge amount of data have been collected and stored in databases. The rate at which such data is stored is growing at a phenomenal rate. As a result, traditional ad hoc mixture of statistical techniques and data management tools are no longer adequate for analyzing this vast collection of data. Several domains where large volumes of data are stored in centralized or distributed databases include the financial Investment (e.g., stock indexes and prices, interest rates, credit card data) (Dwinnell *et al*., 2002). In the following subsections, some useful data mining and statistical methods for the prediction of time series stock market data are discussed.

2.5.1 Linear and Non-Linear Statistical Models

The autoregressive conditional heteroskedasticity (ARCH) model, a class of models, was originally introduced by (Engle, 1982) and has become a core part of empirical finance. The issue of forecasting with these models has been discussed by (Baillie and Bollerslev, 1992; Koenker and Zhao, 1996). However, the procedures developed are restricted to a limited class of ARCH models, often do not take account of parameter uncertainty, and often have questionable finite sample properties (Baillie and Bollerslev, 1992). Gaussian asymptotic prediction intervals were developed for the class of ARMA-GARCH models. However, for other ARCH models, such as ARCH-in-mean (Engle *et al*., 1987), analytical closed form formulae for prediction intervals do not currently exist.

Auto Regressive Integrated Moving Average (ARIMA) models have been already applied to forecast commodity prices (Weiss 2000; Chinn *et al*., 2001) such as oil (Morano, 2001) or natural gas (Buchanan *et al*., 2001). In power systems, ARIMA techniques have been used for load forecasting (Gross and Galiana, 1987; Hagan and Behr, 1987) with good results. Currently with the restructuring process that is taking place in many countries, simpler Auto Regressive (AR) models are also being used to predict weekly prices, like in the Norwegian system (Fosso *et al*., 1999). However, since

the ARIMA models are linear and most real world applications involve non-linear problems, this introduces a limitation in the accuracy of the predictions generated (Ferreira *et al*., 2004). In order to overcome this limitation, many other approaches have been developed by independent researches such as the bilinear models (Subba and Gabr, 1984), the threshold models (Ozaki, 1985), the exponential autoregressive models (Priestely, 1988), and the general state dependent models (Rumelhart and McCleland, 1987). However these nonlinear models are still limited in that an explicit relationship for the data series at hand has to be hypothesized with little knowledge of the underlying data generating process.

2.5.2 Artificial Neural Networks (ANNs)

Recent research activities in artificial neural networks (ANNs) have shown that ANNs have powerful pattern classification and pattern recognition capabilities (Wang, 2003; Raymond, 2004). Inspired by biological systems, particularly by research into the human brain, ANNs are able to learn from and generalize from experience. Currently, ANNs are being used for a wide variety of tasks in many different fields of business, industry and science (Widrow *et al*., 1994).

If a consensus has arisen regarding ANNs, it is that they can outperform traditional models in certain situations, but neither approach dominates the other (Callen *et al*., 1996). Gorr (1994) and Hill *et al*. (1994) suggest that future research should investigate and better define the borderline between where ANNs and traditional regression techniques outperform one another. Since construction and implementation of neural network models is considerably more difficult and time consuming than using simpler regression-type models, forecasters may want to build ANN models only if there is a strong a priori belief that additional complexity is warranted (Balkin and Ord, 2000). The foremost reason for using ANNs is that there is some non-linear aspect to the forecasting problem being considered (Balkin and Ord, 2000; Tacz, 2001). The non-linearity may take the form of a complex non-linear relationship between the

independent and dependent variables, the existence of upper or lower thresholds for the influence of independent variables, or differences between forecasting up or down movements of the dependent variable. The advantage of using an ANN is that the researcher need not know the type of functional relationship that exists between the independent and dependent variables (Darbellay and Slama, 2000).

Szkuta *et al*. (1999) present the system marginal price (SMP) short-term forecasting implementation using the ANN computing technique. The described approach uses the three-layered ANN paradigm with back-propagation. The retrospective SMP real world data acquired from the deregulated Victorian power system was used for training and testing the ANN. Despite the fact that the achieved results were very satisfactory but the training data set was relatively small which together with the large daily variation in SMP made the ANN model rather difficult to evaluate. The model response appeared highly sensitive to the training parameters.

Using neural networks to model and predict stock market returns has been the subject of recent empirical and theoretical investigations by academics and practitioners alike (Wang *et al*., 2003). However several design factors significantly impact on the accuracy of neural network forecasting. Wang *et al*. (2003) investigate whether trading volume can significantly improve the forecasting performance of neural networks, or whether neural networks can adequately model such nonlinearity. Such types of neural networks cannot handle the nonlinearity between stock return and trading volume. Also, trading volume cannot significantly improve the forecasting accuracy of the suggested (Wang *et al*., 2003) neural networks. And, the improvements, if any, in mean absolutely percentage errors are quite erratic and infrequent by using neural networks.

Raymond (2004) introduce a feasible and efficient solution (iJADE Stock Advisor) for automatic intelligent agent-based stock prediction. With respect to the stock forecasting engine, Raymond (2004) adopts the hybrid RBF network (HRBFN) model for stock prediction. Through the integration of iJADE framework with the HRBFN in the iJADE stock analyst agent, Raymond (2004) illustrates the efficiency and

effectiveness of iJADE Stock Advisor for online stock prediction. From an application point of view, Raymond (2004) demonstrates how HRBFN model can be successfully integrated with mobile-agent technology to provide a truly intelligent, mobile and interactive stock advisory solution. However, the model appeared highly sensitive to the training parameters and also these methods have various numbers of hidden neurons. As tested in the experiments, no significant improvements appear, it may be due to not finding the optimal architecture and available training methods.

Several distinguishing features of neural networks make them valuable and attractive for a forecasting task. Similarly many of the publications on financial markets by neural network researches have an “ad-hoc” or “financial engineering” flavor and do not relate to the broader theoretical infrastructure and “behavioral assumption” used in Economics and Finance. For this reason, unfortunately the broader academic community in Economics and Finance do not take much of this research seriously.

Some of recent studies, however, showed that ANN had some limitations in learning the patterns because stock market data has tremendous noise and complex dimensionality (Kim, 2003). Neural network training requires nonlinear optimization and bears the danger of getting stuck at local minima (Kim, 2003; Huang *et al*., 2005; Cao and Tay 2003). Back-propagation (BP) neural network, the most popular neural network model, however, suffers from difficulty in selecting a large number of controlling parameters which include relevant input variables, hidden layer size, learning rate, momentum term (Kim, 2003).

2.5.3 Support Vector Machine (SVM)

Recently, support vector machine (SVM), a novel neural network algorithm, was developed by Vapnik and his colleagues (Vapnik, 1998). SVM is a very specific type of learning algorithms characterized by the capacity control of the decision function, the use of the kernel functions and the sparsity of the solution (Vapnik, 1995, 1998;

Cristianini & Taylor, 2000). Many traditional neural network models had implemented the empirical risk minimization principle, whereas SVM implements the structural risk minimization principle. The former seeks to minimize the miss-classification error or deviation from correct solution of the training data but the latter searches to minimize an upper bound of generalization error. In addition, the solution of SVM may be global optimum while other neural network models may tend to fall into a local optimal solution. Thus, over-fitting is unlikely to occur with SVM.

Recently, due to the advantage of the generalization power with a unique and global optimal solution, the Support Vector Machine (SVM) has attracted the interest of researchers and has been applied in many applications, e.g., pattern recognition (Burges, 1998), and function approximation (Vapnik *et al*., 1997). Its regression model, the Support Vector Regression (SVR), has also been successfully applied in the time series prediction (Mukherjee *et al*., 1997), especially in the financial time series forecasting (Cao, 2003; Huang *et al*., 2005; Kim, 2003; Cao and Tay, 2003; Yang *et al*., 2002, 2004). This model, using the ε-insensitive loss function, can control the sparsity of the solution and reduce the effect of some unimportant data points. Extending this loss function to a general ε-insensitive loss function with adaptive margins has shown to be effective in the prediction of the stock market (Yang *et al*., 2002, 2004). In financial data, due to the embedded noise, one must set a suitable margin in order to obtain a good prediction (Yang *et al*., 2002). Yang *et al*. (2002) has extended the standard SVR with adaptive margin and classified into four cases. The model proposed by them requires the adaptation of the margin width and the degree of asymmetry.

In modeling the financial time series, one key problem is its high noise, or the effect of some data points, called outliers, which differ greatly from others. Learning observations with outliers without awareness may lead to fitting those unwanted data and may corrupt the approximation function. This will result in the loss of generalization performance in the test phase. Hence, detecting and removing the outliers are very important. Specific techniques, e.g., a robust SVR network (Chuang *et al*., 2002) and a weighted Least Squares SVM (Suykens *et al*., 2001) have been proposed to enhance the

robust capability of SVR. Some researchers tend to include novel factors in the learning process (Ince & Trafalis, 2004; Kim, 2003, Cao and Tay, 2003). Detail about SVM is given in section 2.8.

2.5.4 Association Rules

Association rule is a popular data mining task for discovering knowledge from large amount of data in databases. It has been applied successfully in a wide range of business predicting problems (Bing and Wynne, 1999; Ke *et al*., 2000; Sarjon and Sap 2002a). Pasquier and Bastide (1997) proposed a new efficient algorithm, called a-close, for finding frequent closed itemsets and their support in a stock market database. A-close algorithm used closed itemset lattice framework that can be used for discovering frequent itemsets from stock market. Some advantages of a-close method and algorithms for the generation of all frequent itemsets and reduced itemsets from a database can be found in appendix B. However, a-close method is costly when mining long patterns or with low minimum support threshold in large database like stock market, and also it cannot generate association rules at higher levels. Typical algorithms for discovering frequent itemsets in stock market by using association rules operate in a bottom-up, breadth-first search direction (Lin and Kedem, 2002). The computation starts from frequent 1-itemsets (the minimum length frequent itemsets) and continuous until all maximal (length) frequent itemsets are found. During the execution, every frequent itemset is explicitly considered. Such algorithms perform well when all maximal frequent itemsets are short. However, performance drastically depreciates when some of the maximal frequent itemsets are long.

Bing and Wynne (1999) proposed a method, called MSApriori, for mining association rules with multiple minimum supports. In MSApriori algorithm, the minimum support of a rule is expressed in terms of minimum item support (MIS) of the items that appear in the rule. Using MIS in MSApriori algorithm offers some of the following advantages: (i) the user effectively expresses different support requirements

for different rules, and (ii) enable users to achieve the goal of having higher minimum support for rules that only involve frequent item-sets, and having lowering minimum support for rules that involve less frequent items. However, this method cannot discover association rules from different abstraction levels since it is conducted for mining association rules from raw data, and also it cannot identify which of the generated rules are interesting or not since it does not apply measurement association rules techniques.

Intelligent Mining Association Rules (IMAR) is another method for stock market prediction proposed by Sarjon and Sap (2002a; 2002b). IMAR consists of two main steps; training and running steps. In the training step, neural network knowledge base of data cleaning is created. Then, association rules step is used on the created neural networks knowledge base to support the running step for generating learnt complete data, transformed data and interesting rules intelligently and accurately. IMAR can reduce the number of generated interesting rules without loss of information and increase the performance accuracy up to 99% (Sarjon and Sap, 2002a; Sarjon and Sap, 2002b). IMAR has the following limitations: 1) cannot handle incomplete categorical and text data, 2) transforming raw data into higher levels and other data types, i.e. categorical and text data.

2.5.5 Fuzzy Sets

The modeling of imprecise and qualitative knowledge, as well as the transmission and handling of uncertainty at various stages are possible through the use of fuzzy sets (Mitra *et al*., 2002). Fuzzy logic is capable of supporting, to a reasonable extent, human type reasoning in natural form. Despite a growing versatility of knowledge discovery systems, there is an important component of human interaction that is inherent to any process of knowledge representation, manipulation, and processing. Fuzzy sets are inherently inclined toward coping with linguistic domain knowledge and producing more interpretable solutions. Many attempts have been made to identifying interesting patterns and describing them in a concise and meaningful

manner by using fuzzy sets and knowledge discovery in databases (Umanol *et al*., 1994; Wang *et al*., 2001; Dong and Kothari, 2001).

Experts predict the stock market using vague, imperfect and uncertain knowledge. As stock market prediction involves imprecise concepts and imprecise reasoning, fuzzy logic is a natural choice for knowledge representation. Fuzzy logic, founded by Zadeh (1965), is formalism for reasoning with vague knowledge. Hiemstra (1994) presents a forecasting support system that uses a fuzzy logic model for the prediction of quarterly stock market excess returns. The excess return is the market return minus the risk-free rate of return and is important for asset allocation, the diversification of an investment portfolio among the diversification of an investment portfolio among asset classes like stocks and cash. The basic model (Hiemstra, 1994) shows acceptable performance, however, given the great number of design parameters, tuning is too complex to follow a trial and error approach.

Pellizzari and Pizzi (1997) develop a fuzzy local approach to model and forecast time series. The method appears to be flexible both in modeling nonlinearities and in coping with weak non-stationarities. They estimate local linear approximation (LLA) by a fuzzy weighted regression and test the model on data from a simulated noisy chaotic map and on two real financial time series, namely FIAT daily stock returns and USD-LIT exchange return rates. The LLA produces very accurate forecasts and is able to identify the correct order of the chaotic map (Pellizzari and Pizzi, 1997). However, LLA might be more difficult to model when dealing large financial datasets because of structural changes and other irregularities.

Modeling and forecasting financial time series are challenging subjects, both because of their intrinsic complex behavior and as classical global models appear quite often to be unsatisfactory. One of the reasons of poor performance of global models is the fact that different parts of financial time series exhibit different degrees of nonlinearity or non-stationary. This amounts to say that the data generating process is smoothly changing with time. How to overcome some of the limits of classical modeling

approaches to get better fit and forecasts? It might be possible that a local approach might better cope with non-stationary and nonlinearity possibly present in financial data. Some kind of stationary behavior is of course necessary to estimate meaningful parameters and suppose that the data generating process is locally stable in some sense.

Complexity and vagueness of above issues lead us to consider some elements of fuzzy logic. Moreover, a fuzzy approach need not assume restrictive hypothesis on the underlying data generating process and can be intuitively interpreted in terms of similarity concepts. If desired, this technique can be embedded in a statistical framework to obtain sound probabilistic results (Pellizzari and Pizzi, 1997). Trying to get the best from the above ideas, WFPRs approach has been used in this project.

2.5.6 Classification

Building accurate and efficient classifiers for large databases is one of the essential tasks of data mining and machine learning research. Given a set of cases with class labels as training set, classification is to build a model (classifier) to predict future data objects for which the class label is unknown. Common classification methods are Apriori-like approach (Agrawal, 1994; Klemettinen *et al*., 1994), FP-growth (Han and Pei 2000; Xu *et al*., 2002), Naive Bayes Classifier (Huang and Hsu, 2002; Tang *et al*., 2002).

Most of the previous studies adopt an Apriori-like approach, whose essential idea is to iteratively generate the set of candidate patterns of length ($fp + 1$) from the set of frequent patterns of length fp (for $fp \geq 1$), and check their corresponding occurrence frequencies in the database. An important heuristic adopted in these methods, called Apriori heuristic (Agrawal, 1994; Klemettinen *et al*., 1994), which may greatly reduce the size of candidate pattern set, is the anti-monotonicity property of frequent sets (Agrawal, 1994; Klemettinen *et al*., 1994): if any length k pattern is not frequent in the database, its length ($fp + 1$) super-pattern can never be frequent. The Apriori heuristic

achieves good performance gain by (possibly significantly) reducing the size of candidate sets. Recently another classification approach have been introduced called FP-tree (Han and Pei 2000; Xu *et al*., 2002). Efficiency of mining is achieved by three phases: 1) A large database is compressed into a highly condensed, much smaller data structure, which avoids costly, repeated database scans, 2) FP-tree-based mining adopts a pattern fragment growth method to avoid the costly generation of a large number of candidate sets, and 3) a partitioning-based divide-and-conquer method is used to dramatically reduce the search space.

Nevertheless, these proposed approaches (Agrawal, 1994; Klemettinen *et al*., 1994; Han and Pei, 2000; Xu *et al*., 2002) may still encounter some difficulties in different cases. First huge space is required to serve the mining. Second real databases contain all the cases. The only particular sets of cases cannot be selected, for example, telecommunication data analysis, market basket analysis, census data analysis, classification and predictive modeling, etc. It is hard to select an appropriate mining method on the fly if no algorithm fits all. Third, large applications need more scalability.

Some researchers investigate that classification is a useful data mining technique for stock market prediction (Mark *et al*., 2000). In recent years, there have been a growing number of studies looking at the direction or trend of movements of various kinds of financial instruments (such as Maberly, 1986; Wu and Zhang, 1997; O'Connor *et al*., 1997). However, none of these studies provide a comparative evaluation of different classification techniques regarding their ability to predict the sign of the index return. Given this notion, Mark *et al*. (2000) examine various financial forecasting models based on multivariate classification techniques and compare them with a number of parametric and nonparametric models that forecast the level of the return.

The geometrical meaning of M-P's (McCulloch-Pitts) neuron model indicates that classifying samples according to the requirements by constructing neural networks is equal to finding a collection of domain with which vectors of the preset sample sets are partitioned (Zhang *et al*., 2002). But in some applications, such as time queue

forecasting including stock share forecasting, because the preset sample sets may contain some exceptions and erroneous results, it is desirable to introduce some self-adjusting and probabilistic decision making mechanism for the covering algorithm. Based on this method, Zhang *et al*. (2002) developed a self-adjusting and probabilistic decision-making classifier and applied the software package to forecast the share index of Shanghai stock market. However, the algorithm still needs to improve the following three points: 1) to optimize further and divide sample set reasonably and construct covering domains that the feature can be drawn, thus concludes the space point of representative models characteristic, 2) to define the decrease values and increase values (being the equivalent at present) of the confidence coefficients with regard to the characteristic point and the exceptive point, so the refusal point can be recognized with the characteristic point, 3) to optimize the algorithm and reduce the learning time for making the algorithm to be practical.

2.6 Decision Tree Classification

In classification, a set of data is analyzed and a set of grouping rules is developed to classify future data. This approach is especially useful when there are many patterns in the data and it is difficult to determine a typical pattern for the entire set of data. Decision trees (Quinlan, 1986; Yeung *et al*., 2002) have been found very effective for classification of huge and frequently modifiable databases e.g., stock market, shopping mall, etc. Decision trees are analytical tools used to discover rules and relationships by systematically breaking down and subdividing the information contained in data set. There are two basic types of decision tree classification: crisp decision tree (Agarwal *et al*., 2001; Quinlan, 1986; Quinlan, 1993) and fuzzy decision tree (Umanol *et al*., 1994; Wang *et al*., 2001; Yeung *et al*., 2002). These decision trees for classification and prediction are discussed in the following subsection.

2.6.1 Crisp Decision Tree

ID3 learning algorithm is a popular example of crisp decision tree proposed by Quinlan (1986) that uses maximum information gain to select expanded attributes among the candidate attributes at each step while it grows the tree. In classification, a set of data is analyzed and a set of grouping rules is developed to classify future data. This approach is especially useful when there are many patterns in the data and it is difficult to determine a typical pattern for the entire set of data. Crisp decision trees are analytical tools used to discover rules and relationships by systematically breaking down and subdividing the information contained in data set. Crisp decision trees are composed of a hierarchy of "if-then" statements and they are appropriate for categorical and interval data. Tree Projection (Agarwal *et al*., 2001) and ID3 (Quinlan, 1986; Quinlan, 1993) are popular approaches in decision tree classification.

The Tree-Projection algorithm proposed by Agarwal *et al*. (2001) recently is an interesting algorithm, which constructs a lexicographical tree and projects a large database into a set of reduced, item based sub-databases based on the frequent patterns mined so far. Tree-Projection may still encounter difficulties at computing matrices when the database is huge. The study in Tree-Projection (Agarwal *et al*., 2001) has developed some smart memory caching methods to overcome this problem. However, it could be wise not to generate such huge matrices at all instead of finding some smart caching techniques to reduce the cost (Han and Pei, 2000). Moreover, even if the matrix can be cached efficiently, its computation still involves some nontrivial overhead. And, also when there are many long transactions containing numerous frequent items, transaction projection becomes a nontrivial cost of Tree-Projection.

Inductive decision tree learning methods, ID3 and its successors C4.5 and C5.0 due to Quinlan (1996), are very popular and widely used for the last two decades. Before the last decade, only categorical information has been considered. With the immanent need to handle numerical information too, various methods to create discrete partitions (and accordingly predictions) for numerical attributes have been invented (Zeidler and

Schlosser, 1996; Quinlan, 1996). The problem that arises when this discretization is done by means of partitions into crisp sets (intervals) is that small variations (e.g., noise) on the input side can cause large changes on the output. When a numerical output is computed, this leads to a discontinuous and instable behavior of the output function. This entails the demands for admitting vagueness in the assignment of samples to predicates.

2.6.2 Fuzzy Decision Tree

There have been many methods for constructing decision trees from collection of crisp examples (Quinlan, 1986; Zeidler and Schlosser, 1996; Quinlan, 1993; Han and Pei, 2000; Agarwal *et al*., 2001). The decision trees generated by these methods are useful in building knowledge-based expert systems. Due to the rapid growth of uncertainty in the knowledge-based systems, it is found that using crisp decision trees alone to acquire imprecise knowledge is not enough. Uncertainty such as fuzziness and ambiguity should be incorporated into the process of learning from examples such as decision tree induction. These decision tree induction techniques introduce fuzzy decision tree generation suggested by many authors (Umanol *et al*., 1994; Wang *et al*., 2001; Dong and Kothari, 2001; Mitra *et al*., 2001; Yeung *et al*., 2002). The fuzzy decision tree with minimal number of leaf-nodes is usually thought to be optimal. However, the optimal (fuzzy) decision tree generation has been proved to be NP-hard. Therefore, the research on heuristic algorithms is necessary to mine knowledge from hidden patterns from huge databases. The heuristic information used in constructing fuzzy decision trees can be various and each heuristic may be better than the other in some aspects. Mainly three heuristics are popular for generating fuzzy decision trees among the existing ones. These three heuristics are based on

1. classification using information-entropy to select expanded attributes (Umanol *et al*., 1994; Dong and Kothari, 2001).

2. classification using ambiguity to select expanded attributes (Yuan and Shaw, 1995)
3. classification using importance of attribute contributing to its consequents to select the expanded attributes to select expanded attributes (Wang *et al*., 2001)

One powerful technique for generating crisp decision trees called ID3. Quinlan proposed the earlier version of ID3, which is based on minimum classification information-entropy to select expanded attributes, in (Quinlan, 1986). As the increasing uncertainty incorporated into the knowledge-based system, the fuzzy version of ID3 has been suggestion by several authors (for instance, Umanol *et al*., 1994; Dong and Kothari, 2001). Classification information-entropy based on probabilistic models, i.e., Shannon Entropy, is a well-known concept to describing probabilistic distribution and uncertainty. Subsequently, this concept was extending to describe the possibilistic distribution uncertainty, called fuzzy entropy. The typical extension was given in (Yuan and Shaw 1995). Yuan and Shaw (1995) refers a possibilistic distribution to a vector whose components are in [0, 1] while a probabilistic distribution is possibility distribution with the property that the sum of all components is equal to 1.

For a probabilistic distribution, each component considered as a probability with which the corresponding event occurs. A possibilistic distribution usually considered as a fuzzy set vector), and each component of the vector, i.e., the membership degree regarded as the possibility with which the corresponding event occurs. For the difference and consistency between probability and possibility, one can refer to (Zadeh, 1999). The difference between the uncertainties described by the entropy of a probabilistic distribution and described by the fuzzy entropy of a possibilistic distribution is that the former attains its maximum at all components being 0.5 but the latter does not. Fuzzy ID3 uses the fuzzy the entropy of a possibilistic distribution.

Another existing powerful heuristic algorithm to generate fuzzy decision tree was introduced by Yuan and Shaw (1995). Instead of using minimum fuzzy entropy, this

heuristic (Yuan and Shaw, 1995) used the minimum classification ambiguity to select expanded attributes. The classification ambiguity is called non-specificity (also called U-uncertainty). Recently, Wang *et al*. (2001) proposed another heuristic, which uses the maximum classification importance of attribute contributing to its consequents to select the expanded attributes. This concept firstly proposed by Pawlak (1991) while investigating the reduction of knowledge. It was used to extract the minimum indispensable part of equivalent relations. Wang *et al*. (2001) extend this concept to a fuzzy case and then used it to select the expanded attribute at a considered node while generating fuzzy decision trees. In Wang *et al*. (2001) method, aims to search for an attribute that its average degree of importance contributing to the classification attains maximum, i.e., selecting such an integer k_o (the $k_o{}^{th}$ attribute) that $P_{k_o} = \max_{1 \le k \le n} P_k$.

Proposition 1: For fixed k and i, consider two functions

$$Entr_i^{(k)} = -\sum_{j=1}^{m} p_{ij}^{(k)} \log_2 p_{ij}^{(k)} \tag{2.1}$$

$$Ambig_i^{(k)} = \sum_{j=1}^{m} (\pi_{ij}^{(k)} - \pi_{i,j+1}^{(k)}) \tag{2.2}$$

Where $Entr_i^{(k)}$ and $Ambig_i^{(k)}$ show the classification information-entropy and classification ambiguity for each (k) respectively. $p_{ij}^{(k)}$ shows the probability of classification attributes. In j within the area $\{0 \le p_{ij}^{(k)} \le 1 \mid j = 1,2,...m\}$, the first function attains its minimum at a vector of which each component is either 0 or 1, and the second attains its minimum at a vector in which one component is 1 but the other components are 0. Here make the appointment $0 \log_2 0 = \lim_{x \to 0} (x \log_2 x) = 0$. In which $(\pi_{i1}^{(k)}, \pi_{i2}^{(k)}, \ldots\ldots \pi_{im}^{(k)})$ with descending order $\pi_{i,m+1}^{(k)} = 0$ is a permutation of $(\tau_{i1}^{(k)}, \tau_{i2}^{(k)}, \ldots\ldots, \tau_{im}^{(k)})$ which is a normalization of $(p_{i1}^{(k)}, p_{i2}^{(k)}, \ldots\ldots, p_{im}^{(k)})$, i.e., $\tau_{ij}^{(k)} = p_{ij}^{(k)} / \max_j p_{ij}^{(k)}$. (for proof see appendix C)

This proposition indicates that fuzzy ID3 aims averagely to search the expanded attribute with relative frequencies as close to 0 or 1 as possible while Yuan and Shaw's method aims averagely to search one with relative frequencies as close to 0 (except for the maximum frequency) as possible.

It is easy to see from Proposition 1 that the minimum of the function $Ambig_i^{(k)}$ implies the minimum of the function $Entr_i^{(k)}$ and the inverse is invalid. Particularly, if $\{0 \le p_{ij}^{(k)} \le 1 \mid j = 1,2,...m\}$ is a probabilistic distribution then the two minima are equivalent.

Proposition 2: From proposition 1, function $Entr_i^{(k)}$ and $Ambig_i^{(k)}$ attains their maxima at $p_{i1}^{(k)} = ... = p_{im}^{(k)} = e^{-1}$ and $p_{i1}^{(k)} = ... = p_{im}^{(k)} = 1$ respectively (for proof see appendix C).

Proposition 1 implies that when all frequencies are 1 the fuzzy entropy is 0 but the non-specificity attains maximum. That indicates such a situation in which using fuzzy ID3 techniques, Yuan and Shaw (1995) select different expanded attributes. However, proposition 1 indicates that if Yuan and Shaw's technique selects an expanded attributes with very small value of $Ambig_i^{(k)}$, then fuzzy ID3 select the same expanded attribute at the same non-leaf node. Moreover, for the frequency distribution the smaller the non-specificity, the closer it is to a probabilistic distribution. Thus proposition 1 intuitively indicates that the two techniques are likely to select the same expanded attribute while the non-specificity is small. Particularly, if the two techniques select the same expanded attribute at the root with small fuzzy entropy and non-specificity, then the two techniques for selecting expanded attributes are gradually consistent. That is the expanded attribute selection of fuzzy ID3 techniques, to some extent is identical to the one of Yuan and Shaw's technique. These two techniques show the expanded attributes of the two heuristics are the same at most non-leaf nodes.

In Wang *et al*. (2001) and Yeung *et al*. (2002) fuzzy decision tree, which is based on the maximum degree of importance of attribute contributing to the fuzzy classification. It aims, on the considered node with several attributes to be chosen to select an attribute whose contribution to classification is maximal.

Proposition 3*:* Under an assumption of uniform distribution, either maximum or minimum degree of importance implies maximum fuzzy entropy when the classification is crisp and implies maximum non-specificity when the classification is fuzzy. The uniform distribution assumption is formulated in the proof (for proof see appendix C).

Proposition 3 indicated that the relation between (Wang *et al*., 2001; Yeung *et al*., 2002) FDT and the other is very complicated. It implicitly proposes that there exists such an attribute at which the maximum (minimum respectively) degree of importance and maximum (minimum respectively) entropy can be achieved simultaneously at a node.

These three heuristic has some strength and weakness, Table 2.5 presents the summary of comparative results in terms of complexity, applicability, comprehensibility, learning accuracy, handling of classification ambiguity, robustness and scalability. With regard to the complexity of the fuzzy decision tree, the relation among the three heuristics is non-deterministic, dependent mainly on the expanded attribute selection. First consider the computation effort while expanding a non-leaf node and then consider the size of trees. The number of leaves is an important index to measure the size of a tree. Obviously bigger numbers of leaves are creating more complexity in construction of FDT. While expanding a non-leaf node and then consider the size of trees. The following assertion is valid:

$$\text{CE (fuzzy ID3)} \approx \text{CE (Yuan and Shaw's method)} \leq \text{CE (Wang method)}$$

where CE represents the term, Computation-Efforts, that refers mainly to the number of times of operations such as addition, multiplication, max, min, etc.

Table 2.5: Summary of analytic comparison of fuzzy decision trees

	Fuzzy ID3 (1994)	**Yuan & Shaw (1995)**	**Wang *et al*. (2001)**
Heuristic Information	Fuzzy entropy	Non-specificity of possibility distribution	Importance of attributes contributing to classification
Criterion of expanded attribute	Minimum fuzzy entropy	Minimum non-specificity	Maximum importance degree
Expanded attribute used in the tree	Partially same as Yuan and Shaw's heuristic	Partially same as fuzzy ID3 heuristic	Not same as others
Reasoning mechanism	Mini-maxi operation of memberships	Multiplication-addition operation of memberships	Weighted average of similarity
Comprehensibility of tree	Lower	Higher	Medium
Reasoning accuracy	Medium	Less	Greater
Complexity (time, space)	Same as Yuan and Shaw's	Same as Fuzzy ID3	Greater than both
Robustness	Medium	Greater	Less
Scalability	Less	Medium	Greater

The number of leaves is an important index to measure the size of a tree. The generic standard of leaf-node is a frequency-threshold, which is node (fuzzy set) regarded as a leaf if the relative frequency of some class at the node exceeds a given threshold. Fuzzy rules extracted from Yuan and Shaw's tree includes only one parameter CF, one can see that the comprehensibility of Yuan and Shaw's tree is better than that of Wang *et al*., tree which is turn better than that of fuzzy ID3, that is:

Comprehensibility (fuzzy ID3) $\leq$ Comprehensibility (Wang *et al*., tree)
$\leq$ Comprehensibility (Yuan and Shaw's)

2.7 Fuzzy Reasoning Methods

Fuzzy knowledge representation schemes such as the fuzzy production rule (FPR) are usually in a form of fuzzy IF-THEN rule in which fuzzy labels like "tall" or "short" in the antecedent and the consequent part are allowed. If the given fact for an antecedent in a FPR does not match exactly with the antecedent of the rule, the consequent part still is drawn by technique such as fuzzy reasoning. Many existing fuzzy reasoning methods are based on Zadeh's Computational Rule of Inference (CRI) (Zadeh, 1973), which requires setting up a fuzzy relation between the antecedent and the consequent part. Despite their success in various rule-based system applications, Zadeh's CRI has been criticized to be too complex and its underlying semantic to be unclear (Turksen and Zhong, 1989, 1990). Therefore, similarity-based methods (Turksen and Zhong, 1990; Chen, 1994; Yeung and Tsang, 1997b) and linear revising methods (Ding *et al*., 1989; Ding *et al*., 1992; Mukaidono *et al*., 1990) have also been proposed for reasoning.

Among them, the similarity-based fuzzy reasoning methods, which make use of the degree of similarity between a given fact and the antecedent of the rule to draw the conclusion, are well known.

The notation for FPR with single antecedent is as follows:

R: IF A THEN C, ($CF = \mu$), Th, W e.g., if a_1 then C, ($CF = \mu$), $Th = \{\lambda_{a_1}\}$, $W = \{w_1\}$. C is represented as a "concluded disorder" in Chen's Diagnosis problem (Chen, 1988; Chen, 1994) or a consequent in other problems.

Given four cases of facts with single antecedent:

Case 1: $A' = A$

Case 2: $A' =$ very A

Case 3: $A' =$ more or less A

Case 4: $A' =$ not A

What conclusion C' can be drawn? In fuzzy reasoning methods these possible four cases of facts have been considered in (Turksen and Zhong, 1990; Chen, 1994; Yeung and Tsang, 1994; Yeung and Tsang, 1997a; Yeung and Tsang, 1997b).

There are many ways to automatically acquire imprecise knowledge. Induction fuzzy decision trees are one of them. This method first generates a fuzzy decision tree and then extracts a set of fuzzy production rules from the tree. Since a fuzzy decision tree generation needs a heuristic to select expanded attributes, most researchers on fuzzy decision trees focus on the selection of expanded attributes by using a set of examples with same type of attributes (Chang and Pavlidis, 1977; Wang and Hong, 1998). So far, the heuristic used in fuzzy decision tree generation is entropy-based. Although the entropy-based heuristic has been proven to be correct for classification in information theory and can generate a relatively small tree, it cannot learn weighted fuzzy production rules which have been considered to be useful and important for representing complex knowledge (Yeung and Tsang, 1998). It is necessary to propose an alternative heuristic that help us to generate a set of weighted fuzzy production rules with the required knowledge parameters from data with continuous mixed attributes. In the following subsections, the existing fuzzy reasoning mechanisms are recalled. The advantage of these methods is measured to check: 1) the ability to assign certainty factors, threshold values and weights to FPRs; and 2) the similarity computation.

2.7.1 Turksen *et al.*'s Approximate Analogical Reasoning Schema (AARS)

According to Turksen and Zhong (1989; 1990) the FPR manipulated by AARS has a threshold value $\lambda_{\circ}$ but no certainty factor is assigned to it. Furthermore, each antecedent of a composite FPR does not have a weight assigned to it. Rules of different type have been considered: a simple rule, a composite conjunctive rule and a composite disjunctive rule and a composite disjunctive rule.

Table 2.6: FPRs with a single antecedent

Value of A'	Turksen *et al.*'s AARS Method (1990)	Chen *et al.*'s Method (1994)	Yeung *et al.*'s EC Method (1997)
A	C	C	C
Very A	very C	Very C	very C
more or less A	more or less C	more or less C	more or less C
not A	unknown	Unknown	Unknown

Table 2.7: FPRs with a multiple antecedents

Value of A'	Turksen *et al.*'s AARS Method (1990)	Chen *et al.*'s Method (1994)	Yeung *et al.*'s EC Method (1997)
A	C	C	C
Very A	very C	Very C	very C
more or less A	more or less C	more or less C	more or less C
not A	rule misfiring is possible	rule misfiring is possible	rule misfiring is possible

The fuzzy reasoning method used to draw conclusion C' is rather complex because all the distance measure based on the Euclidean distance has to be computed. The similarity measure is then applied and when $S_{AARS} \geq \lambda_{\circ}$ one of the modification functions is used to compute the consequent. On the other hand, the method can generate more specific conclusion as one can see from Table 2.6 and Table 2.7 that the conclusion drawn for C' is "very C" when the given fact A' is " very A". Despite the adoption of Euclidean distance measure, the drawn consequent is accurate because two modification functions are proposed to compensate for the undesirable commutative property of similarity measure based on the Euclidean distance measure. However, rule misfiring may result if a single threshold value is assigned to the entire rule. In order to

achieve reasonable conclusion drawn using similarity-based reasoning methods, similarity measure function must be selected with care.

The similarity measure adopted by Turksen *et al*. in (Turksen and Zhong, 1989, 1990) is based on the Euclidean distance measure. It is widely acceptable measure but may not be suitable for comparing similarity between two fuzzy sets. The order of input parameters for this measure is irrelevant and the method is applicable to multi-level reasoning.

2.7.2 Chen's Function T (FT) Method

In (Chen, 1988; Chen, 1994) weight is assigned to each proposition in the antecedent of a FPR designed for medical diagnosis application. Threshold value and certainty factor are also assigned to a FPR. Rules considered consist of multiple propositions connected by "AND" in the antecedent. Composite disjunctive rule has not been mentioned here.

The similarity measure is based on the absolute difference between the elements of two vectors. The assigned weight is calculated by familiar statistical technique such as weighted mean. So the method used to draw conclusion is simple. The value of the conclusion reflects the degree of confirmation that a patient might have a disease C. Summary can be found in Table 2.6 and Table 2.7.

The accuracy of the conclusion is considered to be reasonable if the fuzzy value of the consequent C as well as the strength of confirmation of C is expressed by a real number lying between 0 and 1. The universe of discourse consists of elements of many symptom features used in medical diagnosis. However rule misfiring may result. The similarity measure function is very simple and straightforward which is considered to be acceptable. The order of the input parameters is also irrelevant. For the medical diagnostic problem, the method is applied to one-level reasoning only.

2.7.3 Yeung *et al.*'s Equality and Cardinality (EC) Method

Different kinds of fuzzy production rules can be handled by this method (Yeung and Tsang, 1997b). Weight and threshold value assigned to each proposition in the antecedent are considered. Certainty factor is used in calculating the consequent. This method draws the conclusion by using two enhanced modification functions. It is the most complex one among the above mentioned fuzzy reasoning methods because the absolute distance difference between two fuzzy sets must be computed and an enhanced modification function as well as the square root of the similarity measure and certainty factor has to be considered in calculating the consequent C. It gives more specific results than the other methods proposed by (Yeung and Tsang, 1994; Yeung and Tsang, 1997b).

The value of the conclusion drawn is accurate because two modification functions are used to draw meaningful consequent and it reduces or prevents rule misfiring by requiring each similarity measure to be compared with a threshold value before an overall similarity measure is obtained. This similarity measure leads to reasonable result for two fuzzy sets because the sums of the absolute differences of the membership values of two fuzzy sets and the cardinality of the fuzzy set in the antecedent are taken into consideration. The order of input parameters must be followed. Furthermore, the method can be applied to multi-level reasoning. All the above analysis results have been summarized in Table 2.8.

ARRS method (Turksen and Zhong, 1989; Turksen and Zhong, 1990) gives the complex results, but the accuracy of similarity measure and multi-level reasoning capabilities are reasonable. Also weight and certainty factors cannot be assigned to each proposition in antecedent however weight and certainty factors are very important in rule firing/miss firing (Yeung *et al*., 1997b). Accuracy of drawn conclusion and accuracy of similarity measure is reasonable in MF method (Chen, 1988; Chen, 1994) but good in assigning weight and certainty factors to whole rule. EC methods by Yeung *et al*. (1997) are accurate in drawing conclusion and in similarity measures and also

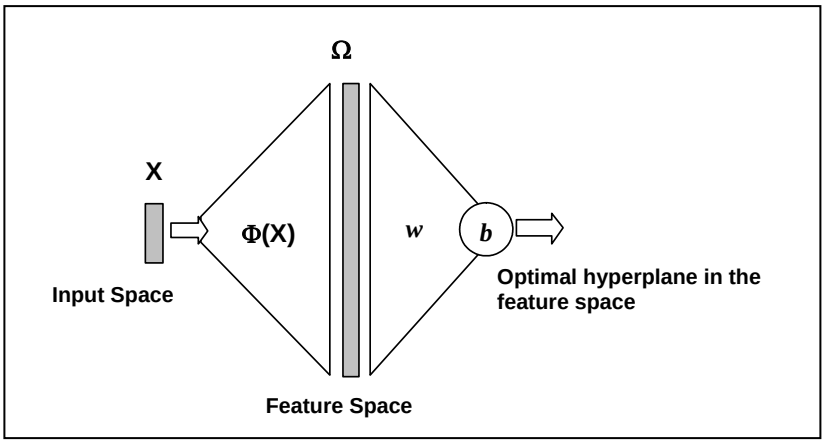
these methods have capability to multilevel reasoning. These methods still need to improve in careful weight and certainty factor assigned to each proposition in antecedent. In addition, these methods give very complex results in drawing conclusion.

Table 2.8: Analysis of existing similarity-based methods

Properties	Weight assignments	Certainty factor	Threshold value	Input parameters	Complexity in drawing conclusion	Accuracy of drawn conclusion	Accuracy of Similarity	Multi-level reasoning capability
Turksen *et al.*'s AARS (1990)	No	No	Whole antecedent	No order	Complex	Accurate	Acceptable	Yes
Chen's MF (1994)	No	Whole rule	Whole antecedent	No order	Simple	Reasonable	Reasonable	No
Yeung *et al.* 's EC (1997)	Yes	Whole rule	Each proposition in antecedent	Has order	Complex	Accurate	Reasonable	Yes

2.8 Support Vector Machine

SVMs were first suggested by Vapnik in the 1960s for classification and have recently become an area of intense research owing to developments in the techniques and theory coupled with extensions to regression and density estimation. SVMs deliver the state of the art performance in real world applications such as text categorization, hand-written character recognition, image classification, financial forecasting and so on (Bao *et al*., 2002). The support vector machine (SVM) is a training algorithm for learning classification and regression rules from data, for example the SVM can be used to learn polynomial, radial basis function (RBF) and multi-layer perceptron (MLP) classifiers (Osuna, *et al.*, 1997).



- Vector **X** is mapped into a high-dimensional feature space
- After the transformation the optimal separating hyperplane is to be built in the high dimensional space **Ω**

Figure 2.3: Support vector machine.

The theory of statistical learning founded by Vapnik (1998) serves as the basis of support vector machine (SVM). An interesting point about SVM is that it has been successfully applied to a number of applications ranging from time series prediction (Lu *et al.*, 2002; Raicharoen and Lursinsap 2003) to cancer detection (Kong *et al.*, 2004), to storm cell classification (Ramirez *et al.*, 2001) and image retrieval (Tong and Chang, 2001). The motivations behind the introduction of SVM are due to its capability to improve the generalization performance of neural network and can achieve global minimum (Lu *et al.*, 2002).

Initially developed for solving classification problems, SV technology can also be successfully applied in regression, for example functional approximation problems. Unlike pattern recognition problem, where desired outputs are discrete values like Booleans, here there are real-valued functions (Cristianini and Taylor, 2000). The benefits of the SVM in regression (also known as a support vector regression; SVR) lies in not assuming that there is a probability density function (pdf) over the return series

and it adjusts the parameters relying on the empirical risk minimization inductive principle (Vapnik, 1998).

Support vector machine is a new machine-learning paradigm that works by finding an optimal hyperplane as to solve the learning problems (Evgeniou and Pontil, 2001). Originally SVM were developed for pattern recognition problems until recently SVM have been implemented in non-linear regression estimation and time series prediction with the introduction of ε-insensitive loss function. (Lu *et al.*, 2002; Muller *et al.*, 1997; Mukherjee *et al.*, 1997).

In the study done by Chen *et al.* (2004), they implemented a support vector machine model by introducing a support vector regression for load forecasting and the performance of SVM is then compared with other techniques such as local model and neural network models. Based on the result, SVM outperformed other techniques by producing an overall lower mean absolute percentage error (MAPE) of less that 4% on different data preparation as compared to local model with 8.81% MAPE. Neural network model on the other hand, produced inconsistent MAPE that range from as high as 7.8% to the lowest 3.63% due to difficulties in parameter selection.

In the study done by Vanajakshi and Rilett (2004), they have compared two machine learning techniques performance in predicting traffic speed for intelligent transportation system (ITS). The two chosen techniques are Artificial Neural Network and Support Vector Machine. In their research, they try to investigate the ability of both techniques in predicting traffic variables such as speed, travel time or flow based on real time data and historic data, collected by various systems in transportations networks. For the SVM, they selected support vector regression for the prediction of speed with a RBF kernel function and the value of insensitivity function ε is selected to be 0.5 using trial and error processes.

Table 2.9: Advantages of SVM.

Area	Advantages
Bioactivity prediction and compound classification (Rahayu, 2004)	• Better prediction on unseen test data • Offers a unique optimal solution for a training problem • Number of free parameters are less as compared to other methods • Performed well with data that have large number of features.
Forecasting short-Term stock price indexes (Bao and Shiou, 2002)	• SVM belongs to the family of structural risk minimization principle. • Offers few controlling parameters. • Provides global unique solution derived from a quadratic programming.
Time series prediction via least square SVM (Zhu *et al.*, 2002)	• Simple algorithm. • Fast convergence. • Good capability of prediction. • Do well in the high noise time series.
Air pollutant parameter forecasting using SVM (Lu *et al*., 2002)	• Achieve a good level of generalizations performance. • Absence of local minima. • Sparsity of solution. • Estimates a function by minimizing an upper bound on the generalizations error.

Based on their result, it is clearly shown that the proposed Support Vector Machine model using Support Vector Regression (SVR) is a viable alternative to Artificial neural network in short term prediction (Vanajakshi and Rilett, 2004). This is because ANN performance depends largely on the amount of data available for training the network while SVM method is independent of the training data.

SVM usually achieves higher generalization performance than traditional neural networks that implement the empirical risk minimization (ERM) principle in solving many machine learning problems (Cao and Francis, 2003). Another key characteristic of SVM is that training SVM is equivalent to solving a linearly constrained quadratic programming problem so that the solution of SVM is always unique and globally optimal.

Since SVM can be applied for classification and regression problems, in this section we will discuss the SVM for classification. However, the SVM for regression problem will be described in detail in chapter 6.

2.8.1 Support Vector Machine for Classification Problem

Support vector machines (SVM), having its roots in machine learning theory, utilize optimization tools that seek to identify a linear optimal separating hyperplane to discriminate any two classes of interest (Vapnik, 1995; 1998). SVMs are powerful tools for data classification (Cherkassky *et al.*, 1998; Burges, 1998). Classification is achieved by a linear or nonlinear separating surface in the input space of the dataset. The separating surface depends only on a subset of the original data. This subset of data, which is all that is needed to generate the separating surface, constitutes the set of support vectors.

In classification problems, SVM tries to place a linear boundary between two different classes and adjust it in such a way that the margin is maximized (Vanajakshi and Rilett, 2004). Moreover, in the case of linearly separable data, the method is to find the most suitable one among the hyperplanes that minimize the training error. After that, the boundary is adjusted such that the distance between the boundary and the nearest data points in each class is maximal. This can be seen in Figure 2.4 and 2.5 respectively.

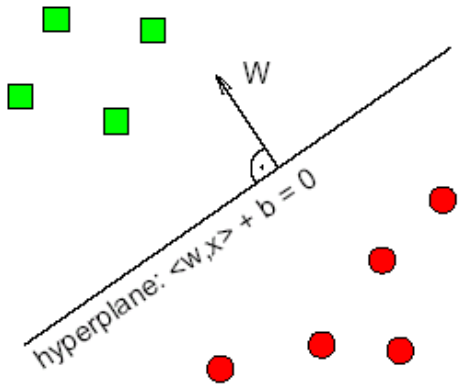


Figure 2.4: Hyperplane = { x | <w,x> + b = 0 }.

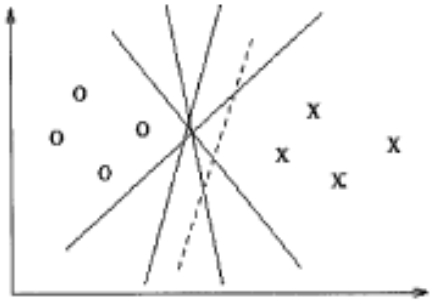


Figure 2.5: Separating hyperplanes.

The chosen boundary is placed in the middle of the margin between two data points as shown in Figure 2.4. The points nearest to the separating hyperplane are called Support Vectors. Only they determine the position of the hyperplane. All other points have no influence. The weighted sum of the Support Vectors is the normal vector of the hyperplane.

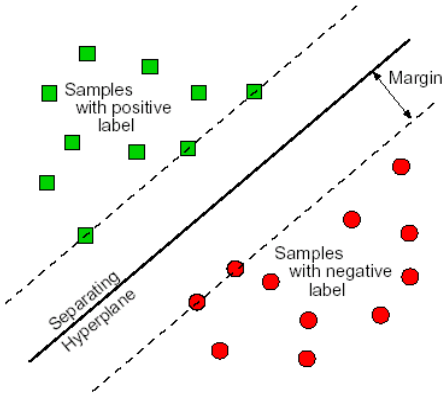


Figure 2.6 Support vector with maximum margin boundary.

2.8.1.1 Calculation for SVM Classification

Consider a binary classification problem, where the given dataset is partitioned into two classes with a linear hyperplane separating them (Figure 2.7). Assume that the training dataset consists of k training samples represented by $(x_1, y_1), \ldots (x_k, y_k)$, where $x_i \in \mathrm{R}^N$ is an N-dimensional data vector with each sample belonging to either of the two classes labeled as $y_i \in \{-1, +1\}$. The goal of SVMs is to find a linear decision function defined by $f(x) = w \cdot x + b$, where $w \in \mathrm{R}^N$ determines the orientation of a discriminating hyperplane, and $b \in \mathrm{R}$ is a bias. The hyperplanes for the two classes are, therefore, represented by $y_i (w \cdot x + b) \geq 1 - \xi_i$. Sometimes, due to the noise, variables $\xi_i > 0$, called slack variables, are used to account for the effects of misclassification. The optimal hyperplane (i.e., $f(x) = 0$) is located where the margin between two classes of interest is maximized and the error is minimized. This can be achieved by solving the following constrained optimization problem using Lagrange multipliers,

$$\min_{\alpha} \frac{1}{2} \sum_{i=1}^{k} \sum_{j=1}^{k} y_i y_j \alpha_i \alpha_j x_i \cdot x_j - \sum_{i=1}^{k} \alpha_i \tag{2.3}$$

such that $0 \leq \alpha_i \leq C$ and $\sum_{i=1}^{k} y_i \alpha_i = 0$

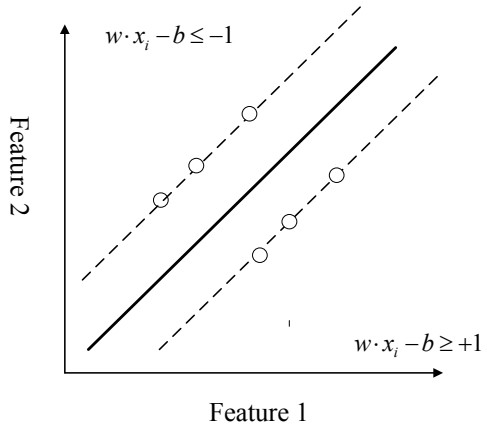


Figure 2.7: Illustration of an SVM construction in a two dimensional feature space. Dashed lines pass through the support vectors defined by data in circles forming the boundary of classes. *w*: slope of the linear decision boundary, *b*: intercept of the linear decision boundary.

The solution of the optimization problem given in (2.3) is obtained in terms of the Lagrange multi-pliers α_i. The multipliers that have nonzero values which lie on the two parallel hyper-planes are called the *support vectors*.

Given an optimal solution

$$w^0 = \sum_{i=1}^{l} y_i \alpha_i^0 x_i$$

$$b^o = \frac{1}{2}\left[w^o \cdot x^o(1) + w^o \cdot x^o(-1)\right]$$

The decision rule is then applied to classify the dataset into two classes viz. +1 and –1

$$f(x) = sign\left(\sum_{\text{support vectors}} y_i \alpha_i{}^o (x_i \cdot x_j) - b^o\right) \tag{2.4}$$

where *sign*(·) is the signum function. It returns +1 if the element is greater than or equal to zero and –1 if it is less than zero.

There are instances where a linear hyperplane cannot separate classes without misclassification, relevant to our problem domain; however, those classes can be separated by a nonlinear separating hyperplane. In this case, data may be mapped to a higher dimensional space with a nonlinear transformation function. In the higher dimensional space, data are spread out, and a linear separating hyperplane may be found. This concept is based on Cover's theorem on the separability of patterns. According to Cover's theorem on the separability of patterns, an input space made up of nonlinearly separable patterns may be transformed into a feature space where the patterns are linearly separable with high probability, provided the transformation is nonlinear and the dimensionality of the feature space is high enough. Figures 2.8 and 2.9 illustrates that two classes in the input space may not be separated by a linear separating hyperplane. However, when the two classes are mapped by a nonlinear transformation function, a linear separating hyperplane can be found in the higher dimensional feature space.

Let a nonlinear transformation function Φ maps the data into a higher dimensional space. Suppose there exists a function *K*, called a kernel function, such that,

$$K(x_i, x_j) = \Phi(x_i) \cdot \Phi(x_j)$$

A kernel function is substituted for the dot product of the transformed vectors, and the explicit form of the transformation function Φ is not necessarily known. Further, the use of the kernel function is less computationally intensive. The formulation of the kernel function from the dot product is a special case of *Mercer's theorem* (Scholkopf and Smola, 2002). The optimization problem then becomes,

$$\min_{\alpha} \frac{1}{2} \sum_{i=1}^{k} \sum_{j=1}^{k} y_i y_j \alpha_i \alpha_j K(x_i, x_j) - \sum_{i=1}^{k} \alpha_i \tag{2.5}$$

The decision function becomes,

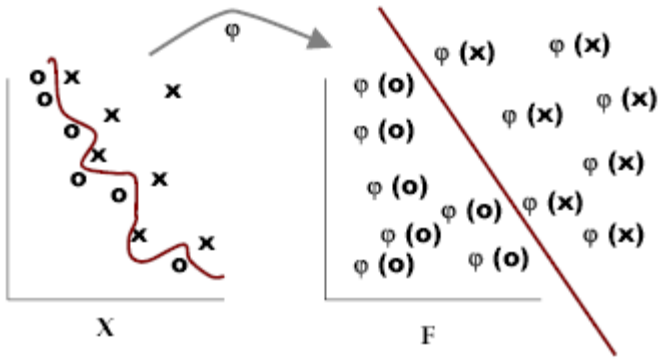


Figure 2.8: SVM input and feature space.

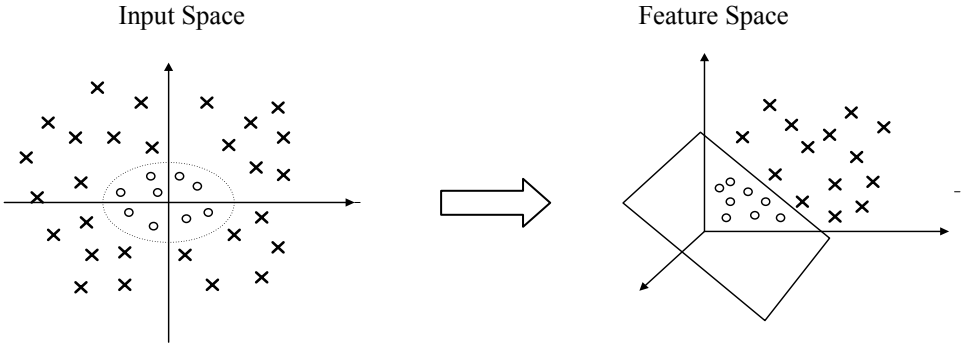


Figure 2.9: Mapping nonlinear data to a higher dimensional feature space where a linear separating hyperplane can be found. When mapped into a feature space via the non-lin $\Phi(x) = (z_1, z_2, z_3) = \left([x]_1^2, [x]_2^2, \sqrt{2}[x]_1[x]_2\right)$

$$f(x) = sign\left(\sum_{\text{support vectors}} y_i \alpha_i^{\,o} K(x_i, x_j) - b^o\right) \tag{2.6}$$

Examples of some well-known kernel functions are given in Table 2.10.

Table 2.10: Some well-known kernel functions

Polynomial	$K(x_i, x_j) =< x_i, x_j >^d$	d is a positive *integer*
Radial Basis Function (RBF)	$K(x_i, x_j) = \exp(-\|x_i - x_j\|^2 / 2\sigma^2)$	σ is a user defined value
Sigmoid	$K(x_i, x_j) = \tanh(\alpha < x_i, x_j > + \beta)$	α, β are user defined values

Consider an *l*-by-*D* data matrix (*X*) of examples. A (Mercer) kernel $K(x_i , x_j)$ implicitly projects the examples from *D*-dimensional input space into some (possibly higher-dimensional) feature space and returns their dot product in that feature space. That is, it computes

$$K(x_i, x_j) \equiv \phi(x_i) \cdot \phi(x_j) \equiv \phi(x_i)' \phi(x_j)$$

for some mapping function φ, but without explicitly computing the coordinates of the projected vectors. In this way, kernels allow large non-linear feature spaces to be explored while avoiding curse of dimensionality.

2.9 Support Vector Machine for Regression Problem

The application of Support vector machine to time-series forecasting, called support vector regression (SVR), has shown many breakthroughs and promising performance, such as forecasting of financial market (Yang *et al*., 2002), forecasting of Australian National Electricity market price (Sansom *et al.*, 2002), estimation of power consumption (Chen *et al.*, 2004), and travel time modeling (Wu *et al.*, 2004).

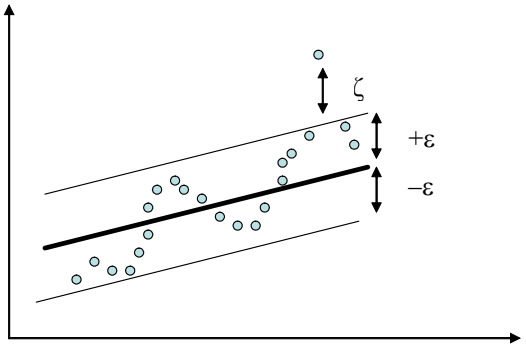


Figure 2.10: SVR to fit a tube with radius ε to the data and positive slack variables ζ measuring the points lying outside of the tube.

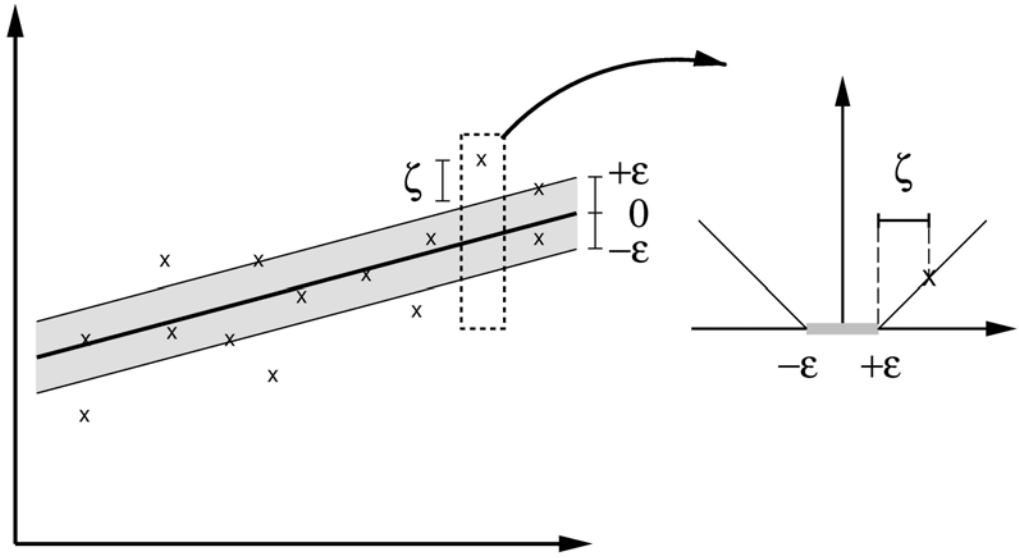


Figure 2.11: **ε**-insensitive regressor.

The basic idea of the regression problem is to determine a function that can approximate future values accurately. In the regression, the goal is to construct a hyperplane that lies close to as many of the data points as possible. Therefore, the objective is to choose a hyperplane with small norm while simultaneously minimizing the sum of the distances from the data points to the hyperplane (Theodore, 2000).

Therefore, SVR differs from SVM used in classification problem by introducing an alternative loss function that is modified to include a distance measure. Moreover, the parameters that control the regression quality are the cost of error $\dot{C}$, the width of

tube ε and the mapping function Φ. The Figures 2.10 and 2.11 show how SVR works with slack variable ς and the ε insensitive loss function. The slack variable ς is used to measure errors outside the ε tube.

The basic idea of SVR is to map the data into a high dimensional feature space via a nonlinear mapping and to do a linear regression in the space. The Support Vector method can also be applied to the case of regression, maintaining all the main features that characterize the maximal margin algorithm: a non-linear function is learned by a linear learning machine in a kernel-induced feature space while the capacity of the system is controlled by a parameter that does not depend on the dimensionality of the space (Cristianini and Taylor, 2000). Further detail on developing an SVR based system for financial forecasting is given in chapter 6.

2.10 Summary

In this chapter, some existing statistical and data mining techniques used for stock market prediction have been discussed. The comparative studies are also highlighted for the evaluation of different classification techniques regarding their ability to predict the sign of the index return. We also examine various financial forecasting models based on multivariate classification techniques and compare them with a number of parametric and nonparametric models that forecast the level of the return. In addition, some data mining techniques have been discussed: 1) to demonstrate and verify the predictability of stock index direction using classification models; 2) to compare the performance of various multivariate classification techniques relative to some econometric and artificial intelligence forecasting techniques; and 3) to develop effective trading strategies guided by the directional forecasts and to test the relative performance of these investment schemes.

CHAPTER 3

RESEARCH METHODOGLOGY

3.1 Introduction

In this project, for developing the intelligent knowledge discovery system we focus more on the behavior of the stock with time rather than on its value on a single day of the stock market. The time series data is converted into equal sets of patterns that are appropriate for general data mining methods and we use the attributes of equal sets of patterns as the basis for exploratory fuzzy production rules. This chapter presents the research operational framework. Operational framework can be divided into three phases. In problem background phase 1, we analyze the current system and find that existing techniques (i.e., data mining and statistical techniques) for prediction of time series stock market data are time consuming, complex, and give inaccurate predictive results. In the system development phase, we define the problem to solve. In the remaining steps of this phase, we collect the relevant data, clean the data, engineer the data to be maximally useful with the problem at hand, engineer a mining algorithm, run the mining algorithm and explain the results to the expert. In the third implementation and integration phase, we test the performance of data mining algorithms with the existing standard methods, if the results are not satisfied then repeat the process.

3.2 Operational Framework

This project is conducted according to the workflow process as illustrated in Figure 3.1. The operational framework is divided into three phases. The every phase describes the methods, steps and procedure for the proposed system. The following subsections present the illustration of these phases. The details steps involved in the development of the system are discussed in the next chapters.

3.2.1 Problem Formulation (Phase 1)

It seems odd to even mention the step of understanding and defining the problem but as a component of the whole process it can take a significant amount of time. In this project, problem formulation stage starts with the innovative idea and performs the extensive literature review and current system analysis. Due to tremendous increase of financial data, there has been a critical need for automated approaches to effective and efficient utilization of massive amount of financial data to support companies and individuals in strategic planning and in decision-making for investment. In this project, data mining techniques are considered different from other analytical processes in the sense that is deceptively easy to apply a data mining algorithm to a database and get some results, but without a clear understanding of the problem the result may be worthless. It is important to include this first step as a reminder of the importance of understanding the problem clearly. By defining the problem, it is meant that the data mining analyst should work with a financial expert to make the problem specific enough to be solvable and for the results to be quantifiable. As stated by the expert, the problem may be something that seems clear to him or her but could actually be solved in a number of different ways. Some solutions may be appropriate but others might be completely useless.

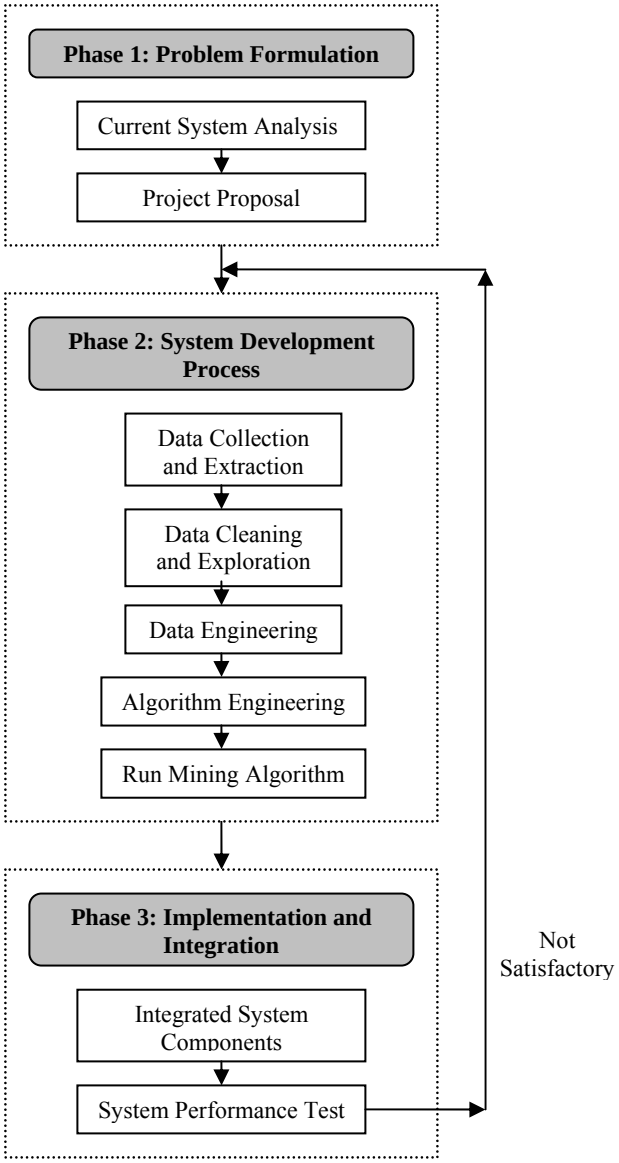


Figure 3.1: Research operational frame work

3.2.2 System Development Process (Phase 2)

System development phase helps to understand the data mining process for the prediction of stock market time series data. The methodology involved in the development of whole system can be divided into five major steps: data collection and extraction, data cleaning and exploration, data engineering, algorithm engineering,

running the data mining algorithm. These steps are discussed in the following subsections.

3.2.2.1 Data Collection and Extraction

Once a problem has been defined, relevant data must be collected. In most cases, the relevant data is extracted from an existing operational database or a data warehouse that was originally created to serve a variety of analytical needs. Usually, data mining algorithms cannot be run directly against a database with multiple tables. Thus, the data is extracted from a relational database and stored in some format that is accessed by the data mining algorithms. In these cases, the manual extraction step is really just an artifact of the inability of the mining algorithm to run directly against a database. In this research, the data is collected from three domains including KLSE, NYSE and LSE for the 100 different stocks and stored in text format. In these domains stock market returns data can be thought to fall into one of five categories as follows:

1. Five time series: index value at open, index value at close, highest index value, lowest index values and trading volume.
2. Fundamental factors: e.g., the price of gold, retail sales index, oil price, natural disaster, industrial production indices, foreign currency exchange rates.
3. Nominal factors: e.g., local stock market, affect of other companies, affect of political situations, affect of world situations.
4. Lagged returns from the time series of interest.
5. Technical factors: variables that are functions of one or more time series, e.g., moving average.

To model different types of stock market returns or exchange rates with continuous real time data, the proposed algorithms have been tested on historical data

from the KLSE, NYSE and LSE accumulated over 5-years of period. The set of 100 stocks from which system can build various portfolios some are as follows: AHP, Alcoa, American Brands, Arco, Coca Cola, Commercial Metals, Dow Chemicals, Dupont, Espey Manufacturing, Exxon, Fischbach, Ford, GE, GM, GTE, HP, IBM, Ingersoll, Iroquois, JNJ, Kimberly-Clark, Kin Ark, Kodak, Lukens, Meicco, Merck, Mobil, MMM, Morris Mining, P&G, Pillsbury, Schlum, Sears, Sherman Williams, and Texaco.

3.2.2.2 Data Cleaning and Exploration

Once the relevant data has been collected, it is important to spend some time exploring the database for two reasons. First, the data analyst must be intimate with the data–not just knowing the attribute names and what they are intended to mean but the actual contents of the database. Second, there are many sources of error when collecting data from multiple databases into a single analytical database and a good analyst must perform several sanity checks to validate the extracted data.

Working within almost every problem, there has been something wrong with the database collected for analysis. The three main problems have been: mislabeling of fields in the database; data values with special semantics; and what we call temporal infidelity or time travel-cases where an attribute in the database contains information that could not be known at the time the model is intended to be applied, because the information only becomes known in the future.

For example, in the financial database studied in the next chapter, one field was labeled “volume”, but upon inspection it was found that the values in that field were uncorrelated with the change in price of the stocks. After talking to several people, it had been found out that the fields had been mislabeled and it was actually the next field that contained the returns. Had we just run a data mining algorithm without first doing some sanity checking, it would have got completely useless results but would not have realized it. Thus, one kind of sanity checking is looking at the names of the fields,

thinking about how they ought to be related and checking to see that the expected relationship does hold.

The general rule here is to do some preliminary data mining runs and checks for unbelievably good results and ensure that there is no temporal infidelity in the database. In this research, an iterative optimization method is used (George, 1997) that alternately builds a model on a training database, and then removes that set of records whose removal from the training set will most increase the model's estimate of its own performance (its "self-esteem") on the training data. This process of deleting records that disagree with the model is called Esteem; for Elimination of Suspicious Training Examples with Error on the Model. While standard data mining algorithms penalize a model for being too complex, this method also penalizes the data itself for being too hard to learn, by removing records from the training database. As in a case study using the C4.5 algorithm for building classification tree models, the Esteem method results in much more understandable patterns (in this case, trees) with slightly better predictive ability.

3.2.2.3 Data Engineering

The previous steps have been concerned with creating and cleaning the mining base which is a static database containing all of the information that would be ever wanted to use in our data mining runs. There are three problems at this stage. First, the mining base might contain many attributes that could profitably be ignored. Selecting which subset of attributes to use is an important problem addressed in Chapter 4 and Chapter 6, in view of the respective algorithms to be used. Second, the mining base might contain many more records than can be analyzed during the available time, in which case we must sample the mining base. John and Langley (1996) discuss the desirable properties of a sampling method. Third, some of the information in the mining base might be profitably expressed in a different way for a particular analysis. As one explores different solutions to these problems in the course of a data mining project, the

data engineering step is repeated many times to converge on the best customized mining base for our purposes, where customization addresses all the three problems mentioned above.

In contrast to the mining base, which is usually created only once during the data extraction step, many customized mining bases are created during the data engineering step, to explore the use of different attributes, different sample sizes, and different precise definitions of the problem to be solved. For example, it could be tried to predict stock market for one, two, or three months in the future using only the services consumed as predictors.

It might also be wanted to ignore records if it were felt they were incorrect, or if they represented customers that were somehow odd and should not be used to infer general patterns. Chapter 4 and 5 describe one way to detect potentially incorrect values in the database, by making a pass through the data mining process and using the discovered model itself to identify suspicious records. Considerable human intelligence is injected into the process during this step in the data mining process. Langley and Simon (1995) contend that much of the success of machine learning, and by inheritance data mining, is due to the problem formulation and representation of engineering steps, which correspond to our problem definition and data engineering steps.

3.2.2.4 Algorithm Engineering

Often an "of-the-shelf" mining algorithm can be run on the customized mining base, but there remains the problem of selecting which algorithm to use and deciding specifically how to apply it. For example, it could be used a classification tree algorithm on the customer retention problem. In this case, the algorithm and the problem seem well matched: the problem is to predict the value of a single attribute (whose values are nominal or categorical) and the inputs are a set of categorical and numeric attributes. Most classification tree algorithms are built to work in precisely this setting. There are

many such algorithms from which to choose and this choice is likely to have a significant select on the quality of the mined patterns. Even once a single classification tree algorithm is chosen; there are still several parameters to set.

For example, the patterns discovered by the C 4.5 classification tree algorithm (Quinlan, 1993) can depend heavily on the parameter setting, and it is difficult to have any kind of intuition about what parameter setting will give the best results. Standard practice is to either ignore the fact that there are parameters that can be tuned, and just leave them set at their default values, or to run a few experiments with different parameter settings and see which perform the best. Both approaches explore only a miniscule portion of the whole parameter space. In Chapter 4, some classification methods are described for tuning the parameters of a mining algorithm. It requires some objective function that can estimate the quality of the discovered patterns, and use this objective function to perform an optimization in their parameter space.

As shown in Figure 3.2, the overall stock market prediction system, developed in this project, mainly consists of two parts: 1) based on Fuzzy decision tree (FDT); and 2) based on support vector regression (SVR). The FDT based subsystem is described in Chapter 4 and 5, whereas the SVR based subsystem is described in chapter 6. From the implementation point of view, a mainmenu window of the SVR-FDT Financial Forecaster is shown in Figure 3.3. A user can select whether to use KnowledgeMiner (based on Fuzzy predictive decision tree) or SVR-Forecaster (based on support vector regression), or use one after the other.

The FDT based subsystem consists of two major steps (as shown in Figure 3.2): first construction of predictive FDT, second similarity-based fuzzy reasoning method is used to mine the production rules. The summary of the classification data mining model as well as that of support vector regression model is presented in the following subsections.

(i) Predictive Fuzzy Decision Tree Construction

The classification model is based on fuzzy decision tree. Fuzzy decision trees have been found as a useful classification method in building knowledge-based expert systems. In this project, the predictive FDT algorithm is presented for the classification of stock market index. Before applying the predictive FDT algorithm, first training data sets are evaluated using three important steps as: 1) centroids of fuzzy sets (*k*-means), 2) fuzzification of numerical numbers, 3) triangular membership function. The following paragraphs present the summary of these steps. In this project, *k*-means clustering algorithm is used to find the centers for every data sample from historical stock market data. K-means (MacQueen, 1967) is one of the simplest unsupervised learning algorithms that solve the well known clustering problem.

The procedure follows a simple and easy way to classify a given static data set through a certain number of clusters (assume k clusters) fixed a priori. The main idea is to define k centroids, one for each cluster. These centroids should be placed in a cunning way because different locations cause different result. This algorithm aims at minimizing an objective function, in this case a squared error function. The objective function is:

$$J = \sum_{j=1}^{k} \sum_{i=1}^{n} \| x_i^{(j)} - c_j \|^2 \tag{3.1}$$

where $\| x_i^{(j)} - c_j \|^2$ is a chosen distance measure between a data point $x_i^{(j)}$ and the cluster centre c_j, $\sum \| \, . \, \|$ is an indicator of the distance of the *n* data points from their respective cluster centers. The detail about this method is presented in chapter 4.

Fuzzification is a process of fuzzifying numerical numbers into linguistic terms, which is often used to reduce information overload in human decision-making process. Linguistic terms are simple forms of fuzzy values but generally their membership functions are unknown and need to be determined.

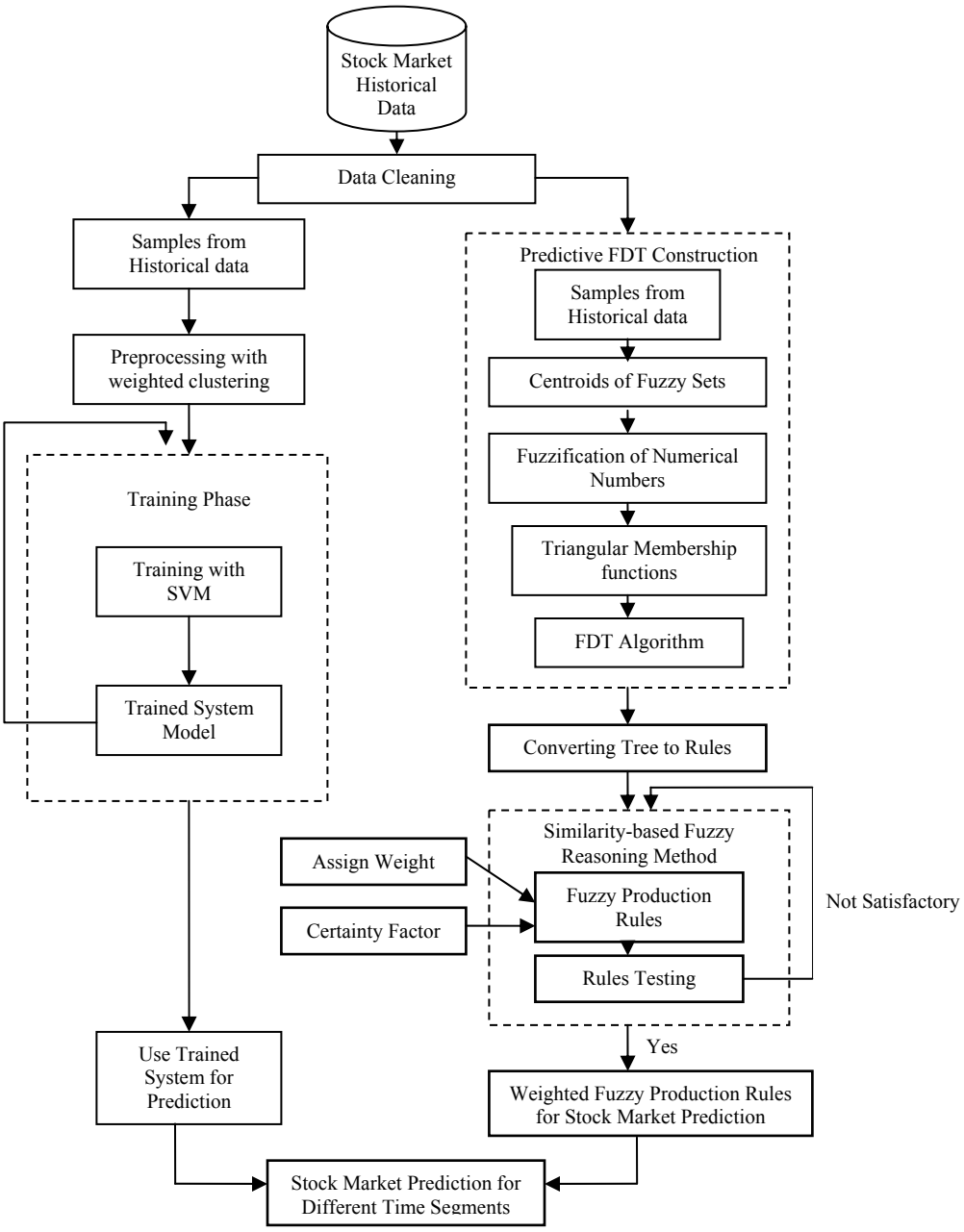


Figure 3.2: System development process

Figure 3.3: MainMenu window of the overall Stock Market Prediction System: A user can select whether to use KnowledgeMiner (based on Fuzzy predictive decision tree) or SVR-Forecaster (based on support vector regression)

Membership functions are very important in fuzzy rules. Membership functions affect not only the performance of fuzzy rule based systems but also the comprehensibility of rules. One way of determining membership functions of these linguistic terms is by expert opinion or by people's perception. Yet another way is by statistical methods (Mukaidono *et al*., 1990) and fuzzy clustering based on self-organized learning (Kohonen, 1988). In this project, for fuzzification of numerical numbers, triangular membership functions are used to calculate fuzzy numbers.

Predictive FDT algorithm is based on the concept of degree of importance of attribute contributing to the classification. This concept was firstly proposed by Pawlak (1991) while investigating the reduction of knowledge. It was used to extract the

minimum indispensable part of equivalent relations. Later on Yeung *et al*. (1999) and Wang *et al*. (2001) extended this concept to a fuzzy case and then used it to select the expanded attribute at a considered node while generating fuzzy decision trees. The same idea is extended in this research for the construction of fuzzy decision tree. In this project we extend this idea by assigning different weights to each arc of decision tree from root node to leaf node.

(ii) Similarity-based Fuzzy Reasoning Method

After the construction of decision tree classification, similarity-based fuzzy reasoning method is applied on extracted weighted fuzzy production rules. In similarity-based fuzzy reasoning method, the basic idea is taken from equality and cardinality (EC) method proposed by Yeung and Tsang (1997b).

In fuzzy reasoning method, first simple rule case is considered, where only one observation A' and one simple rule in the form $R: A \rightarrow C$ is presented. The basic idea is to modify the consequent C of a simple rule $R: A \rightarrow C$ according to the closeness of an observation (fact) A' to the antecedent (pattern) A of the rule R. If they are close (similar) enough in comparison to a threshold, then the rule R can be fired and the consequent can be deduced by some modification technique to be addressed later in chapter 5.

(iii) Support Vector Regression Method

As already mentioned, for this project, in addition to FDT method, the SVM model is built using support vector regression that is widely used for time series prediction. In other words, both the subsystems (FDT and SVR based) developed in this project are complementary to each other. As the fuzzy decision tree based system gives easily interpretable results, we mainly use it to classify past and present data records. Whereas we use the stronger aspect of the SVR based approach for prediction of future trend of the stock market.

There are several issues in building the SVM model that include the preparation of training and testing data sets, and selection of parameters. Because, in the financial time series, the recent data points could provide more important information than the distant data points, before applying the SVR method, we preprocess the data using a weighted kernel-based clustering algorithm incorporating neighborhood constraints, in which the weights are assigned accordingly. The developed algorithm is described in chapter 6. The algorithm also has a mechanism to handle outliers. The SVR based system follows the following procedure:

1. Preprocess and transform data
2. Conduct simple scaling
3. Choose kernel function
4. Use cross-validation to find the best parameters $\dot{C}$ and σ
5. Use the best parameters $\dot{C}$ and σ for the whole training data to get the trained model
6. Conduct testing

3.2.2.5 Running the Data Mining Algorithm

The next step, running the data mining algorithm is the point where the analyst and expert can just sit back and watch. Thousands of researchers and developers are busily developing robust algorithms that work well on a wide variety of problems, and this is the step in the process that leverages (mechanical advantage) their work. Some refer to this single step in the process as data mining, while referring to the whole process as knowledge discovery in databases (Fayyad *et al*., 1995). Our contributions to the set of algorithms, all addressing classification and regression problems are described in Chapters 4–6. Such algorithms produce predictive patterns, which are generally

evaluated by seeing how well they do their job, predicting the value of one or more attributes given the rest.

To do this evaluation properly, some percentage of the records must be held out from the customized mining base, and this set is call the validation set or holdout set. The remainder of the records constitutes the training set and is mined by the algorithm to produce a set of patterns. Modern data mining algorithms are designed to avoid discovering "patterns" that may exist in the training set but are in fact just accidental and do not hold in the validation set. Still the performance of a model on the data from which it was built is usually better than its performance on an independent sample of data from the same distribution, so it is important to use an independent sample to evaluate the quality of the discovered patterns.

Keeping a fraction of the records in a holdout set is a sufficient way to get an unbiased estimate of the performance of a single model. But if the set of models are built by trying different algorithms or different custom mining bases and if we use the holdout set performance to choose the best among the set of models, then we have the problem of accurately estimating the performance of the best model. Since the holdout set was used to choose the best model, it no longer gives an unbiased estimate of the performance of the best model. What we really need is yet another independent sample of data, called the test set that will only be used once to evaluate the final recommended model. However, even the results on the test set should always be treated as an estimate of the performance of the model. The only true test is to decide how to actually use the model in practice, use it, and evaluate the actual results.

3.2.3 Implementation and Integration (Phase 3)

System has been implemented for historical and real time data by using decision tree inductive learning method, and the support vector regression method. In inductive learning method, first step is learning from example and then constructing predictive

fuzzy decision tree for every sets of records. After that rules can be extracted from predictive FDT and these rules are used in the similarity-based fuzzy reasoning method to mine significant WFPRs for stock market prediction. Whereas, the SVR based system is mainly used for the prediction of future trends in the stock market.

3.2.3.1 Integrated System Components

The developed intelligent knowledge discovery system mainly consists of two subsystems–FDT based system and SVR based system–that are complementary to each other.

The FDT-based system further consists of two major parts: first evaluation of comprehensive predictive fuzzy decision tree and second to extract weighted fuzzy production rules from predictive fuzzy decision tree to predict up or down of the stock prices, especially to classify the stock market data patterns. For the construction of predictive FDT equal sets of patterns are taken from historical stock market data and the centroid for every set of patterns are found by using *k*-means clustering algorithm. These centroids are used to find degree of membership function for linguistic terms. When the degree of membership for every linguistic term has been calculated, these terms are applied to predictive FDT algorithm. Every path from root to leaf is considered as a rule. The different weights are assigned to antecedent part and certainty factor is calculated for every set of rules. When the sets of rules are extracted with different weights from predictive FDT then similarity-based fuzzy reasoning mechanism is applied to mine more accurate rules for better predictive accuracy for up or down movement of the stock market prices.

In addition, the support vector regression based system is implemented for forecasting future trends of the stock prices. Here, we integrate weighted clustering algorithm for getting improved prediction results.

3.2.3.2 System Performance Testing

The performance of the system can be tested by the extraction of precise weighted fuzzy production rules from proposed algorithm of predictive fuzzy decision tree for 100 different companies during the period January 1, 1999 to December 31, 2004 from KLSE, NYSE and LSE. The performance of the system is tested by the parameters like: accurate prediction, complexity, and comprehensibility. For precise WFPRs from predictive FDT, system can extract rules by calculating certainty factors. Each rule has associated strength, since rules may overlap the class exceptional or unexceptional that system finally predicts for a stock is a combination of the predictions made by each rule. A rule's prediction is weighted by the amount of evidence supporting the rule, letting system not only make a prediction but also assign some measure of certainty or strength to that prediction.

The system begins with all single-antecedent rules and then adds conditions, using a combination of several heuristics to concentrate on interesting rules, ultimately yielding a set of rules with high predictive accuracy. Each rule is characterized by statistics that describe the training-sample estimates of the conditional distribution of the class given whether or not the antecedent of the rule is true; thus the search through the space of rules is essentially only a search through the space of the multivariate statistics of the data that are pertinent to the classification problem. Moreover, mean squared error is a popular metric for measuring the accuracy of regression models.

3.3 Summary

This chapter presents the summary of all methods that have been used in development of the proposed system for the classification and regression for stock market prediction. The project operational framework can be divided into three phases. In the first phase, with the help of most recent literature and study of current systems the

actual problems is analyzed. In the second phase, the data is collected from KLSE, NYSE and LSE for analyzing the behavior of stock indices and also to develop the predictive FDT and SVR method. In predictive FDT, WFPRs are extracted with the help of proposed algorithm and then these rules are used for stock market prediction. In the third phase, the performance of system is tested by applying the different sets of extracted rules for the prediction of stock market data. In addition, the system begins with all single-antecedent rules and then adds conditions, using a combination of several heuristics to concentrate on interesting rules, ultimately yielding a set of rules with high predictive accuracy. Each rule is characterized by statistics that describe the training-sample estimates of the conditional distribution of the class given whether or not the antecedent of the rule is true; thus the search through the space of rules is essentially only a search through the space of the multivariate statistics of the data that are pertinent to the classification problem.

CHAPTER 4

INDUCTIVE LEARNING OF PREDICTIVE FUZZY DECISION TREE

4.1 Introduction

Decision trees are a well-known and widely used method for classification problems. For handling numerical attributes or even for numerical prediction, traditional decision trees based on crisp predicates are not suitable. Through the usage of fuzzy predicates for different types of attributes not only the expressive power of decision trees can be extended, but it also allows creating models for numerical attributes in a very natural manner. In this chapter, predictive FDT algorithm is used for decision making of real time stock markets indices. In order to handle stock market time series data, inductive learning method has been used. Inductive learning is one of the powerful methods for decision-making.

4.2 Decision Trees Induction

Decision tree induction is one of the most important branches of inductive learning. It is one of the most widely used and practical methods for inductive inference. It creates a decision tree from the instance table according to heuristic information and classifies some instances by using a set of rules, which is transformed from decision tree. In decision tree each edge of the tree links two or more nodes, the initial node and

the terminal node. The former is called the parent-node of the latter while the latter is said to be the child-node of the former. The node that has no parent-node is the root whereas the nodes that have no child-nodes are called leaves.

A crisp decision tree has two key aspects, training and matching. The former is a process of constructing trees from a training set which is a collection of objects whose classes are known, while the latter is a process of judging classification for unknown cases. A fuzzy decision tree is a generalization of the crisp case. It is more robust tolerating imprecise information. Compared with the crisp case, the fuzzy decision tree has the characteristics listed in Table 4.1.

There are two kinds of instance attributes: symbolic valued attribute and continuous valued attribute. For the latter, it needs to do pretreatment (discretization or fuzzification) before inductive learning. It can be analyzed from Table 4.1 that fuzzy decision trees are better than crisp decision trees in terms of testing accuracy (i.e., the prediction accuracy) and the size of the tree (i.e., number of internal and leaf nodes, and number of rules) in order to handle symbolic attribute as well as continuous valued attributes. Also it can be explicitly analyzed and compare the generalization capability between fuzzy and crisp decision tree generation algorithm.

4.3 Fuzzy Decision Tree Induction

Many algorithms for construction fuzzy decision trees focus on the selection of expended attributes using different criteria. They attempt to obtain a small-scale tree via the expanded attribute selection and to improve the classification accuracy for unknown cases. It is easy to see that choosing an attribute to split the examples based on those algorithms (Quinlan, 1993; Umanol *et al*., 1994; Yuan and Shaw, 1995; Wang *et al*., 2001) results in maximize the number of instances correctly classified at a given node. These algorithms are greedy search strategies correspond to a locally optimum decision. The difficulty is that a combination of locally optimal decisions can not guarantee the

Table 4.1: A comparison between the fuzzy decision tree and the crisp decision tree

Parameters	Crisp Decision Tree	Fuzzy Decision Tree
Node types	Nodes are crisp subsets of X	Nodes are fuzzy subsets of X
Attribute selection criteria	Select the attribute with the smallest entropy as the root decision node.	Select the attribute with the smallest average fuzzy entropy as the root decision node.
Classification criteria	If all examples are in the same class, return the single-node tree root.	If the truth level of classifying into one class is above a given threshold β return as a leaf.
Utility of rules	A path from the root to a leaf corresponds to a production rule	A path from the root to a leaf corresponds to a fuzzy rule with some degree of truth
Matching paths	An example remaining to be classified matches only one path in the tree	An example remaining to be classified can match several paths in the tree
Sub-nodes on same layer	The intersection of sub-nodes located on the same layer is empty	The intersection of sub-nodes located on the same layer can be nonempty
Algorithms	If all attributes have been used return the single-node tree. Otherwise begin a) Choose the unused attribute with the max information gain as target attribute and add a new tree branch below root. b) If the max information gain is less than a given threshold α return a leaf. c) Clearly partition the current node according to the value of target attribute and create a child-node d)Below this new branch add the sub-tree	If all attributes have been used return as a leaf. Otherwise begin a) Choose the unused attribute with the max fuzzy information gain as target attribute and add a new tree branch below root. b) If fuzzy gain is less than a given threshold α return as a leaf. c) Fuzzily partition the current node according to the value of target attribute creates a child-node. d) Below this new branch add the sub-tree.
Extracted rules types	Convert the decision tree into a set of classification rules.	Convert decision tree into a set of fuzzy rules.

global optimum tree i.e. a tree with smallest size. Indeed it is an NP-hard problem to find the smallest tree or one with the least number of leaves.

For a probabilistic distribution, each component is considered as the probability with which the corresponding event occurs. A possibilistic distribution is usually considered as fuzzy sets (vector), and each component of the vector, i.e., the

membership degree, is regarded as the possibility with which the corresponding event occurs. For the difference and consistency between probability and possibility one can refer to (Zadeh, 1999). The difference between the uncertainties described by the entropy of a probabilistic distribution and described by the fuzzy entropy of a possibilistic distribution is the former attains its maximum at all components being 0.5 but the latter does not. Fuzzy ID3 (Umanol *et al*., 1994) uses the fuzzy entropy of a possibilistic distribution. The expanded attribute selection of fuzzy ID3 (Umanol *et al*., 1994) is briefly described as follows:

Consider a test node S having n attributes $T^{(1)}, T^{(2)}, ..., T^{(n)}$ to be selected. For each $k(1 \le k \le n)$, the attribute $T^{(k)}$ takes m_k fuzzy subsets (linguistic terms) S. $T^{(n+1)}$ denotes the classification attribute, taking values $L_1^{(n+1)}, L_2^{(n+1)}, ..., L_m^{(n+1)}$. For each attribute value (fuzzy subset), $L_i^{(k)} (1 \le k \le n, 1 \le i \le m_k)$, its relative frequency concerning the j^{th} fuzzy class $L_j^{(n+1)} (1 \le j \le m)$ at the considered nonleaf node S is defined as

$$p_{ij}^{(k)} = M(L_i^{(k)} \cap L_j^{(n+1)} \cap S) / M(L_i^{(k)} \cap S) \tag{4.1}$$

where p is the probability for classification attributes. At the considered nonleaf node S, the fuzzy classification entropy of $L_i^{(k)} (1 \le k \le n, 1 \le i \le m_k)$ is defined as

$$Entr_i^{(k)} = -\sum_{j=1}^{m} p_{ij}^{(k)} \log_2 p_{ij}^{(k)} \tag{4.2}$$

The averaged fuzzy classification entropy of the k^{th} attribute is defined as

$$E_k = \sum_{i=1}^{mk} w_i Entr_i^{(k)} \tag{4.3}$$

in which w_i denotes the weight of the i^{th} value of $L_i^{(k)}$ and is defined as

$$w_i = M(S \cap L_i^{(k)}) / \sum_{j=1}^{mk} M(S \bigcap L_j^{(k)}) \tag{4.4}$$

The average fuzzy classification entropy of the *kth* attribute $T^{(k)}$ is defined as

$$E_{ko} = Min_{1 \le k \le n} E_k \tag{4.5}$$

in which w_j is defined by equation (4.4).

Fuzzy ID3 approach aims to search for an attribute such that its average fuzzy entropy attains minimum, i.e., selecting an integer $k_\circ$ (the $k_\circ th$ attribute) such that $E_{ko} = Min_{1 \le k \le n} E_k$.

Yuan and Shaw (1995) present another powerful FDT algorithm. In this method instead of using minimum fuzzy entropy, they use the minimum classification ambiguity to select expanded attributes. Continuing to use the notations in above paragraph Yuan and Shaw's method (Yuan and Shaw, 1995) is briefly formulated below.

At the considered nonleaf node S, the classification ambiguity of $L_i^{(k)} (1 \le k \le n, 1 \le i \le m_k)$ is defined as

$$Ambig_i^{(k)} = \sum_{j=1}^{m} (\pi_{ij}^{(k)} - \pi_{i,j+1}^{(k)}). \tag{4.6}$$

Where π is the fuzzy subsets (linguistic term). In j in $(\pi_{i1}^{(k)}, \pi_{i2}^{(k)}, \ldots, \pi_{im}^{(k)})$ with descending order $\pi_{i,m+1}^{(k)} = 0$ is a permutation of $(\tau_{i1}^{(k)}, \tau_{i2}^{(k)}, \ldots, \tau_{im}^{(k)})$ which is a normalization of $(p_{i1}^{(k)}, p_{i2}^{(k)}, \ldots, p_{im}^{(k)})$, i.e., $\tau_{ij}^{(k)} = p_{ij}^{(k)} / \max_j p_{ij}^{(k)}$. The average classification ambiguity of the *kth* attribute is defined as

$$G_k = \sum_{i=1}^{mk} w_i Ambig_i^{(k)} \tag{4.7}$$

in which w_i is defined identically as for fuzzy ID3.

Yuan and Shaw's (1995) method aims to search for an attribute such that its averaged classification ambiguity attains minimum, i.e., selecting such an integer $k_\circ$ (the $k_\circ th$ attribute) that $G_{k0} = Min_{1\le k\le n} G_k$.

In another fuzzy decision tree method proposed by Wang *et al*. (2001) which is based on the maximum degree of importance of attribute contributing to the fuzzy classification. It aims, on the considered node with several attributes to be chosen to select an attribute whose contribution to classification is maximal. The main strength of Wang *et al*. (2001) is that the algorithm can generate weighted fuzzy production rules from FDT. A weight fuzzy rules means that for each proposition of antecedent of the rule, a weight is assigned to indicate the degree of importance of the proposition.

The construction of the trees, based on above three methods by using the data sample of size 20 shown in Figure 4.1 (a)-(c). The possible explanations about these techniques are as follows. On a nonleaf node Fuzzy ID3 (Umanol *et al*., 1994) aims on average to select an attribute such that the components of the frequency vector are as close to zero or one as possible. Whereas Yuan and Shaw's (1995) method aims on average to select an attribute such that only one component is as close to one as possible and other components are as close to zero as possible. Proposition 2 and 3 (in section 2.4.2) indicate that there is a significant difference between fuzzy ID3 and Yuan and Shaw's method when training examples have much classification uncertainty (ambiguity). Moreover, Propositions 1 and 2 (in section 2.4.2) show that these two heuristics are gradually consistent when classification ambiguity is small, which gives us such guidelines that ID3 method is better if more than one fuzzy set in the consequent part of the generated fuzzy rule is acceptable and Yuan and Shaw's method is better if its is unacceptable.

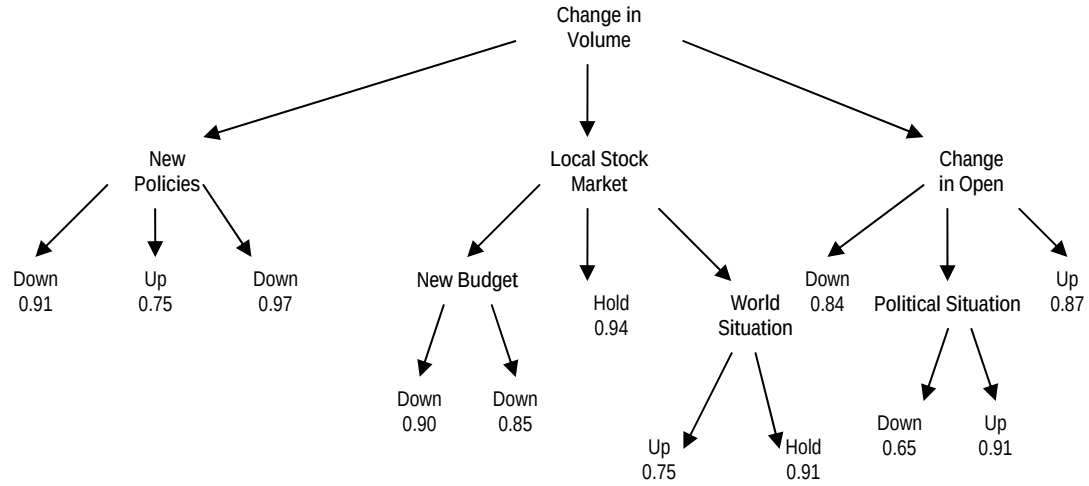


Figure 4.1(a): Fuzzy decision tree using Fuzzy ID3 heuristic (Umanol *et al*., 1994)

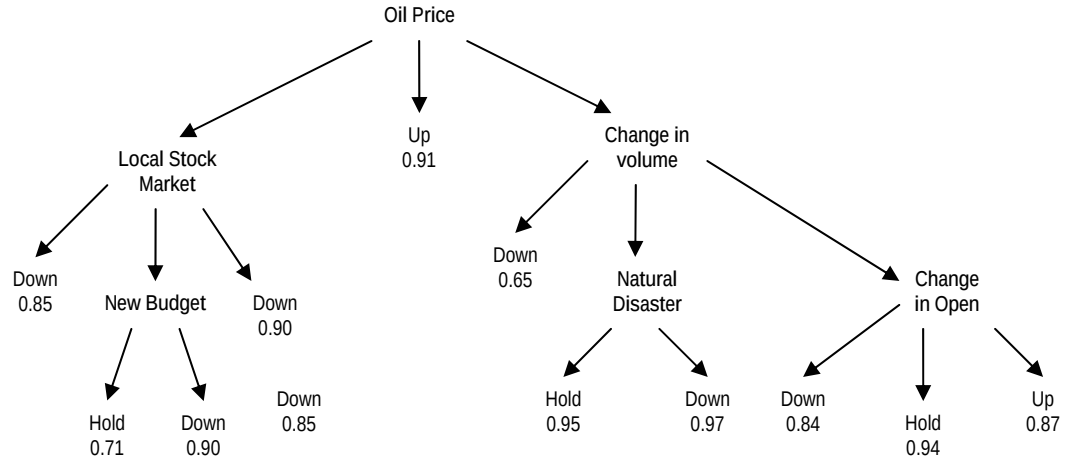


Figure 4.1(b): Fuzzy decision tree by Yuan and Shaw (1995) heuristic

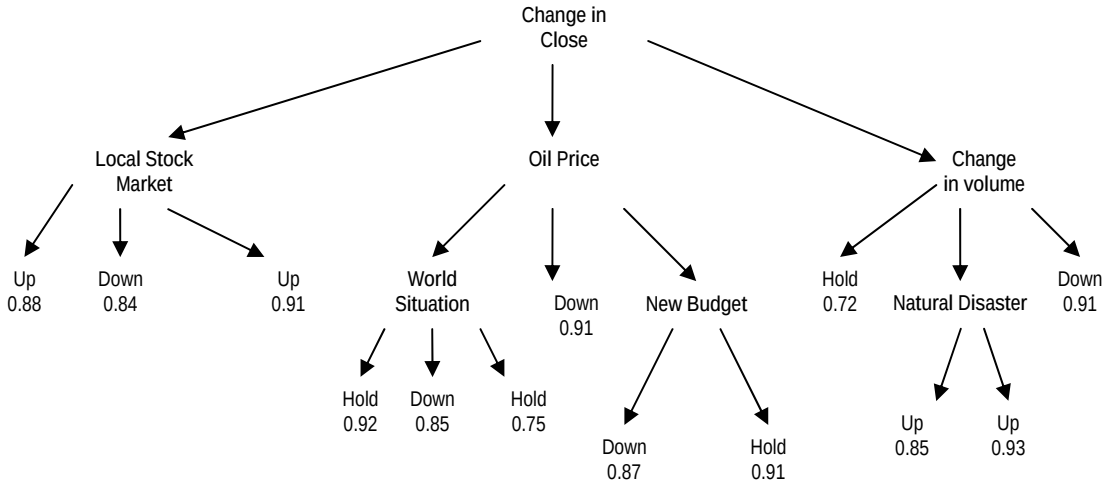


Figure 4.1(c): Fuzzy decision tree by using Wang *et al*. (2001) heuristic

Wang *et al*.'s (2001) method gives a new way to roughly assigned the weights to every arc of decision tree while generating the fuzzy decision tree (fuzzy rules). That is for a linguistic term of an expanded attribute its degree of importance is regarded as the weight of the corresponding proposition. In this way, while converting the tree into a set of rules, a number of weighted fuzzy rules are derived. In Yeung *et al*. (1999) and Wang *et al*.'s (2001) method, weight parameters and certainty factor have been assigned to each proposition of the rules without any proper calculation. However such assignments of rules are very sensitive and are not suitable in every problem. Instead of impressive results of these three methods of fuzzy decision trees, these methods still cannot solve the problems: complexity, comprehensibility and high learning accuracy.

4.4 Stock Market Data Collection and Extraction

To illustrate about the predictive FDT and fuzzy reasoning method, it is important to know how the data is collected and extracted from time series stock market. In this research, the three basic attribute P_O = price of open, P_C = price of close, V = volume are taken from stock market data. Also, some other factors are also considered that can affect stock market after 5 minutes of period. These factors are as follows:

P_{OP} = Oil Price, P_{ND} =Natural Disaster, P_{PS} =Political Situation, P_{NP} =New Policies, P_{NB} =New Budget, P_{OC} =Other Companies, P_{LSM} =Local Stock Markets, P_{OWS} =Overall World Situation.

These factors are calculated for different associated stocks from KLSE, NYSE and LSE after analyzing the daily stock prices in different periods of times. Table 4.2 presents an example of real time data sample for every 5 minutes of periods during 9:30 to 11:05.

Table 4.2: 20 real time (5-Minute Bars) examples of stock market data

No	Data and Time	P_O	P_C	V	P_{OP}	P_{ND}	P_{PS}	P_{NP}	P_{NB}	P_{OC}	P_{LSM}	P_{OWS}	Primary Signal
1	08/02/04 9:30	39.8	39.84	79,100	0	-0.03	0.03	0.08	0	0.75	-0.06	-0.36	0.0090
2	08/02/04 9:35	39.84	39.76	19,000	0	-0.35	-0.07	0	0	0.58	-0.26	0.02	0.0150
3	08/02/04 9:40	39.74	39.8	30,800	0	0.24	-0.09	-0.01	0	-0.32	-0.25	0.26	-0.0006
4	08/02/04 9:45	39.8	39.85	64,200	0	-0.45	-0.03	0.09	0	0.85	-0.65	0.32	0.0292
5	08/02/04 9:50	39.85	39.97	31,900	0	-0.12	-0.04	-0.04	0	-0.54	0.09	-0.32	-0.0222
6	08/02/04 9:55	39.97	39.88	39,300	0	-0.34	-0.25	-0.49	0	-0.04	-0.62	0.25	-0.0238
7	08/02/04 10:00	39.87	39.9	43,800	0	0.48	0.18	-0.08	0	-0.39	-0.85	-0.69	-0.0806
8	08/02/04 10:05	39.9	39.86	27,100	0	0	0.25	0	0	-0.07	0.22	0.32	-0.0237
9	08/02/04 10:10	39.86	39.78	8,800	0	0	0.19	0	0	0.26	-0.02	0.01	0.0245
10	08/02/04 10:15	39.77	39.68	13,500	0	0	-0.78	0	0	-0.24	-0.55	-0.33	0.0886
11	08/02/04 10:20	39.67	39.71	14,800	0	-0.26	-0.19	0	0	0.02	0.02	0.08	0.0706
12	08/02/04 10:25	39.72	39.77	17,400	0	-0.24	0.18	0	0	-0.25	0.38	-0.26	0.0598
13	08/02/04 10:30	39.78	39.81	26,100	0	-0.19	0.08	0.05	0	0.02	-0.65	0.38	-0.0451
14	08/02/04 10:35	39.83	39.85	26,200	0	-0.04	-0.08	-0.25	0	-0.05	-0.58	0.28	-0.0949
15	08/02/04 10:40	39.85	39.89	34,100	0	-0.07	-0.09	-0.48	0	0.07	-0.23	0.58	-0.0750
16	08/02/04 10:45	39.92	39.9	23,900	0	-0.09	-0.47	-0.15	0	-0.08	-0.28	-0.38	-0.0425
17	08/02/04 10:50	39.91	39.93	30,800	0	-0.08	0.19	0.35	0	-0.87	-0.26	0.85	-0.0028
18	08/02/04 10:55	39.93	39.94	14,300	0	0.04	0.78	-0.18	0	0.26	-0.36	0.11	-0.0083
19	08/02/04 11:00	39.96	39.92	14,400	0	0.15	-0.32	0.34	0	0.98	-0.47	-0.23	-0.0068
20	08/02/04 11:05	39.92	39.91	7,900	0	-0.04	-0.48	-0.19	0	0.27	-0.15	-0.52	-0.0168

The last column in table 4.2 is showing the target classification attribute with name primary signals. The primary signals are defined after evaluating the final trends of every stock price with in 5 minutes of periods. Particularly, the five basic attributes i.e., price of open, price of low, price of high, and price of close are used to calculate primary signals. Table 4.3 shows the change in stock prices of open, close and volume. In this research, these changes are calculated by taking the difference of current trends from previous trends of stock prices.

Table 4.3: Change in stock trading prices for every 5 minutes of time series stock market

No	C_{P_O}	C_{P_C}	C_V	Difference from Moving Average (%)	Standard Deviation	Primary Signal
1	0.02	0.03	900	-0.0014	0.1339	0.009044
2	0.04	-0.08	-60,100	-0.0032	0.135	0.015078
3	-0.1	0.04	11,800	-0.0021	0.1333	-0.00068
4	0.06	0.05	33,400	-0.0007	0.1304	0.029188
5	0.05	0.12	-32,300	0.0024	0.1287	-0.02223
6	0.12	-0.09	7,400	0.0002	0.1268	-0.02385
7	-0.1	0.02	4,500	0.0008	0.1234	-0.08063
8	0.03	-0.04	-16,700	-0.0001	0.1208	-0.02378
9	-0.04	-0.08	-18,300	-0.0019	0.1189	0.024484
10	-0.09	-0.1	4,700	-0.0043	0.1205	0.088576
11	-0.1	0.03	1,300	-0.0034	0.1211	0.070576
12	0.05	0.06	2,600	-0.0017	0.1201	0.059763
13	0.06	0.04	8,700	-0.0007	0.1173	-0.04507
14	0.05	0.04	100	0.0004	0.1122	-0.0949
15	0.02	0.04	7,900	0.0014	0.1074	-0.07498
16	0.07	0.01	-10,200	0.0017	0.1042	-0.04252
17	-0.01	0.03	6,900	0.0025	0.1006	-0.00279
18	0.02	0.01	-16,500	0.0028	0.0958	-0.00829
19	0.03	-0.02	100	0.0023	0.0905	-0.00679
20	-0.04	-0.01	-6,500	0.0021	0.0855	-0.01675

4.5 Data Cleaning using ESTEEM (Elimination of Suspicious Training Examples with Error on the Model) Method

Data cleaning is a very important step in the data mining process. Errors in large databases can be extremely common, so an important property of the data mining process is robustness with respect to errors in the database. Most sophisticated methods in data mining address noisy data to some extent, but not fully, and they can be improved by addressing the problem more directly. In this research, Esteem (Elimination of Suspicious Training Examples with Error on the Model), method (George, 1997) is used for data cleaning that uses a data mining algorithm to find patterns in a database then “improves the self-esteem” of the data mining algorithm by removing records from the training database that do not fit the discovered patterns, and retraining. The ESTEEM method is used to clean data for the predictive FDT classification induction

algorithm and show that on the stock market databases, the resulting algorithm builds much smaller trees with slightly better accuracy than the standard algorithms.

Pruning is a method of local data selection, since records are effectively removed from the partitions of the training set that induce the nodes deep in the tree. Pruning a node (i.e., removing the sub-tree rooted at the node so that the node becomes a Leaf) has the same effect as removing all records that were not in the majority class in the subset of the database used to build that node's sub-tree. This raises the question: if the records are locally un-informative or harmful, then why should be suspected that they helped the algorithm to discover patterns higher in the tree that is more globally?

ESTEEM (Algorithm, TrainingData)

- repeat {
  - Model ← Apply (Algorithm, TrainingData)
  - fore each record in TrainingData
    - if Model misclassifies Record then
    - remove Record from TrainingData

  } until Model correctly classifies all

- Records in TrainingData

Figure 4.2: The Esteem method of outlier removal

The ESTEEM method repeatedly runs a classification algorithm and removes those records it misclassifies from the training database, until all records in the reduced training set are correctly classified (Figure 4.2). The novelty in the ESTEEM approach is to rerun the mining algorithm using the reduced training set, which is the original training set minus the records which the first model classifies incorrectly. While retraining may seem odd, it is in fact just an extension of the assumptions underlying pruning. In predictive FDT pruning the tree essentially assumes that these confusing records are locally not useful. Retraining merely takes this assumption a step further by

completely removing these records from the training set. This makes explicit the assumption that locally uninformative or harmful records are globally uninformative as well. ESTEEM method continues pruning and retraining until no further pruning can be done. ESTEEM is guaranteed to stop since the size of the training set monotonically decreases as a function of the number of iterations through the main loop.

4.6 Predictive Fuzzy Decision Tree (FDT)

Decision trees are a well-known and widely used method for classification problems. For handling numerical attributes or even for numerical prediction, traditional decision trees based on crisp predicates are not suitable. Through the usage of fuzzy predicates for different types of attributes not only the expressive power of decision trees can extend but it also allows creating models for numerical attributes in a very natural manner. For this purpose, some attempts are being made for the last decade and introduce fuzzy decision tree for numerical prediction but the existing fuzzy decision tree is still suffering some problems like complexity, comprehensibility of tree, over-fitting, robustness, scalability, and mining useful fuzzy rules from numerical attributes.

Before applying the predictive FDT algorithm, first training data sets are evaluated after three important steps as: 1) centroids of fuzzy sets (*k*-means), 2) fuzzification of numerical numbers, 3) triangular membership function (as shown in Figure 4.3). In first step, *k*-means clustering algorithm compresses the data set and to find the center concerning this data set. Usually, a fuzzy attribute can take many values if the representation of the fuzzy value is given directly by a membership degree. Like numerical attributes, the range of such a fuzzy attribute can be described as the interval $[1,0]^M$ where M is the dimension. For a given set of membership functions, the intention in this research is to find several new fuzzy sets, which are regarded as clustering result to reasonably describe this set of membership functions. In second step, fuzzy sets have been proposed to deal with numerical and categorical attributes. In third step, triangular membership functions are used to calculate fuzzy sets. Finally, predictive

FDT algorithm is presented. In the following subsections several basic concepts involved in predictive FDT are discussed.

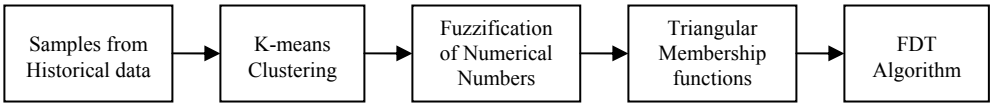


Figure 4.3: Procedure for predictive fuzzy decision tree classification method

4.6.1 Centroids of Fuzzy Sets (*k*-means Algorithm)

In the first step of fuzzy decision tree classification method, find k centers for different sets of static samples from stock market historical data. *K*-means (MacQueen, 1967) is the unsupervised learning algorithms that solve the well known clustering problem. The procedure follows a simple and easy way to classify a given data set through a certain number of clusters (assume k clusters) fixed a priori. The main idea is to define k centroids, one for each cluster. These centroids should be placed in a cunning way because of different location causes different result. So, the better choice is to place them as much as possible far away from each other. The next step is to take each point belonging to a given data set and associate it to the nearest centroid. When no point is pending, the first step is completed and an early group age is done. At this point need to re-calculate k new centroids as centers of the clusters resulting from the previous step. After having these k new centroids, a new binding has to be done between the data set points and the nearest new centroid. A loop has been generated. As a result of this loop it may notice that the k centroids change their location step by step until no more changes are done. In other words centroids do not move any more.

Finally, this algorithm aims at minimizing an *objective function*, in this case squared error function. The objective function

$$J = \sum_{j=1}^{k} \sum_{i=1}^{n} \| x_i^{(j)} - c_j \|^2 \tag{4.8}$$

where $\| x_i^{(j)} - c_j \|^2$ is a chosen distance measure between a data point $x_i^{(j)}$ and the cluster centre c_j, $\| \, . \, \|$ is an indicator of the distance of the *n* data points from their respective cluster centers. The algorithm to find the center for each equal pattern is composed of the following steps (Figure 4.4):

Although it can be proved that the procedure will always terminate, the *k*-means algorithm does not necessarily find the most optimal configuration, corresponding to the global objective function minimum. The algorithm is also significantly sensitive to the initial randomly selected cluster centers. The k -means algorithm can be run multiple times to reduce this effect.

1. *Place k points into the space represented by the objects that are being clustered. These points represent initial group centroids.*
2. *Assign each object to the group that has the closest centroid.*
3. *When all objects have been assigned, recalculate the positions of the k centroids.*
4. *Repeat Steps 2 and 3 until the centroids no longer move. This produces a separation of the objects into groups from which the metric to be minimized can be calculated.*

Figure 4.4: K-means algorithms to find centers of the equal patterns

An example:

Suppose that there are n sample feature vectors $x_1, x_2, ..., x_n$ all from the same class, and we know that they fall into k compact clusters, $k < n$. Let m_i be the mean of the vectors in cluster i. If the clusters are well separated, we can use a minimum-distance

classifier to separate them. That is, it can be said that x is in cluster i if $\| x - m_i \|$ is the minimum of all the k distances (as shown in Figure 4.5). This suggests the following procedure for finding the k-means:

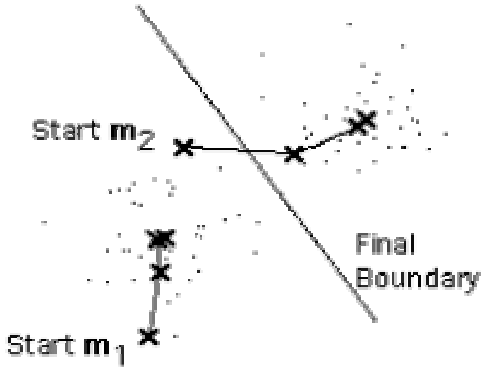


Figure 4.5: Centroids between two clusters

Start

make initial guesses for the means $m_1, m_2, ..., m_k$

until there are no changes in any mean

use the estimated means to classify the samples into clusters

for i from 1 to k

Replace m_i with the mean of all the samples for cluster i

end_for

end_until

End

Here is an example showing how the means m_1 and m_2 move into the centers of two clusters.

K-means is a consistent method to fuzzifying the data for all cases. When an attribute is categorical, the fuzzification is quite straightforward. Each of the possible values of the attribute has been just treated as fuzzy subset. In this case, the membership value in a fuzzy subset is either 0 or 1. For numerical attributes, *k*-means clustering algorithm to

cluster the attribute values in different sizes of clusters representing linguistic terms $T_1, T_2, T_3, ..., T_n$. The choice of the number of clusters is arbitrary though; the notation guided us that a value can typically be thought of as being low, average and high. Memberships have been generated based on a triangular membership function and the 3 cluster centers ($a_1, a_2, a_3, ..., a_n$ with $a_1 < a_2 < a_3 < ... a_n$) are obtained through *k*-means clustering. The following sections present the explanation of these clusters with example and how these clusters are used in triangular membership functions.

4.6.2 Fuzzification of Numerical Number

Fuzzification is a process of fuzzifying numerical numbers into linguistic terms, which is often used to reduce information overload in human decision-making process. Linguistic terms are simple forms of fuzzy values but generally their membership functions are unknown and need to be determined.

Given a fuzzy set of records, D, each of which consists of a fuzzy set of attributes $I = \{a_1, a_2, ...a_n\}$, where $a_r, r = 1, ..., n$, can be quantitative or categorical. Consider a set of examples $\{e_1, e_2, ..., e_N\}$, which is defined as the universe of discourse X (in short X is denoted by $\{1, 2, ..., N\}$). Let $T^{(1)}, ..., T^{(n)}$ and $T^{(n+1)}$ be a set of fuzzy attributes where $T^{(n+1)}$ denotes a classification attribute. Each fuzzy attribute $T^{(j)}$ consists of a set of linguistic terms $L(T^{(j)}) = \{L_1^{(j)}, ..., L_{mj}^{(j)}\} (j = 1, 2, ..., n+1)$. All linguistic terms are defined on the same universe of discourse X. The value of the *ith* example e_i with respect to the *jth* attribute, denoted by μ_{ij}, is a fuzzy set defined on $L(T^{(j)})(i = 1, ..., N, j = 1, 2, ..., n+1)$. In other words, fuzzy set μ_{ij} has a form of $\mu_{ij}^{(1)} / L_1^{(j)} + \mu_{ij}^{(2)} / L_2^{(j)} + ... + \mu_{ij}^{(mj)} / L_{mj}^{(j)}$ where $\mu_{ij}^{(k)}$ denote the corresponding membership degree $k = 1, 2, ..., m_j$. The fuzzy sets and its linguistic terms are as follows:

C_{P_O} = Change in price of open = $\{Low, Average, High\}$

C_{P_C} = Change in price of close = $\{Low, Average, High\}$

C_V = Change in volume = $\{Low, Medium, High\}$

P_{OP} = Oil Price = $\{Decrease, Stable, Increase\}$

P_{ND} = Natural Disaster = $\{No, Yes\}$

P_{PS} = Political Situation = $\{Clear, NotClear\}$

P_{NP} = New Policies = $\{LowAffect, NoAffect, HighAffect\}$

P_{NB} = New Budget = $\{Fair, NotFair\}$

P_{OC} = Other Companies = $\{Low, Medium, High\}$

P_{LSM} = Local Stock Markets = $\{LowAffect, NoAffect, HighAffect\}$

P_{OWS} = Overall World Situation = $\{Bad, Good\}$

Ps = Primary Signals = $\{Down, Hold, Up\}$

Fuzzy sets can have a variety of shapes. However, a triangle or a trapezoid can often provide an adequate representation of the expert knowledge and at the same time significantly simplifies the process of computation. Figure 4.6(a)-(d) represents an example of set of 4 linguistic terms (oil price, change in close, local stock market and primary signals) for fuzzy attribute. Horizontal direction presents universe of discourse and vertical direction presents degree of membership $[0,1]$.

4.6.3 Triangular Membership Function

Let X be a given data set, which is clustered into k linguistic terms $T_j, j = 1,2,...,k$. For simplicity, it is assumed that the type of membership to be triangular as follows:

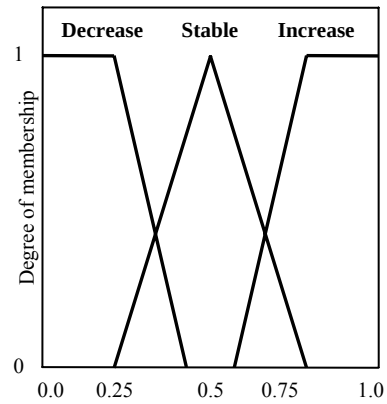


Figure 4.6(a): Linguistic terms for oil price

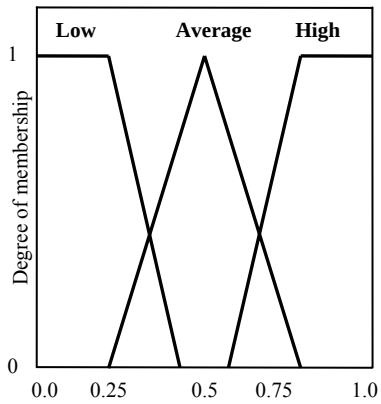


Figure 4.6(b): Linguistic terms for change in close price

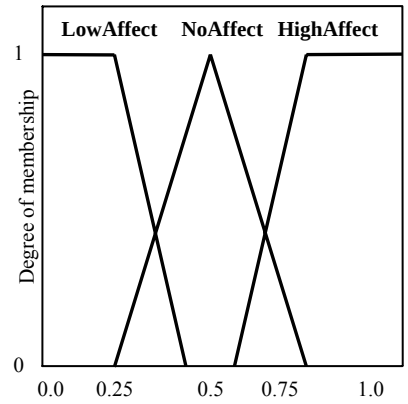


Figure 4.6(c): Linguistic terms for local stock market

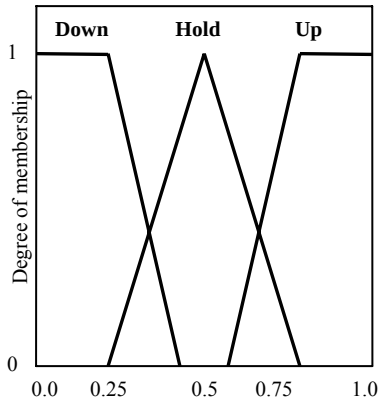


Figure 4.6(d): Linguistic terms for primary signal

$$T_1(x) = \begin{cases} 1 & x \le a_1 \\ \dfrac{a_2 - x}{a_2 - a_1} & a_1 < x < a_2 \\ 0 & x \ge a_2 \end{cases} \tag{4.9}$$

$$T_k(x) = \begin{cases} 1 & x \ge a_k \\ \dfrac{(x - a_{k-1})}{(a_k - a_{k=1})} & a_{k-1} < x < a_k \\ 0 & x \le a_{k-1} \end{cases} \tag{4.10}$$

$$T_j(x)=\begin{cases}0 & x \geq a_{j+1} \\ (a_{j+1}-x)/(a_{j+1}-a_j) & a_j \leq x < a_{j+1} \\ (x-a_{j-1})/(a_j-a_{j-1}) & a_{j-1} \leq x < a_j \\ 0 & x < a_{j-1}\end{cases} \qquad 1 < j < k \tag{4.11}$$

The only parameters to be determined are the k centers $\{a_1, a_2, ..., a_n\}$. An effective method to determine these centers is fuzzy clustering based on *k*-means as described in section 4.6.1.

Consider the numerical attribute "change in open" of the group of examples as shown in Table 4.3 by choosing k=3 and suppose the learning rate $\alpha = 0.02$, three values of center a_1-0.02, $a_2 = -0.13$, a_3=0.01 after 240 time of iteration. The membership functions for attribute "change in open" with linguistic terms $\{Low, Average, High\}$ are described as follows

$$Low = T_1(x) = \begin{cases}1 & x \leq -0.02 \\ \dfrac{-0.13-x}{-0.13-(-0.02)} & -0.02 < x < -0.13 \\ 0 & x \geq -0.13\end{cases}$$

$$Average = T_2(x) = \begin{cases}0 & x \geq 0.01 \\ (0.01-x)/(0.01-(-0.13)) & -0.13 \leq x < 0.01 \\ (x-(-0.02))/((-0.13)-(-0.02)) & -0.02 \leq x < -0.13 \\ 0 & x < -0.02\end{cases} \qquad 1 < j < k$$

$$High = T_3(x) = \begin{cases}1 & x \geq 0.01 \\ \dfrac{(x-(-0.13))}{(0.01-(-0.13))} & -0.13 < x < 0.01 \\ 0 & x \leq -0.13\end{cases}$$

Table 4.4: After training real time examples of stock market with fuzzy representation

No	C_{P_O}			P_{ND}		P_{NB}		P_{OWS}		P_{LSM}			Primary Signal		
	Low	**Med**	**High**	**No**	**Yes**	**Fair**	**NotFair**	**Bad**	**Good**	**Low Affect**	**Med Affect**	**High Affect**	**Down**	**Hold**	**Up**
1	0.10	0.55	0.45	0.45	0.55	0.30	0.70	0.94	0.06	0.00	0.14	0.86	0.00	0.53	0.47
2	0.03	0.10	0.87	0.85	0.15	0.67	0.33	0.88	0.22	0.58	0.25	0.17	0.00	0.67	0.33
3	0.71	0.10	0.19	1.00	0.00	0.00	1.00	0.70	0.30	0.81	0.19	0.00	0.49	0.33	0.18
4	0.09	0.70	0.21	1.00	0.00	0.20	0.80	0.27	0.73	0.00	1.00	0.00	0.25	0.07	0.68
5	0.42	0.30	0.48	0.92	0.08	0.46	0.54	0.09	0.91	0.73	0.27	0.00	0.30	0.20	0.50
6	0.70	0.00	0.30	0.42	0.58	0.70	0.30	0.94	0.06	0.12	0.00	0.88	0.30	0.70	0.00
7	0.00	0.00	1.00	0.54	0.46	0.85	0.15	0.82	0.18	0.00	0.00	1.00	0.41	0.59	0.00
8	1.00	0.10	0.00	0.60	0.40	0.28	0.72	0.00	1.00	0.46	0.33	0.23	0.16	0.40	0.46
9	0.43	0.11	0.46	0.67	0.33	0.05	0.95	1.00	0.00	0.10	0.00	0.90	0.95	0.00	0.05
10	0.45	0.55	0.00	0.91	0.09	0.32	0.68	0.50	0.50	0.00	0.20	0.80	0.69	0.20	0.11
11	0.00	0.69	0.31	0.20	0.80	0.17	0.83	0.44	0.56	0.00	0.67	0.33	0.80	0.20	0.00
12	0.00	1.00	0.00	0.80	0.20	0.48	0.52	1.00	0.00	0.00	1.00	0.00	0.00	1.00	0.00
13	0.20	0.80	0.00	0.51	0.49	0.24	0.76	0.20	0.80	0.00	0.25	0.75	0.49	0.00	0.51
14	0.41	0.38	0.21	0.78	0.22	1.00	0.00	0.82	0.18	0.74	0.20	0.06	0.00	0.82	0.18
15	0.50	0.30	0.20	0.54	0.46	0.18	0.82	0.19	0.81	0.00	0.64	0.36	0.00	0.54	0.46
16	0.50	0.10	0.40	0.41	0.59	0.80	0.20	0.98	0.00	0.00	0.21	0.79	0.00	0.80	0.20
17	0.05	0.30	0.65	0.50	0.50	0.60	0.40	0.05	0.95	0.25	0.00	0.75	0.65	0.00	0.35
18	0.34	0.35	0.31	0.38	0.62	0.40	0.60	0.82	0.18	0.74	0.18	0.08	0.20	0.52	0.20
19	0.50	0.44	0.01	0.94	0.06	0.15	0.85	0.23	0.77	0.00	0.64	0.36	0.34	0.10	0.56
20	0.10	0.01	0.89	0.10	0.90	0.30	0.70	0.98	0.02	0.21	0.10	0.79	0.10	0.30	0.60

It is obvious that the three linguistic terms can be described as Low, Average and High. The second column of Table 4.4 shows the membership degrees of the attribute Change in open belonging to the three membership functions. When the values of an attribute are fuzzy, the values can be written as one of the two forms: simple linguistic terms and membership functions. The information provided in a membership function is more concrete than those provided in linguistic terms but the meaning of membership function is not clear. Linguistic terms are simple forms of fuzzy values but their degree of membership is unknown and need to be determined. Degree of membership for selective 5 attributes by using the Table 4.2 is determined as in Table 4.4.

4.6.4 Predictive Fuzzy Decision Tree Algorithm

Let X represent a discrete universe of discourse, $F(X)$ denote the set of all fuzzy subsets defined on X. For $X = \{e_1, e_2, ..., e_N\}$ and $T \in F(X)$, T can be represented as $T = T(e_1)/e_1 + ... + T(e_N)/e_N$, and $M(T) = \sum_{i=1}^{N} T(e_i)$ denote the cardinality of T (Yeung *et al*., 1999; Wang *et al*., 2001).

By using the same notations in section 4.6.2, consider a test node S having n attributes $T^{(1)}, T^{(2)}, ..., T^{(n)}$ to be selected. For each $k(1 \le k \le n)$, the attribute $T^{(k)}$ takes m_k fuzzy subsets (linguistic terms), $L_1^{(k)}, L_2^{(k)}, ..., L_{mk}^{(k)}$. $T^{(n+1)}$ denotes the classification attribute, taking values $L_1^{(n+1)}, L_2^{(n+1)}, ..., L_m^{(n+1)}$. For each attribute value (fuzzy subset), $L_i^{(k)}(1 \le k \le n, 1 \le i \le m_k)$, its relative frequencies concerning the j^{th} fuzzy class $L_i^{(n+1)}(1 \le j \le m)$ at the considered nonleaf node S is defined as

$$p_{ij}^{(k)} = M(L_i^{(k)} \cap L_j^{(n+1)} \cap S) / M(L_i^{(k)} \cap S) \tag{4.12}$$

The weight of the *ith* value $L_i^{(k)}$ is defined as

$$w_i = M(S \cap L_i^{(k)}) / \sum_{j=1}^{mk} M(S \bigcap L_j^{(k)}) \tag{4.13}$$

Definition 1:

Let $\mu_{ik} = \mu_{ik}^{(1)}, ..., \mu_{ik}^{(j)}, ..., \mu_{ik}^{(mj)}$ be the value of the *ith* example with respect to the *kth* attribute, $w_{ik} = \mu_{ik}^{(1)}, ..., \mu_{ik}^{(j)}, ..., \mu_{ik}^{(mj)}$, v_i be the value of the *ith* example with respect to the classification $(1 \le i \le N, 1 \le k \le n)$, i.e., v_i is a fuzzy set define on $L_1^{(n+1)}, L_2^{(n+1)}, ..., L_m^{(n+1)}$. (the set of linguistic terms of the classification attribute $T^{(n+1)}$), SM be a selected similarity measure, $\lambda_{ip}^{(j)} = \wedge_{q \ne k}(SM(\mu_{iq}, \mu_{pq})) \wedge SM(w_{ik}, w_{pk})$,

(where $\wedge$ denotes minimum, $i \neq p$), and $\sigma_{ip} = SM(v_i, v_i)(i \neq p)$. Then, for the *kth* attribute $T^{(k)}(1 \leq k \leq n)$, the degree of importance of its *jth* linguistic term $L_j^{(k)}(1 \leq j \leq m_k)$ contributing to the classification is defined as

$$\theta_j^{(k)} = \frac{1}{N(N-1)} \sum_i \sum_{p \neq i} g^+(\lambda_{ip}^{(k)} - \sigma_{ip}), \tag{4.14}$$

where $$g^+(x) = \begin{cases} x & x > 0 \\ 0 & x \leq 0 \end{cases}.$$

Definition 2:

The averaged degree of importance of the *kth* attribute $T^{(k)}$ is defined as $P_k = \sum_{j=1}^{mk} w_j \theta_j^{(k)}$, in which w_j is defined by equation (4.12).

Proposed approach aims to search for an attribute such that its average degree of importance contributing to the classification attains a maximum, i.e., selecting an integer $k_\circ$ (the $k_\circ th$ attribute) so that $P_{k\circ} = Max_{1 \leq k \leq n} P_k$ where P_k is given by definition 2. According to the above heuristic, a FDT can be generated by using training a set of data. Before training the initial data, the α-cut is usually used for the initial data (Wang *et al*., 2001). The purpose of using α cut is to reduce the fuzziness. The α-cut of a fuzzy set L is defined as

$$L_\alpha(x) = \begin{cases} L(x) & L(x) \geq \alpha \\ 0 & L(x) < \alpha \end{cases} \tag{4.15}$$

When α is set in the interval $(0,0.5]$, procedure for generating a predictive FDT is described as follows:

Step 1.

Given the cut-standard $\alpha(\alpha \in (0,1))$ and the leaf-standard $\beta(\beta \in (0,1))$.

Step 2.

Use α to cut the initial data set. More specifically, each membership degree less than α is changed to 0 and all others remains unchanged.

Step 3.

Consider the root node $(1,1,...,1)$ as the first candidate node.

Step 4.

Randomly select a non-leaf candidate node. For any attribute which has not been used in forefather nodes of this node, compute P_k.

Step 5.

Select an attribute with average maximum importance to the classification (given by definition 2) as the expanded attribute. According to the expanded attribute, generate son-nodes of the non-leaf candidate node. These son-nodes are considered as new candidate nodes.

Step 6.

For each of these son-nodes, if the relative frequency $P_{ij}^{(k)}$ (given by equation (4.12)) of a certain class exceeds β or the membership sum of the considered son-node is less than a small positive number, then this son-node is labeled leaf.

Step 7.

If all nodes are leaves then stop, else go to Step 4.

Figure 4.7 shows the predictive FDT by using proposed algorithm to train Table 4.2. The weighted fuzzy production rules are extracted in the next chapter.

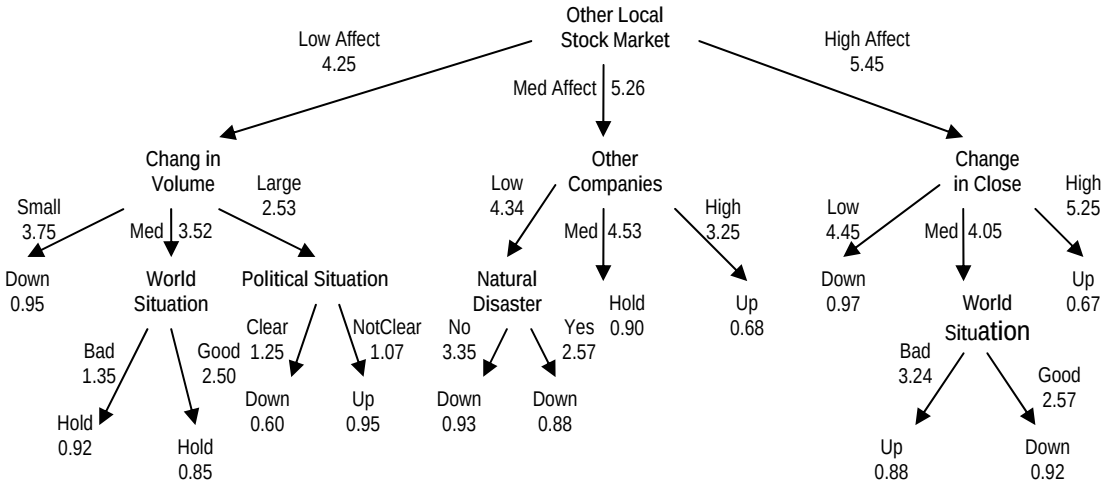


Figure 4.7: Fuzzy Decision Tree by using predictive FDT algorithm to train table 4.2

4.7 Summary

In this chapter, all steps in constructing predictive FDT algorithm are presented. Before applying the FDT algorithm, first we use the *k*-means clustering algorithm to find the centers for equal sets of patterns from historical stock market data. In fuzzification method, all these static sets of samples are converted into linguistic terms by triangular fuzzy membership functions. In fuzzy decision tree, which is based on average minimum fuzzy classification entropy, every node shows the fuzzy sets and every arc shows linguistic terms and every leaf is for classification attribute. The path from root to leaf shows one fuzzy rule. In next chapter, fuzzy rules are extracted from fuzzy decision tree for classification predicates.

CHAPTER 5

FUZZY REASONING METHOD BASED ON SIMILARITY TECHNIQUE

5.1 Introduction

The important factors to be considered in domain knowledge representation are the representability, flexibility, integrity, and expendability, etc., of the stored knowledge. Weighted fuzzy production rules (WFPRs) representation provides us with such advantages, which is used to capture and represent imprecise vague or fuzzy knowledge as well as the degree of importance of each proposition contributing to a conclusion in a given production rule. In this chapter, two important steps are described as 1) WFPRs are extracted from predictive FDT and 2) significant WFPRs are mined by using similarity-based fuzzy reasoning method. In similarity-based fuzzy reasoning method, several knowledge parameters are considered such as weight and certainty factor to enhance the representation power of WFPRs. Weights are assigned to every proposition in the antecedent part and certainty factor is calculated to each proposition of the fuzzy production rule. Finally, high learning WFPRs rules are collected for the prediction of stock market data.

5.2 Weighted Fuzzy Production Rules (WFPRs)

The WFPRs generation from fuzzy decision tree is an extended form of fuzzy production rules (FPR) proposed by (Yeung *et al*., 1994). WFPRs defined here is similar to the conventional production rules with the exception that fuzzy values such as "fat" or "small" are allowed in the propositions. A weight is assigned to each proposition in the antecedent part, and a certainty factor is calculated for each rule.

A WFPR is defined as: R: IF a THEN c ($CF = \mu$), Th, w, where $a = \langle a_1, a_2, ..., a_n \rangle$ is the antecedent portion which comprises of one or more propositions connected by either "AND" or "OR". Each proposition $a_i (1 \le i \le n)$ can have the format "x is f_{ai}", where f_{ai} is an element of a set of fuzzy sets F = { $f_1, f_2, ..., f_n$ }. The consequent of the rule c can be expressed, as "x is f_c", where f_c is also an element of F. The parameter μ is the certainty factor of the rule R and it represents the strength of belief of the rule. The symbol $Th = \langle \lambda_1, \lambda_2, ..., \lambda_n \rangle$ represents a set of threshold values specified for the proposition in the antecedent a. The set of weights assigned to the propositions $\langle a_1, a_2, ..., a_n \rangle$ is given by $w = \langle w_1, w_2, ..., w_n \rangle$. The weight w_i of a proposition a_i shows the degree of importance of a_i contributing to the consequent c when comparing to other proposition a_j, for $j \ne i$. It is obvious that when there is only one proposition in the antecedent of fuzzy production rules, the weight w_i is meaningless. The set of weight w assigned to each proposition in the antecedent is referred as local weights. Another important concept called global weight, which could be assigned to each rule in an inference path, is fully explored in (Yeung and Tsang, 1995).

In general WFPRs are categorized into three types, which are defined as follows:

Type 1: A Simple Fuzzy Production Rule

R: IF a THEN c ($CF = \mu$), λ, w, for this type of rule, since there is only one proposition 'a' in the antecedent, the weight w is meaningless.

Type 2: A Composite Fuzzy Conjunction Rule

R: IF a_1 AND a_2 THEN c ($CF = \mu$), $\lambda_1, \lambda_2, w_1, w_2$,

Type 3: A Composite Fuzzy Disjunction Rule

R: IF a_1 OR a_2 THEN c ($CF = \mu$), $\lambda_1, \lambda_2, w_1, w_2$,

For both types 2 and 3, λ_i is the threshold value for a_i, and w_i is the weight assigned to a_i. Some authors do not assign a certainty factor to a FPR while others ignore the weight and the threshold value assigned to each proposition in the antecedent. We considered that the capturing of fuzzy knowledge using fuzzy production rule with weights and threshold values plays an important role in real world applications. Hence the weight (degree of importance), the threshold value as well as the certainty factor have been taken into account.

5.2.1 Weighted Fuzzy Production Rules with Single Antecedent

R: IF A THEN C, ($CF = \mu$), Th, W, e.g., if a_1 then C, ($CF = \mu$), $Th = \{\lambda_{a_1}\}$, $W = \{w_1\}$. C is represented as a "concluded disorder" in Chen's Diagnosis problem (Chen, 1988, 1994) or a consequent in other problems.

Given four cases of facts:

Case 1: $A' = A$

Case 2: $A' =$ very A

Case 3: $A' =$ more or less A

Case 4: $A' =$ not A

What conclusion $C^{'}$ can be drawn? In order to draw the conclusion $C^{'}$, similarity-based fuzzy reasoning algorithm is analyzed for all of the possible cases and select the most accurate case which is best suited for classification of stock market prediction.

5.2.2 Weighted Fuzzy Production Rules with Multiple Antecedents

If the antecedent portion or consequence portion of fuzzy production rule contains "AND" or "OR" connectors, then it is called a composite fuzzy production rule. According to Looney (1987), the composite fuzzy production rule can be distinguished into the following rule-types:

Type 1: IF a_{j1} AND, a_{j2} AND…AND a_{jn} THEN a_k $(CF = \mu_i)$

Type 2: IF a_j THEN a_{k1} AND ,a_{k2} AND…..AND a_{kn} $(CF = \mu_i)$

Type 3: IF a_{j1} OR, a_{j2} OR…OR a_{jn} THEN a_k $(CF = \mu_i)$

Type 4: IF a_j THEN a_{k1} OR ,a_{k2} OR…..OR a_{kn} $(CF = \mu_i)$

In proposed algorithm, multiple propositions connected by "AND" are used,
R: IF A THEN C, $(CF = \mu)$, Th, W, e.g., if a_1 AND a_2 THEN C, $(CF = \mu)$,
$Th = \{\lambda_{a1}, \lambda_{a2}\}$, $W = \{w_1, w_2\}$ $A = \langle a_1, a_2 \rangle$, a_1 AND a_2 are connected by "AND"
consider the following four cases:

Case 1: $A^{'} = \langle a_1', a_2' \rangle = A = \langle a_1, a_2 \rangle$

Case 2: $A^{'} =$ very $A = \langle$ very a_1, very $a_2 \rangle$ $A \langle a_1', a_2' \rangle \langle a_1', a_2' \rangle \langle a_1', a_2' \rangle$

Case 3: $A^{'} = \langle a_1', a_2' \rangle =$ more or less $A = \langle$ more or less a_1, more or less $a_2 \rangle$

Case 4: $A^{'} = \langle a_1', a_2' \rangle =$ not $A = \langle$ not a_1, not $a_2 \rangle$

5.2.3 Transformation of WFPRs from FDT

The transformation from the tree to WFPRs is described as follows.

1. Each path of branch from the root to a leaf can be converted into a rule. The antecedent of the rule represents the attributes on the passing branches from the root to the leaf and the consequent of the rule represents the cluster labeled at the leaf node.
2. The degree of importance of each attribute value (linguistic term) of the expanded attribute, $\theta_j^{(k)}$, is regarded as the value of the weight w_{ij} of the corresponding proposition of the converted rule.
3. For the converted rule R_i :

   $IF < A_{i1}\ and\ A_{i2} ... and\ A_{in} > THEN\ B_i\ (CF, w_{i1}, w_{i2}, ..., w_{in})$ in which the weight w_{ij} are obtained from step (2) and the certainty factor is defined as

$$CF_i = \sum_{i=1}^{N} (\wedge_{j=1}^{n} (w_{ij} A_{ij}(k) \wedge B_i(k) / \sum_{i=1}^{N} (\wedge_{j=1}^{n} (w_{ij} A_{ij}(k))) \quad (5.1)$$

   where A_{ij} and B_i are fuzzy sets on $\{1,2,..., N\}$ and $\wedge$ denotes the minimum.

5.3 Knowledge Parameters

Several knowledge parameters such as certainty factor, local weight, threshold value and global weight should be incorporated into the WFPRs

1) Certainty factor:
   One type of knowledge parameter called certainty factor was introduced in (Buchanan and Shortliffe, 1984) to represent the degree of certainty of a

proposition or the entire rule. Sometimes it is called MYCIN-linked certain factor (Buchanan and Shortliffe, 1984). Subsequently many researchers used this kind of knowledge parameter to investigate WFPRs with uncertainty.

2) Local weight:

The method used in MYCIN to calculate the certain factor of the consequent suffers from the shortcomings that all propositions in the antecedent part are assumed to be equally important. To overcome this shortcoming another type of knowledge parameter related to WFPRs called local weight which indicates the degree of importance of each proposition in the antecedent was proposed by a number of researchers (Buchanan and Shortliffe, 1984; Yeung *et al*., 1994).

3) Threshold value:

For a given set of WFPRs, the observed fact may approximately match each of the set of rules with different degrees. That will result in some rules being misfired. To overcome this problem, several researchers employed a kind of knowledge parameter called threshold value which is assigned to each proposition (Yeung and Tsang, 1997a) or assigned to entire rule for the purpose of not only obtaining a reasonable result of an approximate reasoning method but also preventing or reducing rule miss-firing

4) Global weight:

This type of knowledge parameter was proposed in (Yeung and Tsang, 1997a) to indicate that a number of rules executed in an reference path leading to a specific goal, or the same rule employed in various inference paths leading different goals have different degrees of importance.

5.4 Similarity-Based Fuzzy Reasoning Methods

In this section, some similarity-based fuzzy reasoning methods are being reviewed. The advantage of such methods includes: 1) the ability to assign certainty factors, threshold values and weights to FPRs; and 2) the similarity computation.

5.4.1 Approximate Analogical Reasoning Schema (AARS)

Turksen and Zhong (1989, 1990) mentioned that although Zadeh's CRI is simple in calculation, the matrix operations on the membership functions are not clear and the outcome may not be the desirable one. Hence a similarity-based fuzzy reasoning method is proposed.

A simple rule with a threshold value $\lambda_\circ$ for a given fact is given below. Note here that the certainty factor and weight have not adopted.

Rule: If A then C, $\lambda_\circ$

Fact A'

Conclusion: C'

The ARRS modifies the consequent C based on the similarity (closeness) between the fact A' and the antecedent A. If the degree of similarity measure is greater than the predefined threshold value $\lambda_\circ$, the rule is fired and the consequent is deduced by some modification techniques.

Turksen and Zhong (1989, 1990) mentioned five good distance measures (DM). One of them is the Disconsistency Measure

$$D_1(A, A') = 1 - \sup_{x \in X} \mu_{A \cap A'}(x) \tag{5.2}$$

The other are the Hausdorff measure (∞), Hausdorff measure ($*$), Kaufman and Gupta measure (∞), and Gupta measure ($*$). For the example used in (Turksen and Zhong, 1989, 1990), the authors used the Euclidean distance measure to calculate the distance between A and $A^{'}$, viz.,

$$D_2(A^{'}, A) = \left[\frac{\sum_{i=1}^{n} [\mu^{'}{}_A(x) - \mu_A(x)]^2}{n} \right]^{1/2} \tag{5.3}$$

The similarity is then defined as

$$S_{AARS} = (1 - DM)^{-1} \tag{5.4}$$

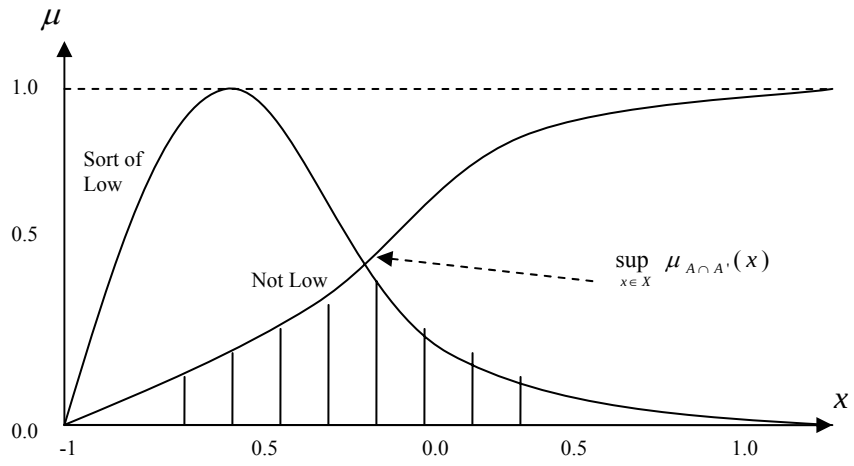


Figure 5.1: Disconsistency measure

If $S_{AARS} \geq \lambda \circ$, the rule is fired and the consequent is modified by a modification function which could appear in one of the two forms:

1) more or less form:

$$C^{'} = \min\{1, C / S_{AARS}\} \tag{5.5}$$

2) membership value reduction form:

$$C' = C * S_{AARS} \tag{5.6}$$

In above ARRS method, authors have proposed a method to process a composite FPR. If a rule with multiple antecedents connected by "OR" is used, it can be split into multiple simple rules. On the other hand, if a rule with multiple antecedents connected by "AND" is used, the overall similarity S_{AARS} is obtained by averaging the similarities of $D_2(a_1', a_1)$ and $D_2(a_2', a_2)$. Since $A'\langle a_1', a_2' \rangle = A = \langle a_1, a_2 \rangle$, the overall $S_{AARS} = 1$, therefore, $C' = C$. On the other hand, $D_2(a_1', a_1) < 1$ and $D_2(a_2', a_2) < 1$ when $A'\langle a_1', a_2' \rangle = \text{very } A = \langle \text{very } a_1, \text{very } a_2 \rangle$, the overall $S_{AARS} < 1$ and the consequent C' is equal to "very C". $\langle$ more or less a_1, more or less $a_2 \rangle$, $D_2(a_1', a_1) < 1$, $D_2(a_2', a_2) < 1$, and hence the overall $S_{AARS} < 1$. The consequent C is equal to "more or less C". For the case $A' = \text{not } A$, the similarity measure taken between A' and A is lead to rule miss firing if a single threshold value is assigned to the entire rule.

5.4.2 Function T (FT) Method

Chen (1994) has proposed enhanced form of Chen's (1988) method based on the function $T(a_i, a_i') = 1 - | a_i - a_i' |$. Furthermore, the FPRs used to represent medical diagnosis knowledge have been enhanced to include weight for each proposition in the antecedent.

The formulation of FPRs can be found in appendix F. Let $A = < a_1, a_2, ..., a_n >$ and $A' = < a_1', a_2', ..., a_n' >$ be the vectors representing the values of the fuzzy quantifiers in the symptoms and the values of the fuzzy quantifiers provided by the user, respectively. With the assignment of a weight to each proposition in the antecedent of the FPR, the

similarity between the two vectors A and A' is now measured by the similarity function $S_{FT}(A, A', W) \in [0,1]$ defined as

$$S_{FT}(A, A', W) = \sum_{i=1}^{n} \left[T(a_i, a_i') * \frac{w_i}{\sum_{k=1}^{n} w_k} \right] \tag{5.7}$$

The larger the value of $S_{FT}(A, A', W)$, the higher the similarity between the vectors A' and A. If $S_{FT}(A, A', W) \geq \lambda$, the rule can be fired and the value of the consequent $C = S_{FT}(A, A', W) * \mu$. Otherwise the rule cannot be fired. In this method, the similarity measure function is very simple and straightforward which is considered to be acceptable. The order of the input parameters is also irrelevant. For the medical diagnostic problem, the method is applied to one-level reasoning only.

5.4.3 Degree of Subsethood (DS) Method

Yeung *et al*. (1994) have proposed a fuzzy reasoning method, which employs similarity measure, based on the degree of subsethood between the proposition in the antecedent and the given facts as proposed by (Kosko, 1992). This method has also been extended to include weights for the propositions in the antecedent (Yeung *et al*., 1994). Recall that the degree of subsethood is defined as

$$S_{DS}(a_i', a_i) = \frac{M(a_i' \cap a_i)}{M(a_i')} \tag{5.8}$$

where $M(a_i')$ is the sigma-count of a_i' i.e., the size or cardinality of a_i' and $M(a_i' \cap a_i)$ is the sigma-count of the intersection of a_i' and a_i. It is observed that $S_{DS}(a_i', a_i)$, which is the degree of subsethood of a_i' in a_i, differs from $S_{DS}(a_i', a_i)$, which is the degree of subsethood of a_i in a_i', and $0 \leq S_{DS}(a_i', a_i) \leq 1$. For instance, if a_i' is a subset of a_i or

$a_i^{'} = a_i$, then $S_{DS}(a_i^{'}, a_i) = 1$, if a_i has all its fuzzy membership values equal to zero, then $S_{DS}(a_i^{'}, a_i) = 0$.

According to Yeung *et al*. (1994), the rule is executed if the computed value of the degree of subsethood $S_{DS}(a_i^{'}, a_i)$ is greater than or equal to the threshold value λ_i for all a_i. First weighted average degree of subsethood function S_w is defined as follows

$$S_w = \sum_{i=1}^{n} S_{w_i} = \sum_{i=1}^{n} \left[S_{DS}(a_i^{'}, a_i) * \frac{w_i}{\sum_{j=1}^{n} w_j} \right] \tag{5.9}$$

where $A = \langle a_1, a_2, ..., a_n \rangle, A^{'} = \langle a_1^{'}, a_2^{'}, ..., a_n^{'} \rangle, and$

$$S_{wi} = \left[S_{DS}(a_i^{'}, a_i) * \frac{w_i}{\sum_{j=1}^{n} w_j} \right] \quad for.i = 1, ..., n \tag{5.10}$$

Turksen *et al*. (1989; 1990) proposed two schemes for calculating the values of $C^{'}$. They are: 1) more or less form and 2) membership value reduction form. However Turksen did not mention how to choose the modification function. Furthermore, the certainty factor and the weight have not been taken into account. Yeung *et al*.'s DS method has adopted the "more or less form" with a modification to include the certainty factor and the $C^{'}$ given by

$$C^{'} = \min\{1, C / S_w * \mu\} \tag{5.11}$$

The following shows what $C^{'}$ can be drawn for each of the three cases:

Case1: The antecedent A has only one proposition. $S_w = S_{DS}(a_1^{'}, a_1) = S_{DS}(A^{'}, A)$

Since $w_1 / \sum_{j=1}^{n} w_j = 1$ with n=1 If $S_{DS}(a_i^{'}, a_i) \geq \lambda_{a1}$ then $C^{'} = \min\{1, C / S_w * \mu\}$

Case2: The antecedent A has two or more propositions connected by "AND"

$$S_w = \sum_{i=1}^{n} \left[S_{DS}(a_i^{'}, a_i) * \frac{w_i}{\sum_{j=1}^{n} w_j} \right] \tag{5.12}$$

if $S_{DS}(a_i^{'}, a_i) \geq \lambda_{ai}$, for all $(1 \leq i \leq n)$, then $C^{'} = \min\{1, C / S_w * \mu\}$

Case 3: The antecedent A has two or more propositions connected by "OR"

This rule can be split into n simple rules shown in case1, i.e.,

$S_w = S_{DS}(a_i^{'}, a_i)$ because for each $i = 1,....,n, \sum_{j=1}^{n} w_j$ is reduced to a single term

w_i and thus $w_i / \sum_{j=1}^{n} w_j$ becomes $w_i / w_i = 1$. If $S_{DS}(a_i^{'}, a_i) \geq \lambda_{ai}, (1 \leq i \leq n)$, then

$C^{'} = \min\{1, C / (\max(S_{w1,} S_{w2,} .... S_{wn}) * \mu)\}$.

5.4.4 Equality and Cardinality (EC) Method

Later two more similarity-based fuzzy reasoning methods were proposed by Yeung and Tsang (1997b) which calculate the similarity between A and $A^{'}$ using equality and cardinality. Let us denote such similarity measure as $S_{EC}(a_i^{'}, a_i)$, which is defined as

$$S_{EC}(a_i^{'}, a_i) = 1 - \frac{M(a_i^{'} \nabla a_i)}{M(a_i)} \tag{5.13}$$

where ∇ is the symmetrical difference of A and $A^{'}$ viz.

$\forall x \in X, \mu_{A \nabla A^{'}} = \mid \mu_{A}^{'}(x) - \mu_{A}(x) \mid$. Thus the equation can be expressed as

$$S_{EC}(a_i^{'}, a_i) = 1 - \frac{\sum_{x \in X} \mid \mu_{a_i^{'}}(x) - \mu_{a_i}(x) \mid}{\sum_{x \in X} \mu_{a_i}(x)} \tag{5.14}$$

It is observed that $S_{EC}(a_i^{'}, a_i)$, which is the degree of equality of $a_i^{'}$ in a_i, differs from $S_{EC}(a_i^{'}, a_i)$ and $0 \le S_{EC}(a_i^{'}, a_i) \le 1,$ for instance, if $a_i^{'} = a_i$, then $S_{EC}(a_i^{'}, a_i) = 1$. Again if $S_{EC}(a_i^{'}, a_i) \ge \lambda_{ai}$ for all a_i, the rule can be fired and the aggregated weighted average, AG_w is defined as

$$AG_w = \sum_{i=1}^{n} AG_{wi} = \sum_{i=1}^{n} \left[S_{EC}(a_i^{'}, a_i) * \frac{w_i}{\sum_{j=1}^{n} w_j} \right] \tag{5.15}$$

After the aggregated weighted average has been calculated, two modification functions are proposed to modify the consequent $C^{'}$. These two forms are also enhancements of those in (Turksen and Zhong, 1989, 1990)

1) Enhanced more or less form:

$$C^{'} = \min\{1, C/(AG_w * \mu)^{0.5}\} \tag{5.16}$$

2) Enhanced membership value reduction form:

$$C^{'} = C * (AG_w * \mu)^{0.5}\} \tag{5.17}$$

The following shows what $C^{'}$ can be drawn for each of the three cases:

Case 1: The antecedent A has only one proposition $AG_w = S_{EC}(a_1^{'}, a_1) = S_{EC}(A', A)$.

Since $w_1 / \sum_{j=1}^{n} w_j = 1$ with n=1

If $S_{DS}(a_1^{'}, a_1) \geq \lambda_{a1}$ then $C^{'} = \min\{1, C/(AG_w * \mu)^{0.5}\}$ or $C^{'} = C * (AG_w * \mu)^{0.5}$

Depending on whether to restrict or dilate the membership value of C

Case2: The antecedent A has two or more proposition connected by "AND"

$$AG_w = \sum_{i=1}^{n} (S_{EC}(a_i^{'}, a_i) * \frac{w_i}{\sum_{j=1}^{n} w_j}).$$

If $S_{DS}(a_i^{'}, a_i) \geq \lambda_{ai}$ for all $(1 \leq i \leq n)$, then $C^{'} = \min\{1, C/(AG_w * \mu)^{0.5}\}$ or

$C^{'} = C * (AG_w * \mu)^{0.5}$

Case3: The antecedent A has two or more propositions connected by "OR"

This rule can be split into n simple rules as shown in Case1, i.e.,

$AG_w = S_{EC}(a_i^{'}, a_i)$. Because for each $i = 1,...,n, \sum_{j=1}^{n} w_j$ is reduced to a single term

w_i, and becomes $w_i / w_i = 1$

If $\exists_i$ s.t. $S_{DS}(a_i^{'}, a_i) \geq \lambda_{ai}$ for all $(1 \leq i \leq n)$, then

$C^{'} = \min\{1, C / \max(AG_{w1}, AG_{w2}, ..., AG_{wi}) * \mu)^{0.5}\}$ or,

$C^{'} = C * \max(AG_{w1}, AG_{w2}, ..., AG_{wi}) * \mu)^{0.5}$, OR,

If $S_{DS}(a_i^{'}, a_i) \geq \lambda_{ai}$ for all $(1 \leq i \leq n)$, then

$C^{'} = \min\{1, C / \max(AG_{w1}, AG_{w2}, ..., AG_{wi}) * \mu)^{0.5}\}$ or,

$C^{'} = C * \max(AG_{w1}, AG_{w2}, ..., AG_{wi}) * \mu)^{0.5}$.

5.5 Fuzzy Reasoning Method

In order to compare all existing reasoning methods (Turksen and Zhong, 1989, 1990; Chen, 1988, 1994; Yeung *et al*., 1994; Yeung and Tsang, 1997b) under the same conditions and with a unified rule, we assume that the rule R has $CF = 1$, $W = \{1\}$, and the threshold value Th is small enough for the value to be fired for all cases. For Case 4, i.e., $A' = \text{not } A$, it does not expect the rule to be fired, but it is possible that rule miss-firing may result, i.e., the rule is fired with the consequent C' not equal to "not C" due to the assignment of a small threshold value. So the above rule R is equivalent to a conventional fuzzy production rule "IF A THEN C". The relationship between the fuzzy set and its modifiers such as "very", "more or less" and "not" satisfies the following relationship proposed by (Zadeh, 1987a, 1987b):

1) $a_i' = \text{very } a_i = a_i^2$ i.e., $\mu_{very\ a_i}(x) = [\mu_{ai}(x)]^2$

2) $a_i' = \text{more or less } a_i = a_i^{0.5}$ i.e., $\mu_{more,\text{or less}\ a_i}(x) = [\mu_{ai}(x)]^{0.5}$

3) $a_i' = \text{not } a_i = 1 - a_i$ i.e., $\mu_{not\ a_i}(x) = 1 - \mu_{ai}(x)$

In fuzzy reasoning algorithm first simple rule case has been considered, where only one observation A' and one simple rule in the form $R: A \rightarrow C$ is presented. The basic idea is to modify the consequent C of a simple rule $R: A \rightarrow C$ according to the closeness of an observation (fact) A' to the antecedent (pattern) A of the rule R. If they are close (similar) enough in comparison to a threshold, then the rule R can be fired and the consequent can be deduced by some modification technique to be shown later in the sequel.

Formally, their "closeness" is expressed as a Similarity Measure (SM), which in turn is obtained form a Distance Measure (DM). More specifically, SM presents a threshold $\lambda_\circ$ of the SM. Once the $\text{SM}(A', A) = \lambda$ between A' and A exceeds the threshold $\lambda_\circ$ the rule R is fired, that is construct a Modification Function (MF) based on

$\lambda : MF = f(\lambda),$ and use this MF to modify the right side C of the rule R to deduce a consequent C'. The selection of $\lambda_{\circ}$ above 0.5 and closer to 1 assures better similarity A and $A^{'}$. Hence it is believed that the specification of a $\lambda_{\circ}$ does improve the behavior of consequents.

5.5.1 Similarity Measure

The similarity measure between linguistic terms is defining their membership functions. In this section, similarity-based fuzzy reasoning method is presented, which calculates the similarity between A and A' using equality and cardinality as (Yeung and Tsang, 1997b). Let us denote such similarity measure as $S_{PR}(a_i^{'}, a_i)$, which is defined as

$$S_{PR}(a_i^{'}, a_i) = 1 - \frac{f(a_i^{'} \nabla a_i)}{f(a_i)} \tag{5.18}$$

Where ∇ is the symmetrical difference of A and $A^{'}$

viz. $\forall x \in X, \mu_{A \nabla A^{'}} = | \mu_A{}^{'}(x) - \mu_A(x) |$. Thus the equation can be expressed as

$$S_{PR}(a_i^{'}, a_i) = 1 - \frac{\sum_{x \in X} | \mu_{a_i^{'}}(x) - \mu_{a_i}(x) |}{\sum_{x \in X} \mu_{a_i}(x)} \tag{5.19}$$

5.5.2 Aggregated Weighted Average

It is observed that $S_{EC}(a_i^{'}, a_i)$, which is the degree of equality of $a_i^{'}$ in a_i, differs from $S_{EC}(a_i^{'}, a_i)$ and $0 \le S_{EC}(a_i^{'}, a_i) \le 1,$ for instance, if $a_i^{'} = a_i$, then $S_{EC}(a_i^{'}, a_i) = 1$. Again if $S_{EC}(a_i^{'}, a_i) \ge \lambda_{ai}$ for all a_i, the rule can be fired and the aggregated weighted average, AG_w is defined as

$$AG_w = \sum_{i=1}^{n} AG_{wi} = \sum_{i=1}^{n} \left[S_{EC}(a_i^{'}, a_i) * \frac{w_i}{\sum_{j=1}^{n} w_j} \right] \tag{5.20}$$

After the aggregated weighted average has been calculated, two modification functions are proposed to modify the consequent $C^{'}$. These two forms are also enhancements of those in (Turksen and Zhong, 1989, 1990).

5.5.3 Modification Functions (MFs)

In fuzzy reasoning algorithm, a rule $R_j : A_j \rightarrow C_j$ is to be fired with the use of an MF that modifies the consequent C_j of the rule R_j. The MF is dependent on SM and its construction is subjective. That is one would if required adjust the form of the MFs based on experts experience or historical data so that the system can function as close to the real situation as possible. However this enables us to bypass the matrix operations of CRI (Computational Rule of Inference) it was first proposed by Zadeh (1973) to deduce a consequent. There could be many forms of modification functions two possible examples are presented here.

After the aggregated weighted average has been calculated, two modification functions are proposed (Yeung and Tsang, 1997b) to modify the consequent $C^{'}$.

(1) *More or Less Form*:

This is an analogy to the definition of linguistic hedge “more or less” (Zadeh, 1973). The induced consequent C'_j is obtained through

$$C'_j = \min\{1, C_j (AG_w * \mu)^{0.5}\} \tag{5.21}$$

Using the similarity transformation SM = $(1+DM)^{-1}$ the "more or less" form of the modification function can be simplified as follows

$$C'_j = \min\{1, C_j * (1 + DM)\} \tag{5.22}$$

Where * represents multiplication.

Figure 5.2 shows the effect of this form of MF. It is observed that additional uncertainty is

$$C'_j = C_j * SM \tag{5.23}$$

introduced with such a modification function, i.e., large value of distance measure increases the values of membership. Obviously, at the extreme case, i.e., when $A'_j = A_j$ for the rule R_j, they are indeed exactly matched, i.e., DM=0 and hence the consequent C_j is not altered based on the proposed schema. This is one of the advantages of similarity based over CRI; recall that the CRI generally produce a different result other than C_j i.e., $A_j \circ (A_j \rightarrow C_j) = C'_j \neq C_j$.

(*2*) *Membership Value Reduction Form*:

In some sense this is an imitation of CRI. It simply multiplies the consequent C_j by SM (A'_j, A_j) for a given observation A'_j, i.e.,

Figure 5.3 shows the effect of this form of modification function. It gradually reduces the membership value of C_j according to SM. In other words the smaller the SM greater the reduction of membership value of C_j, which imitates the min operator's contraction effect of CRI. However, it is avoided the max operator's expansion. At the extreme case, that is when $A'_j = A_j$, we have SM (A'_j, A_j)=1 and hence $C'_j = C_j$.

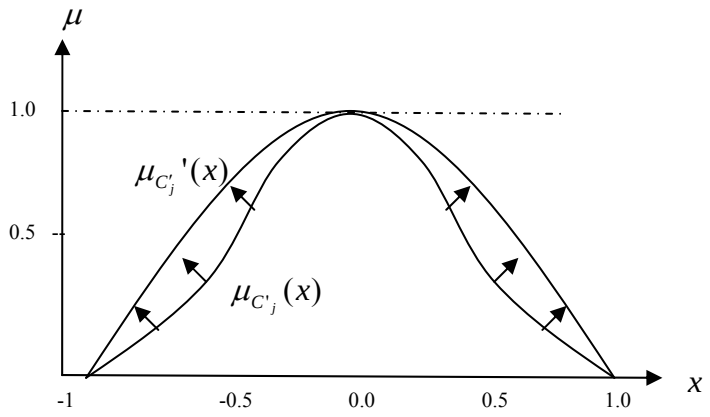


Figure 5.2: More or Less form

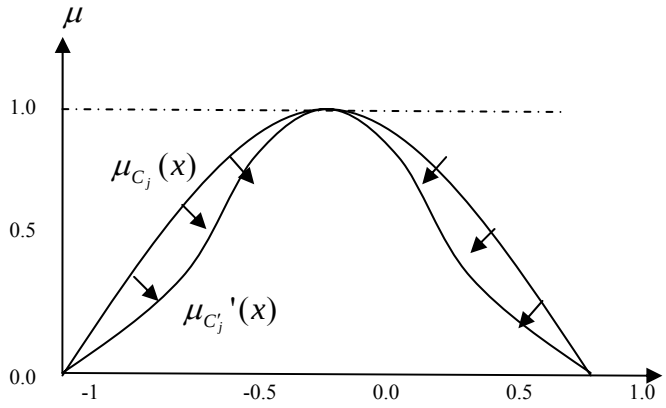


Figure 5.3: Membership value reduction form

5.5.4 Rules Propositions

The following shows what $C_j^{'}$ can be drawn for each of the three cases:

Case 1: The antecedent A_j has only one proposition $AG_w = S_{EC}(a_1^{'}, a_1) = S_{EC}(A', A)$ (Yeung *et al.*, 2002). Since $w_1 / \sum_{j=1}^{n} w_j = 1$ with n=1. If $S_{DS}(a_1^{'}, a_1) \geq \lambda_{a1}$ then $C^{'}{}_j = \min\{1, C_j /(AG_w * \mu)^{0.5}\}$ or $C^{'}{}_j = C_j * (AG_w * \mu)^{0.5}$, depending on whether we want to restrict or dilate the membership value of C_j.

Case2: The antecedent A_j has two or more proposition connected by "AND" (Yeung *et al*., 2002) $AG_w = \sum_{i=1}^{n} (S_{EC}(a_i^{'}, a_i) * \frac{w_i}{\sum_{j=1}^{n} w_j})$. If $S_{DS}(a_i^{'}, a_i) \geq \lambda_{ai}$ for all $(1 \leq i \leq n)$, then $C_j^{'} = \min\{1, C_j / (AG_w * \mu)^{0.5}\}$ or $C_j^{'} = C_j * (AG_w * \mu)^{0.5}$

Case3: The antecedent A has two or more propositions connected by "OR" (Yeung *et al*., 2002). This rule can be split into *n* simple rules as shown in Case1, i.e., $AG_w = S_{EC}(a_i^{'}, a_i)$. Because for each $i = 1,..., n, \sum_{j=1}^{n} w_j$ is reduced to a single term w_i, and becomes $w_i / w_i = 1$

If $\exists_i$ s.t. $S_{DS}(a_i^{'}, a_i) \geq \lambda_{ai}$ $\forall$ $(1 \leq i \leq n)$, then

$C_j^{'} = \min\{1, C_j / \max(AG_{w1}, AG_{w2}, ..., AG_{wi}) * \mu)^{0.5}\}$

or, $C_j^{'} = C_j * \max(AG_{w1}, AG_{w2}, ..., AG_{wi}) * \mu)^{0.5}$ OR

If $S_{DS}(a_i^{'}, a_i) \geq \lambda_{ai}$ $\forall$ $(1 \leq i \leq n)$, then

$C_j^{'} = \min\{1, C_j / \max(AG_{w1}, AG_{w2}, ..., AG_{wi}) * \mu)^{0.5}\}$ or,

$C_j^{'} = C_j * \max(AG_{w1}, AG_{w2}, ..., AG_{wi}) * \mu)^{0.5}$.

IF a_1 is fa_1 AND a_2 is fa_2 THEN C is fc, $(CF = \mu)$, $Th = \{\lambda_{a1}, \lambda_{a2}\}$, $W = \{w_1, w_2\}$, e.g., IF Open is low AND Close is low THEN Signal is low

5.5.5 Fuzzy Reasoning Algorithm

Let $C = \{\lambda_1, \lambda_2, ..., \lambda_n\}$ be an object to be classified, F be a group of WFPRs extracted from the tree, and there are k samples (Yeung *et al*., 2002). The initial state of C with respect to classification is set to be $(x_1, x_2, ..., x_k) = (0,0,...,0)$.

Step 1: From the group F, select a rule $R: IF\ A\ THEN\ B\ [CF, Th, Lw]$ where the antecedent A is supposed to be $(A_1, A_2, ..., A_n)$; the consequent B to be $(b_1, b_2, ..., b_n)$; and the local weight Lw to be $(Lw_1, Lw_2, ..., Lw_n)$.

Step 2: Compute the membership degree of λ_j belonging to $A_j: SM_j = A_j(\lambda_j)$ where the membership function of the fuzzy set A_j is denoted by itself.

Step 3: Let $Th = (Th_1, Th_2, ..., Th_n)$ be the threshold. If for each proposition A_j the inequality $SM_j \geq Th_j$ holds, then the rules are executed. Computed the overall weighted average SM_w of similarity measures as $SM_w = \sum_{j \in T} (Lw_j / \sum_{i \in T} Lw_i) SM_j$ where $T = \{j : SM_j \geq Th_j\}$.

Step 4: Modify the matching-consequent according to one of the beforehand given modification strategies:

4a) more or less form: $B^* = \min\{1, B / SM_w\}$;

4b) membership-value reduction form: $B^* = B_* SM_w$;

4c) keeping the consequent of the rule unchanged (no modification): $B^* = B$.

Step 5: Compute the certainty degree of B^* as $CF_{B^*} = CF * SM_w$;

Step 6: Put $x_j = \max(CF_{B^*}, x_j)$ where j is a number corresponding to the sample of the consequent B^*.

Repeat the above six steps until each rule within the group F has been applied to the object C.

5.6 Experiments and Results

The previous chapter and the preceding sections of this chapter have addressed predictive FDT and similarity-based fuzzy reasoning method. Any lengthy treatment of data mining would be incomplete without some description of the results analysis step. This section explores the analysis of results during a data mining research for stock market prediction. For experimentation perspective, this section can be split into two parts: in first part the database contained quarterly information on 100 companies during the period January 1, 1999 to December 31, 2004 from KLSE, NYSE and LSE is used to build predictive FDT. In different experiments the single day information is used for every 5 minutes of periods during 9:30 to 3:30. In addition, some important factors such as oil price, natural disaster, political situation, new policies, new budget, other companies, local stock markets, overall world situation are also been considered. In second part, the results of significant rules extracted from predictive FDT are discussed. Finally, use these rules for the prediction of stock market data.

5.6.1 Applying the Data Mining Process

The process that is followed in data mining stock market database parallels the description in Chapter 3. During the first pass through the data mining process, the steps taken were:

- **Define the Problem**. In this research, it has been studied that how well a data mining techniques would work on stock data. The user ultimate goal is to use data mining techniques classification to build model that would assist during stock market prediction.

- **Extract Data**. The database provided by the neuro dimension company was already in the proper format, so no typical extraction was needed.

- **Clean the Data.** As mentioned in Chapter 3, some time is spent to explore the stock market database and found that the "Volume" column had been mislabeled. After fixing the column names, the several values of the new "Volume" field are randomly compared with an independent database to ensure the values were correct. ESTEEM method is also use to clean data for predictive FDT algorithm. The ESTEEM method repeatedly runs a classification algorithm and removes those records it misclassifies from the training database, until all records in the reduced training set are correctly classified.

- **Data Engineering.** The stock market database that is used included five basic attributes containing information on trends of the basic attributes, e.g., the change in a price of open, close, etc. By analyzing the current stock trends the eight new attributes are also created.

- **Algorithm Engineering.** Algorithm engineering process consist of two steps: first predictive FDT is used for the classification of time series stock market index, and then similarity-based fuzzy reasoning method is used to mine the significant rules for the prediction of stock market.

- **Run the Data Mining Algorithm**. The algorithm discretizes all numeric attributes and then learns a set of rules whose antecedents are value assignments to some (usually small) number of input attributes, and whose consequents are value assignments to the class. When making a prediction, several rules may all match a given input record and their predictions are combined into a single score. For each quarter in the database, the preceding four quarters of data is concatenated, used them as training set, built a model, and applied that model to the target quarter. The result of this application step was a score for each record. Scores near 1 indicated that the stock was estimated to have a high probability of exhibiting exceptional return during the current quarter and scores near 0 indicated a low probability of

exceptional return. The last four quarters are held out to use as a final independent test later.

5.6.2 Experimental Design

In experiments, single day information (i.e. 78 share stock information from 9:30 to 3:55) is used to build predictive FDT. In the specific stock market prediction problem, we discuss in this section, the database contained quarterly information on 100 companies during the period January 1, 1999 through December 31, 2004 from KLSE, NYSE and LSE. To train and test predictive FDT quarterly stock data from Jan 1, 1999 to Dec 30, 1999 is used as: ten-folded cross-validation is used: first partition database into ten parts of equal size then repeatedly train on nine of the parts and test on the remaining one, running each algorithm a total of ten times. Now we can consider the results analysis step of the process. For every quarter including the last four (now that we have decided on the parameters of our solution) we have a corresponding model that we trained on the preceding four quarters (as shown in Figure 5.4). We apply each model to its target quarter of data, generating a prediction of the likelihood that each stock will exhibit exceptional performance during the quarter.

In such quarterly experiments, the different factors are considered according to different situations. For example, how much different degrees of factors (such as oil price, political situations, over all world situation, natural disaster, other local stock market, other companies, new budget, and new government policies) were affected in every quarter during (Jan 1, 1999 to Dec 30, 1999)? However, it is difficult to get such type of information from historical stock market data. In this research, the first attempt has been made and information is collected for the last one year during Jan 1, 2004 to Dec 30, 2004 from KLSE, NYSE and LSE. Similarly, for the remaining four years, the assumed datasets are used to fill this gap. For the removal of noisy data, the Esteem algorithm is used in the following way, instead of rejecting the records that were misclassified during one pass of the algorithm. Each run was repeated ten times

(throwing out a different random set of records) and the results were averaged. This experiment was important because, after the first set of experiments, there remained the possibility that the good results were not due to the success of the Esteem method but rather that the moderately reduced training set size resulted in slightly smaller models that were about as accurate.

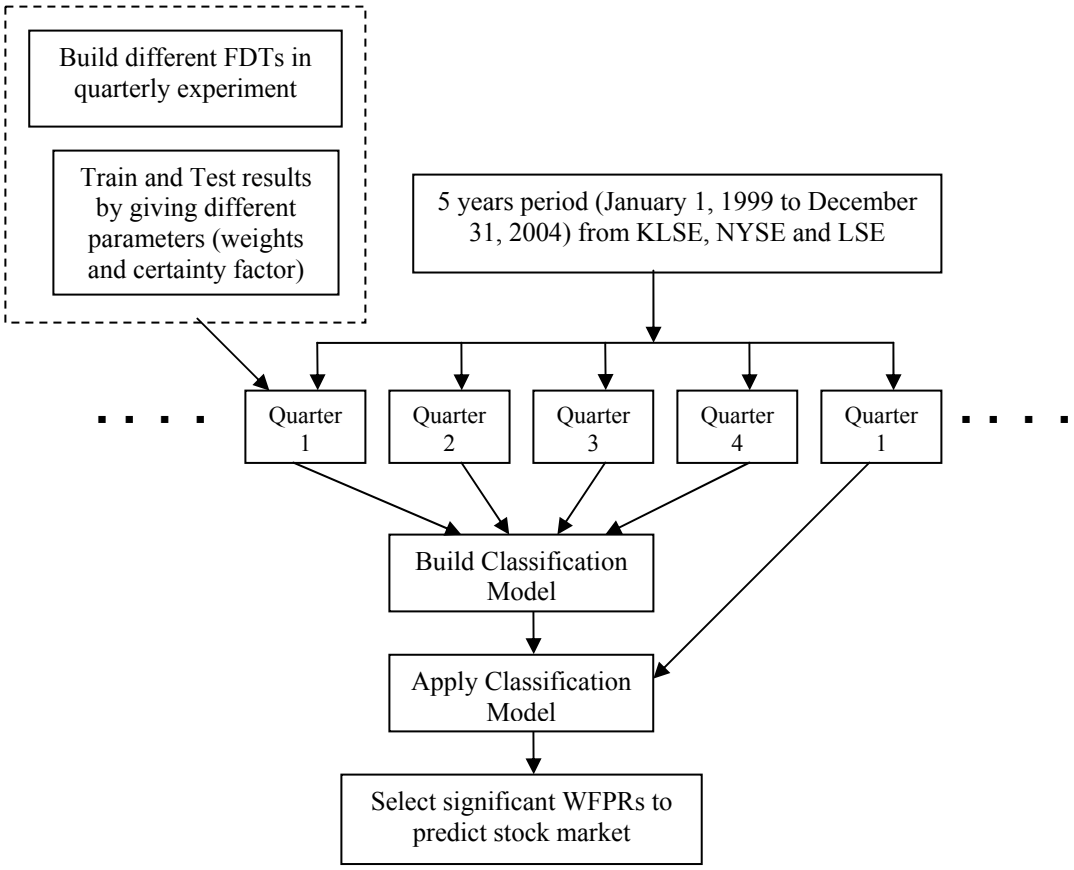


Figure 5.4: Experimental design for stock market prediction by using 5 years information (Jan 1, 1999 to Dec 30, 2004) from KLSE, NYSE and LSE.

The operational steps of the developed financial forecaster based on predictive FDT, named as KnowledgeMiner, are shown in Figure 5.5 to Figure 5.11.

At start up of the “KnowledgeMiner” system, a welcome window, as shown in Figure 5.5, appears on the screen. Here, the user can select the relevant option:

1. Learning the Basics; to know about basics of the working principle of the “KnowledgeMiner”
2. Using the Interface; help on how to use the interface
3. General Information; overview about how to use the system (Figure 5.6)
4. Go to Main Window; for using the operational features of the system

By clicking the “Go to Main Window” button in the welcome window, a user can go further for operating the system. Here the main window of the system appears where the user can follow certain steps for stock market prediction, as given below:

1. Load/Download Data: Click the “Data” button, in the main window shown in Figure 5.7, to load/download historical data up to for the last five years, of a selected stock market and company.
2. Select Training Data: Click the “Train” button to select training data, as shown in Figure 5.7. Results of training appear as shown in Figure 5.8 consisting of corresponding rules extracted from the fuzzy decision tree.
3. Walk Forward Test: Click the “Test” button to run walk forward test for end-of-day or intra-day trading.
4. Select Splitting Function: Please click the “SpFun” button to select degree of importance of various factors like political situation, oil-price, etc.
5. Stock Market Prediction: Click on “Prediction” button for the prediction of next days/weeks/months. Here user can select short term or long term trading as shown in Figure 5.9. The prediction results appear as shown in Figures 5.10 and 5.11.

Figure 5.5: Welcome window of the KnowledgeMiner: user can select the relevant option

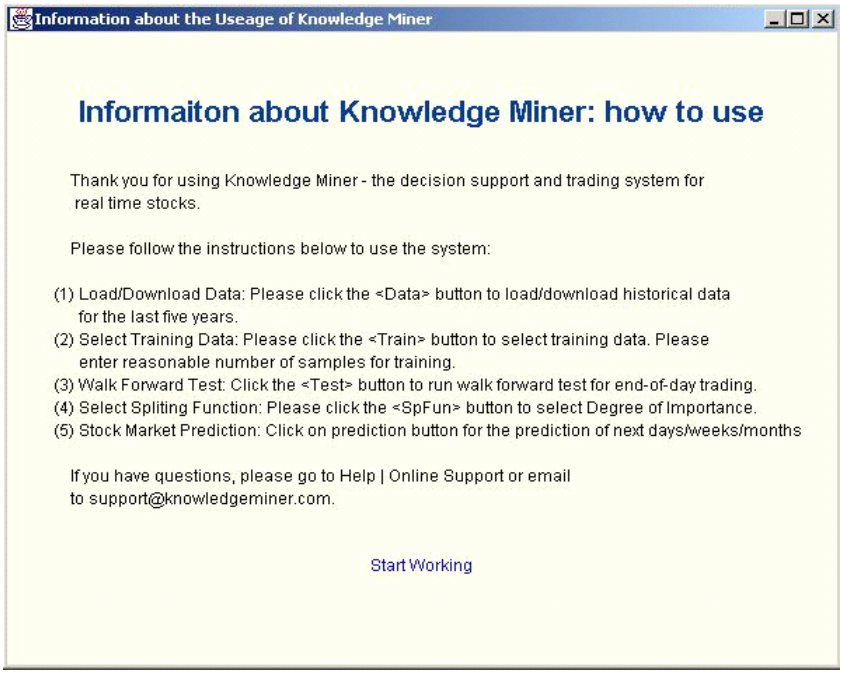


Figure 5.6: KnowledgeMiner consist of five basic steps as given in the figure

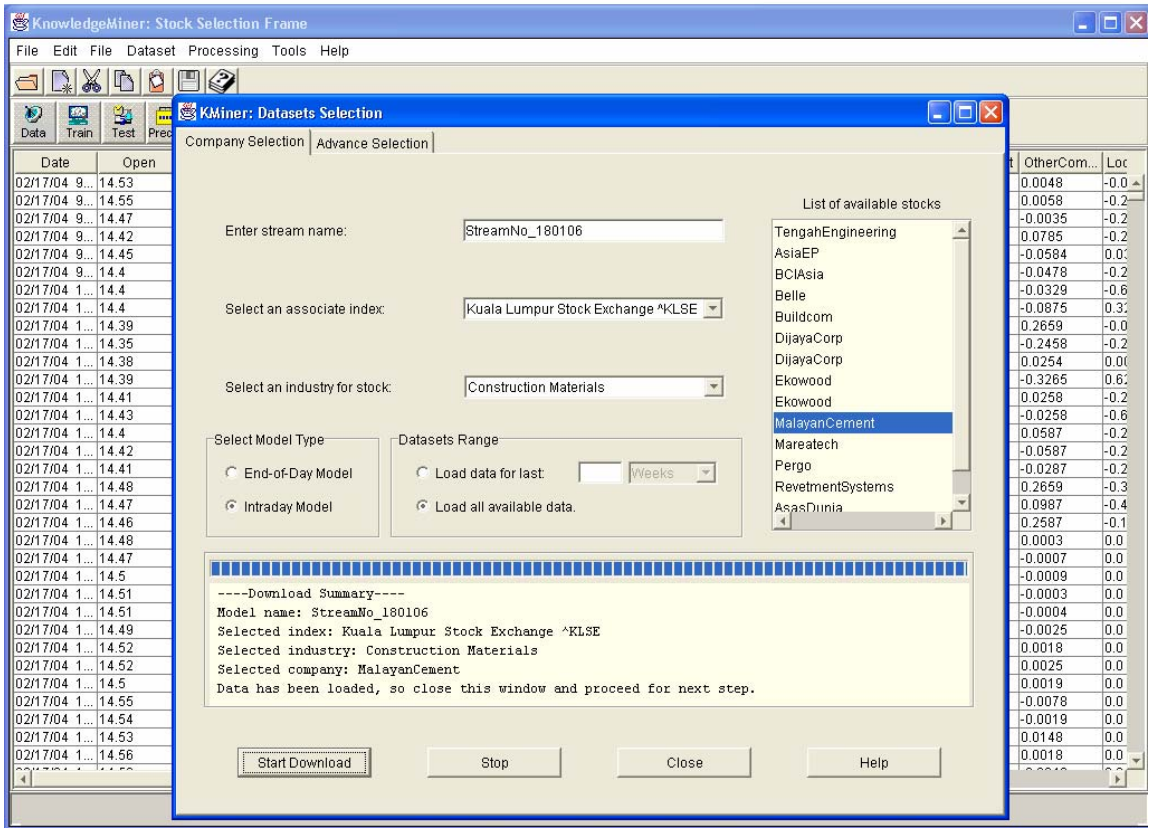


Figure 5.7: Firstly the user selects the type of stocks that he wants to predict.

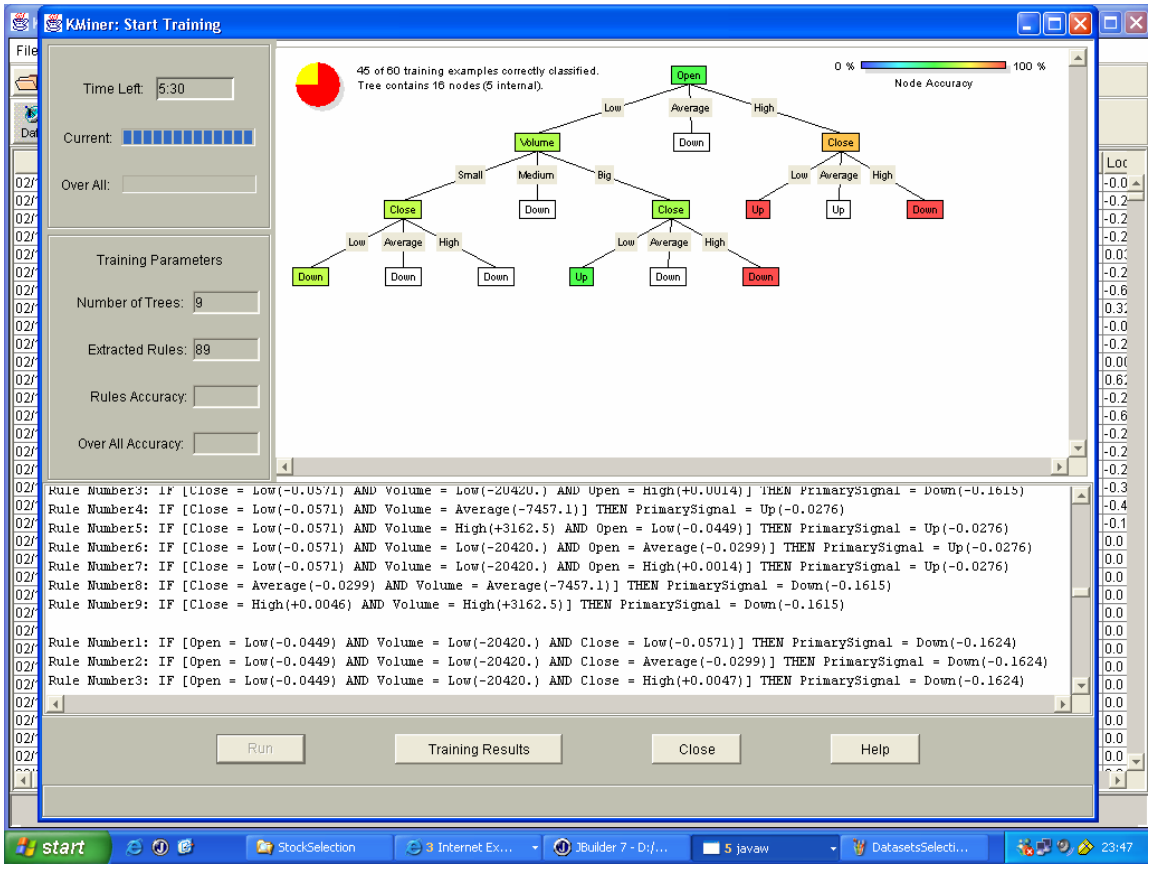


Figure 5.8: This window shows the tree view panel and extracted rules from predictive FDT.

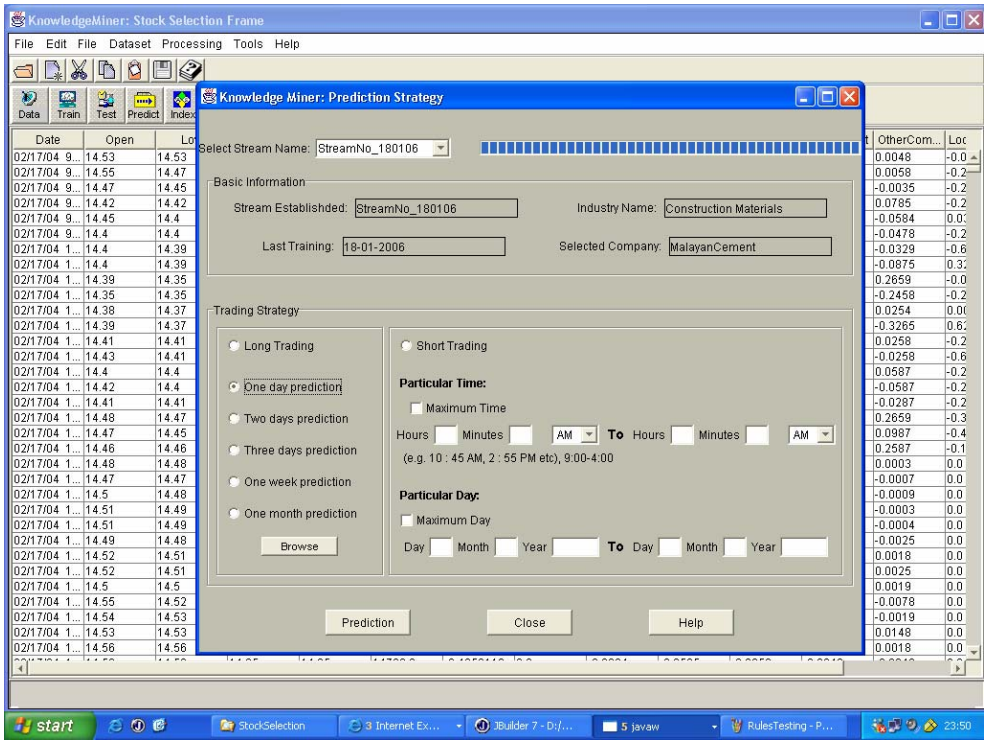


Figure 5.9: User can select either long term or short term trading strategy.

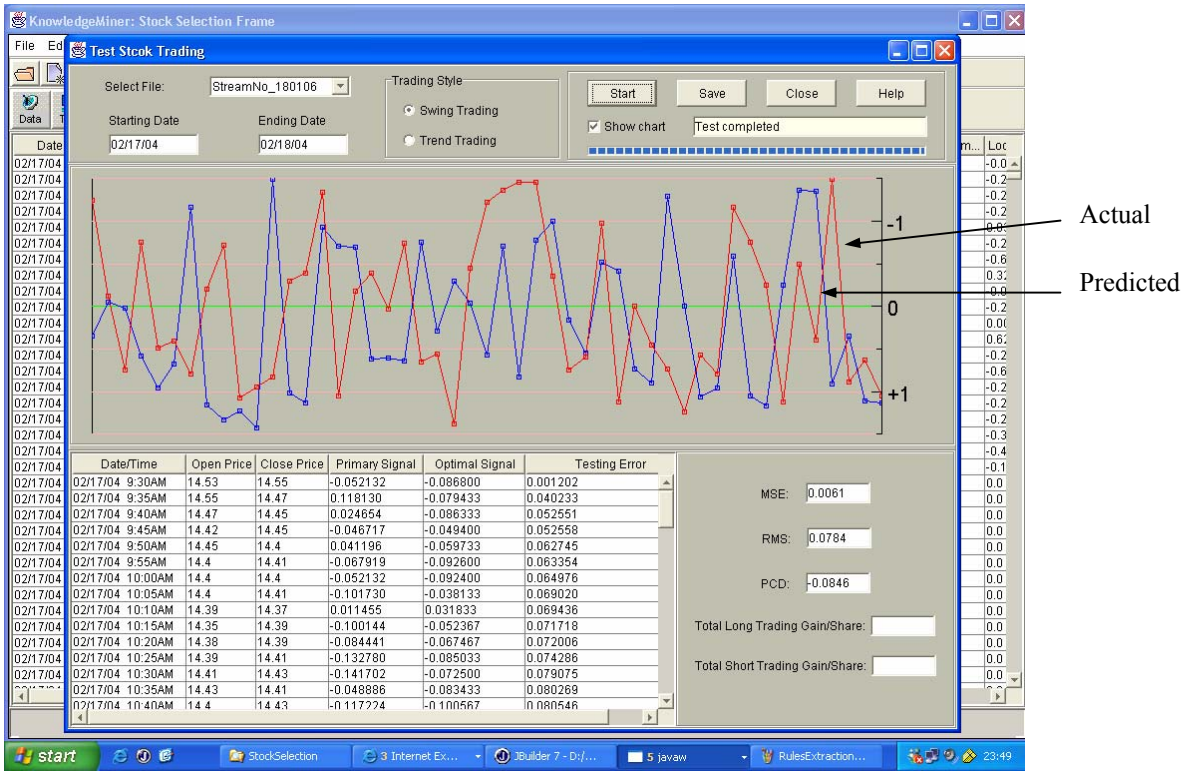


Figure 5.10: This window shows the testing results after extracting the WFPRs

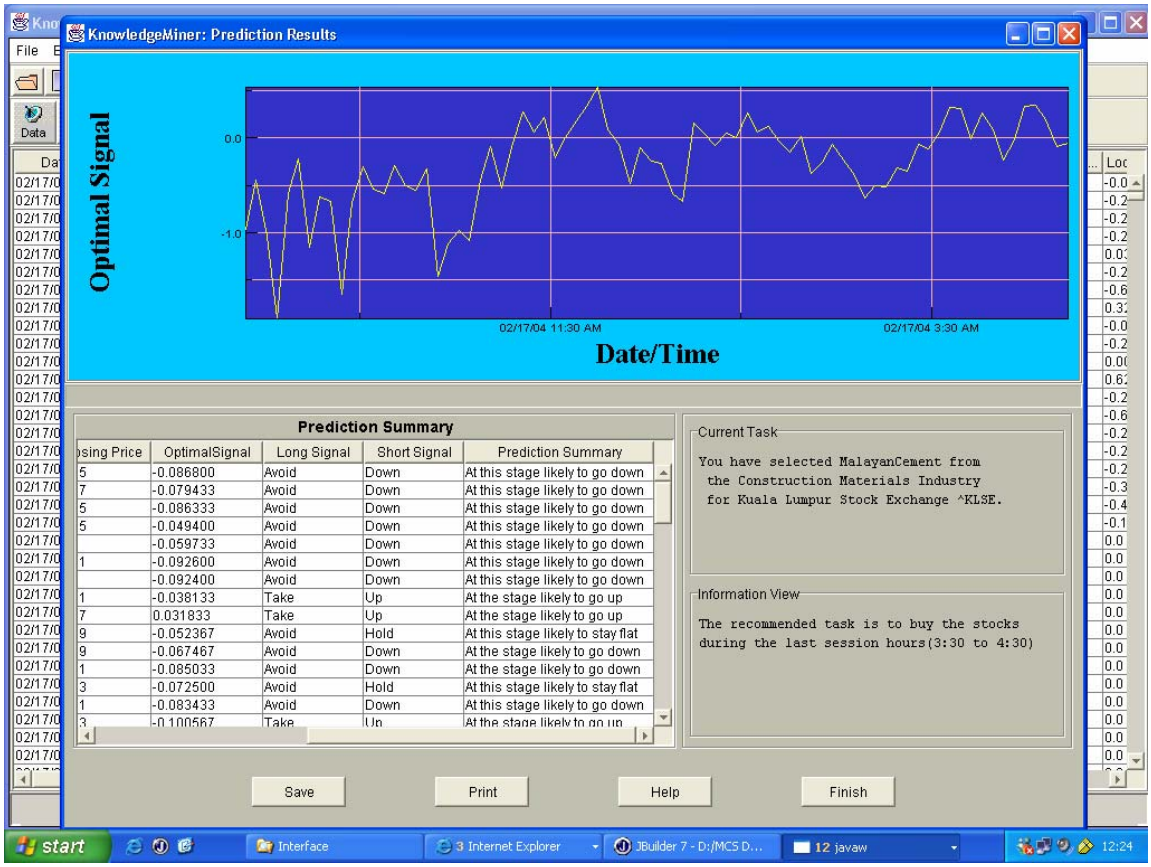


Figure 5.11: This window presents the prediction results

5.6.3 Testing Results and Analysis of Predictive FDT

For the construction of predictive FDT, the number of linguistic terms for each attribute is taken to be 2 or 3, the parameter α reducing the fuzziness in training process is set to 0.35 and the leaf criterion is taken to be 0.75. The learning accuracy, computational complexity, and comprehensibility are used to compare the performance of the two methods with predictive FDT as shown in Figure 5.12 (a)-(c). The learning accuracy includes training and testing accuracy, the complexity is regarded as the numbers of nodes and leaves. For each considered databases, 90% of cases are uniformly and randomly chosen as the training set and the remaining 10% of cases are held for testing. This procedure is repeated ten times for the given cut standard = 0.35 and the leaf standard = 0.75. The number of nodes, the number of leaves, the training accuracy, and testing accuracy are regarded as the average of the five. These targets are considered due to the following reasons.

(i) The number of nodes represents the complexity degree of the generated tree (the complexity of extracted fuzzy rules), which is closely related to the time-complexity and space-complexity. The number of leaves corresponds to the number of extracted fuzzy rules. It is reasonable to argue that a simple tree is considered to be superior to a complex one.

(ii) The training accuracy and the testing accuracy are the two most important factors for fuzzy decision trees. The training accuracy refers to the correctness rate of testing the training set by the extracted rules. Usually the training accuracy of the crisp learning with out noise can attain 100% but the fuzzy learning cannot. The testing accuracy represents the capability of predicting classes of novel examples. The experimental results are summarized in Table 5.1 and the detail can be found in appendix G, H, and I.

Table 5.1: Summary of the experiments for I = FuzzyID3, II = Wang and III = Predictive FDT

Database	Number of Nodes			Number of leaves			Training accuracy			Testing accuracy		
	I	II	III	I	II	III	I	II	III	I	II	III
KLSE	34.55	42.52	47.59	24.26	36.29	42.22	0.86	0.84	0.65	0.84	0.82	0.60
NYSE	32.24	40.19	45.25	30.47	36.49	41.26	0.94	0.96	0.55	0.89	0.90	0.50
LSE	35.44	42.18	49.34	25.88	35.34	40.55	0.90	0.85	0.67	0.88	0.80	0.60

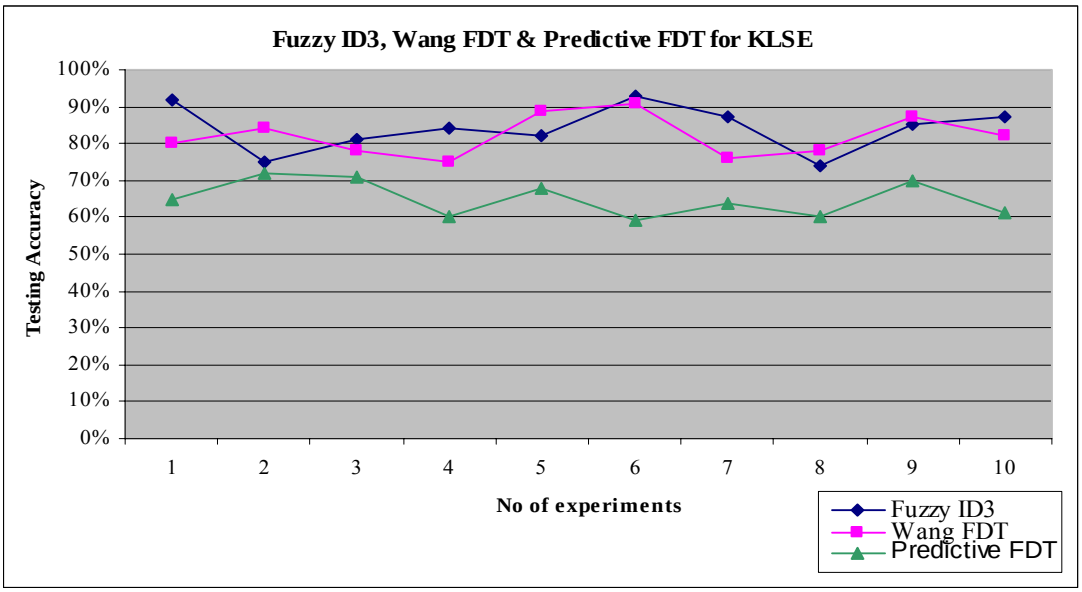


Figure 5.12 (a): Experimental results of FuzzyID3, Wang FDT, and Predictive FDT for KLSE

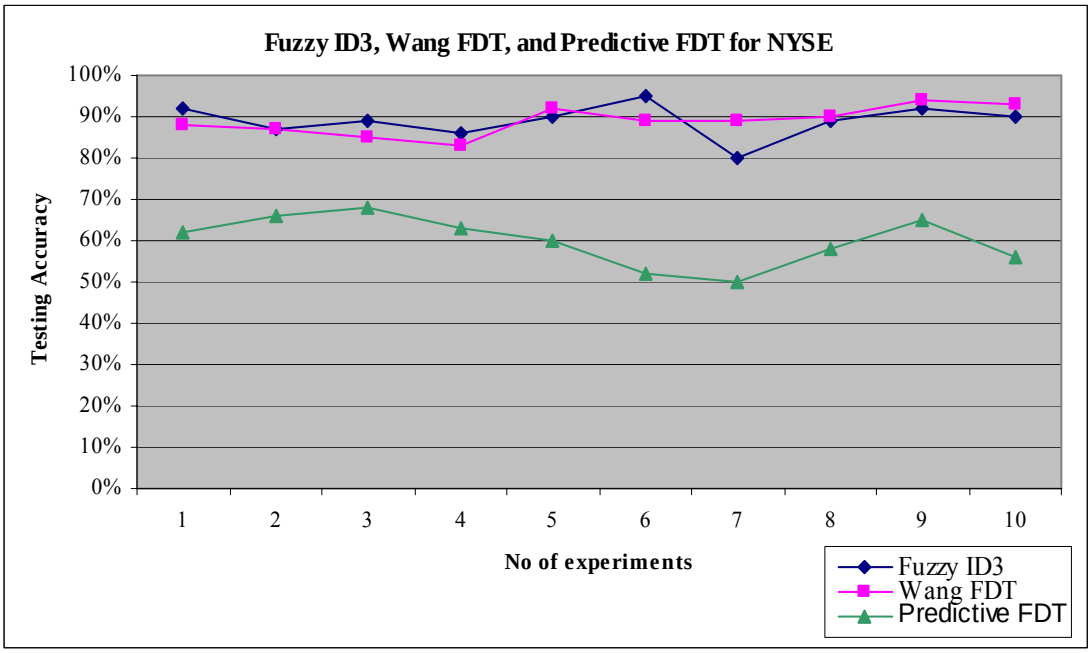


Figure 5.12 (b): Experimental results of FuzzyID3, Wang FDT, and Predictive FDT for NYSE

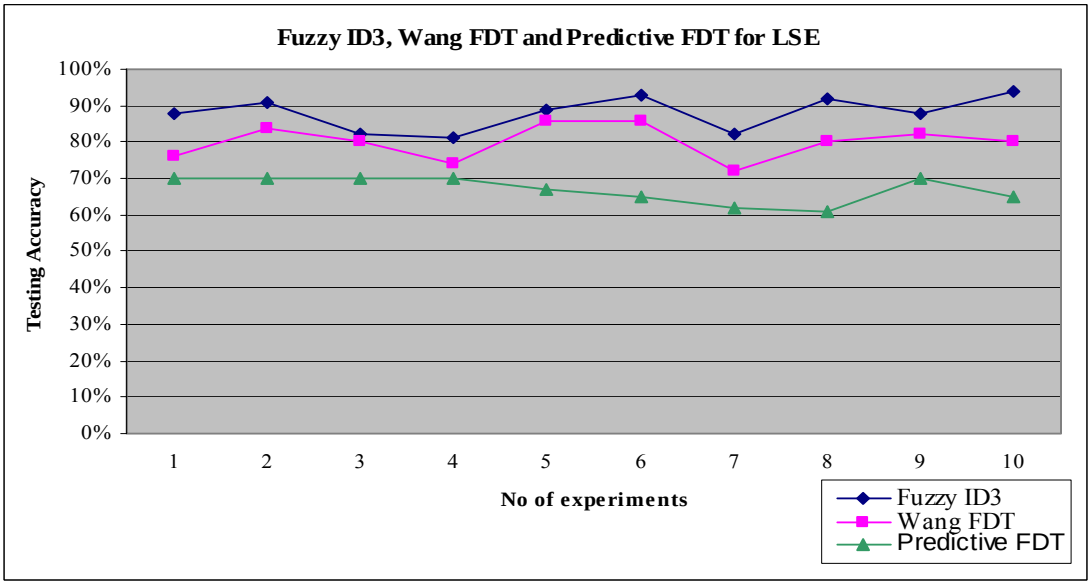


Figure 5.12 (c): Experimental results of FuzzyID3, Wang FDT, and Predictive FDT for LSE

5.6.4 Rules Extraction and Experiments

Predictive FDT algorithm searches through the space of fuzzy production rules to find those likely to be useful for classifying items in a future stock market behavior. For example, one rule extracted from predictive FDT in Figure 4.7 (chapter 4) is: IF [P_{LSM} = $LowAffect(4.25)$ AND C_V = $Med(3.52)$ AND P_{OWS} = $Good(2.50)$ THEN Primary Signal = $Hold(0.85)$]. Predict unexceptional (Hold) return with some strength. The antecedent of the rule represents the attributes on the passing branches from the root to the leaf and the consequent of the rule represents the labeled at the leaf node. Table 5.2 shows the result of some WFPRs by using the conversion method for the extraction of rules from single experiment of predictive FDT.

Similarly, the WFPRs can be extracted from predictive FDT in each experiment. We may have many rules in every experiment. Quarterly data is used so all trading occurred at the beginning of each quarter, and we held the resulting asset mix for the duration of the quarter. At the beginning of each quarter, we used the predictive FDT and fuzzy reasoning method to build a set of rules that accurately predicted exceptional return for stocks during the preceding four quarters. We then used the learned model to predict the stock. For example, the prediction held from Jan 1, 2005 through March 31, 2005, was selected in the following way: we trained the predictive FDT on the four preceding quarters beginning January 1, 2004; April 1, 2004; July 1, 2004 and October 1, 2004 (Figure 5.13). During training, many rules are found that were good predictors of exceptional future return within the training sample. Then these rules are applied to the stock data available on January 1, 2005 and mine the rules that have better predictive accuracy for the coming quarter.

Table 5.2: An example of 13 extracted WFPRs from FDT in single experiment.

Rule No	Weighted fuzzy production rules
1	IF [Other Local Stock Market=LowAffect AND Change in Volume=Small] THEN [Primary Signal=Down], (CF_1=0.95), w_{11}=4.25, w_{12}=3.75)
2	IF [Other Local Stock Market=LowAffect AND Change in Volume=Med AND World Situation=Bad] THEN [Primary Signal=Hold], (CF_2=0.92, w_{21}=4.25, w_{22}=3.52, w_{23}=1.35)
3	IF [Other Local Stock Market=LowAffect AND Change in Volume=Med AND World Situation=Good] THEN [Primary Signal=Hold], (CF_3=0.85, w_{31}=4.25, w_{32}=3.52, w_{33}=2.50)
4	IF [Other Local Stock Market=LowAffect AND Change in Volume=Large AND Political Situation=Clear] THEN [Primary Signal=Down], (CF_4=0.60, w_{41}=4.25, w_{42}=2.53, w_{43}=1.25)
5	IF [Other Local Stock Market=LowAffect AND Change in Volume=Large AND Political Situation=NotClear] THEN [Primary Signal=Up], (CF_5=0.95, w_{51}=4.25, w_{52}=2.53, w_{53}=1.07)
6	IF [Other Local Stock Market=MedAffect AND Other Companies=Low AND Natural Disaster=No] THEN [Primary Signal = Down], (CF_6=0.93, w_{61}=5.26, w_{62}=4.34, w_{63}=3.35)
7	IF [Other Local Stock Market=MedAffect AND Other Companies=Low AND Natural Disaster=Yes] THEN [Primary Signal = Down], (CF_7=0.88, w_{71}=7.26, w_{72}=1.25, w_{73}=2.57)
8	IF [Other Local Stock Market=MedAffect AND Other Companies=Med] THEN [Primary Signal=Hold], (CF_8=0.90, w_{81}=5.26, w_{82}=4.53)
9	IF [Other Local Stock Market=MedAffect AND Other Companies=High] THEN [Primary Signal=Up], (CF_9=0.68, w_{91}=5.26, w_{92}=3.25)
10	IF [Other Local Stock Market=HighAffect AND Change in Close=Low] THEN [Primary Signal = Down], (CF_{10}=0.97, w_{101}=5.45, w_{102}=4.45)
11	IF [Other Local Stock Market=HighAffect AND Change in Close = Med AND World Situation=Bad] THEN [Primary Signal=Up], (CF_{11}=0.88, w_{111}=5.45, w_{112}=4.05, w_{113}=3.24)
12	IF [Other Local Stock Market=HighAffect AND Change in Close = Med AND World Situation=Good] THEN [Primary Signal=Up], (CF_{12}=0.92, w_{121}=5.45, w_{122}=4.05, w_{123}=2.57)
13	IF [Other Local Stock Market=HighAffect AND Change in Close=High] THEN [Primary Signal=Up], (CF_{13}=0.67, w_{131}=5.45, w_{132}=5.25)

Figure 5.13 presents the prediction process for Jan 1, 2005 to March 30, 2005 using the historical stock market data from Jan 1, 1999 to Dec 30, 2004. Figure 5.14 (a) shows the results of core rules points after mining the best rules for the next quarter prediction of stock market. In Figure 5.14 (b), every core point presents the best learning rules and these rules compare with the actual stock price. These core points are used as optimal signals of stock prices for different stocks. Optimal signals are actually that signals that calculate for final stock market prediction on the bases of above predicted rules.

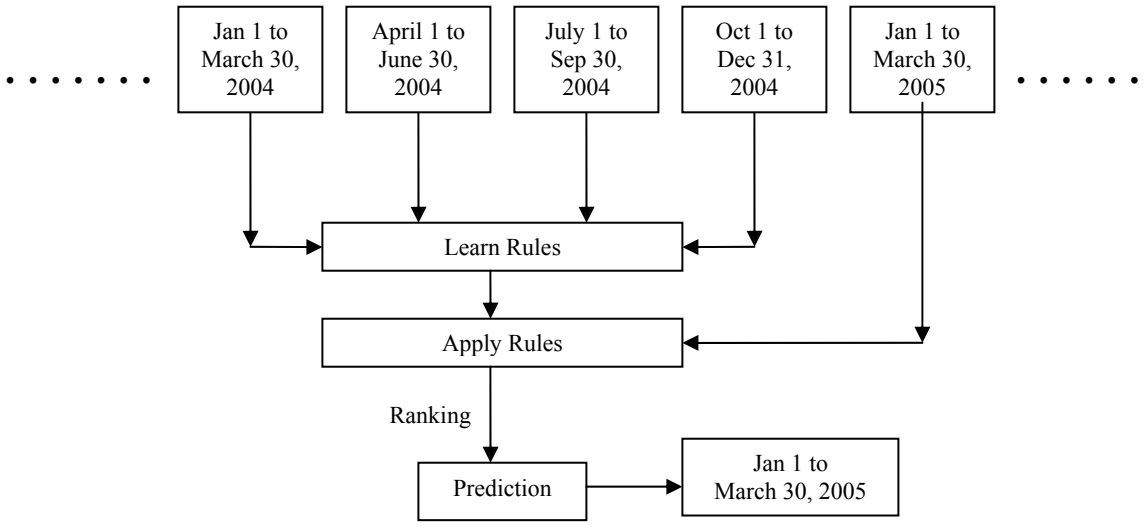


Figure 5.13: The prediction process for Jan 1 to March 30, 2005 using the historical stock market data from (Jan 1, 1999 to Dec 30, 2004).

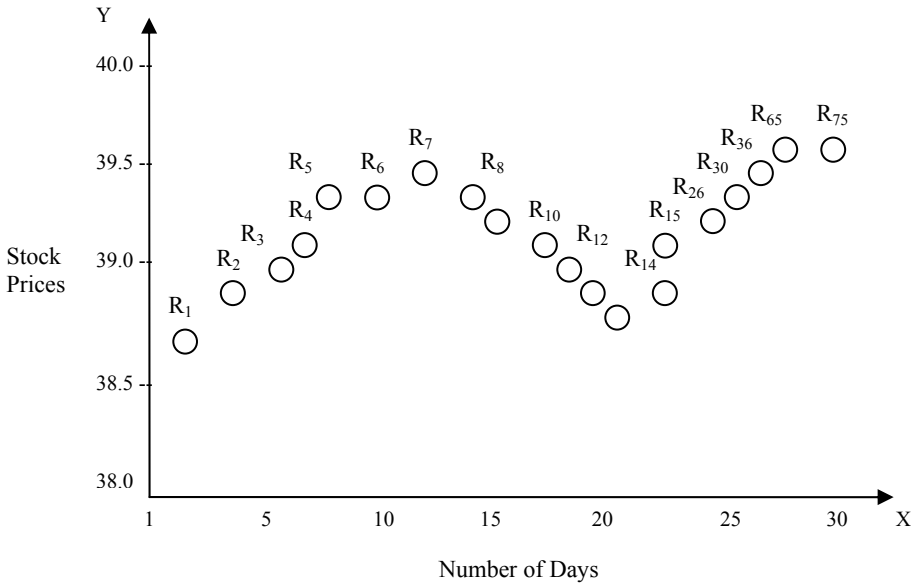


Figure 5.14 (a): Core Points of the rules for three months prediction

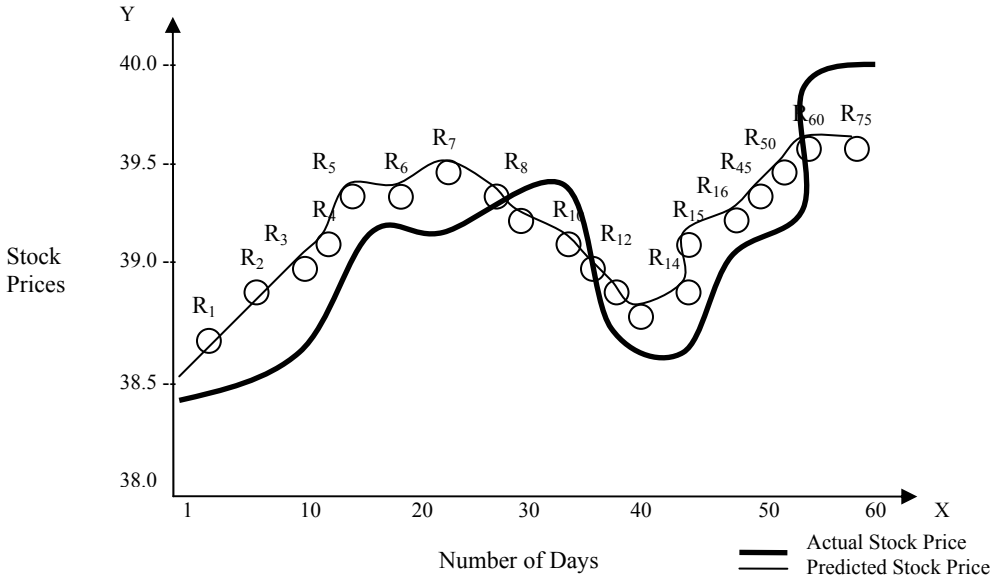


Figure 5.14 (b): Results using core points of rules for three months predictions from (Jan 1, 1999 to Dec 30, 2004)

There are a variety of ways to analyze the results at this point, and we use Mean Square Error (MSE), Root Mean Square Error (RMSE) and Prediction of Change in Direction method (PCD method from Ayob *et al*. (2001)) to test the results (as shown in Table 5.3).

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(P_i - E_i)^2 \tag{5.23}$$

where P_i is the predicted output and E_i is the expected output, and N is the total number of patterns in experiment.

$$\text{PCD} = n \times \frac{\sum_{i=1}^{N} D_i}{N} \tag{5.24}$$

where $$D_i = \begin{cases} 1 & \textit{if} \quad (E_i - E_{i-1}) \times (P_i - P_{i-1}) < 0 \\ 0 & \textit{otherwise} \end{cases} \tag{5.25}$$

Table 5.3: Testing results for significant rules using prediction in change method

Training Sets	MSE	RMS	PCD
1 year	0.8578	0.9262	48.75
2 years	0.8135	0.9019	55.85
3 years	0.6587	0.8116	56.75
4 years	0.6258	0.7911	56.85
5 years	0.5987	0.7738	58.63

5.6.5 Analyzing Prediction Results

From Table 5.4, we can examine that the predictive FDT method has the reasonable mean square error comparing with the other random walk models i.e., Autoregression Moving Average (ARMA), and Autoregression Integrated Moving Average (ARIMA). This implies that the predictive FDT method can obtain the satisfactory results for the short term prediction. As shown in Figure 5.15 (a)-(c), the predicted sequence indicates the predicted results of different random models like ARMA, ARIMA, and predictive FDT algorithm. In these experiments, the most recent 3 months values is considered as a set of input data used for modeling to predict the next actual stock prices. The stock price index predictions for three domains (KLSE, NYSE, and LSE) are experimented as shown in Figure 5.15 (a)-(c). The accuracy of prediction methods that are the ARMA, ARIMA, and predictive FDT can also be compared and the summary of this experiment is listed in Table 5.4. From Figure 5.15 (a)-(c), we can see that predictive FDT algorithm provides the reasonable trading strategy than other random walk techniques like ARMA, and ARIMA for three months prediction of stock market.

Table 5.4: The mean squared error between the actual values and the predicted results for 3-case study stock price indexes is up to 3 months

Methods	KLSE Telekom	NYSE DowChemical	LSE ABN-Amro	Average of MSE
ARMA	11.85	4.25	6.05	7.383
ARIMA	12.75	3.65	5.36	7.253
Predictive FDT	9.25	3.45	4.17	5.6285

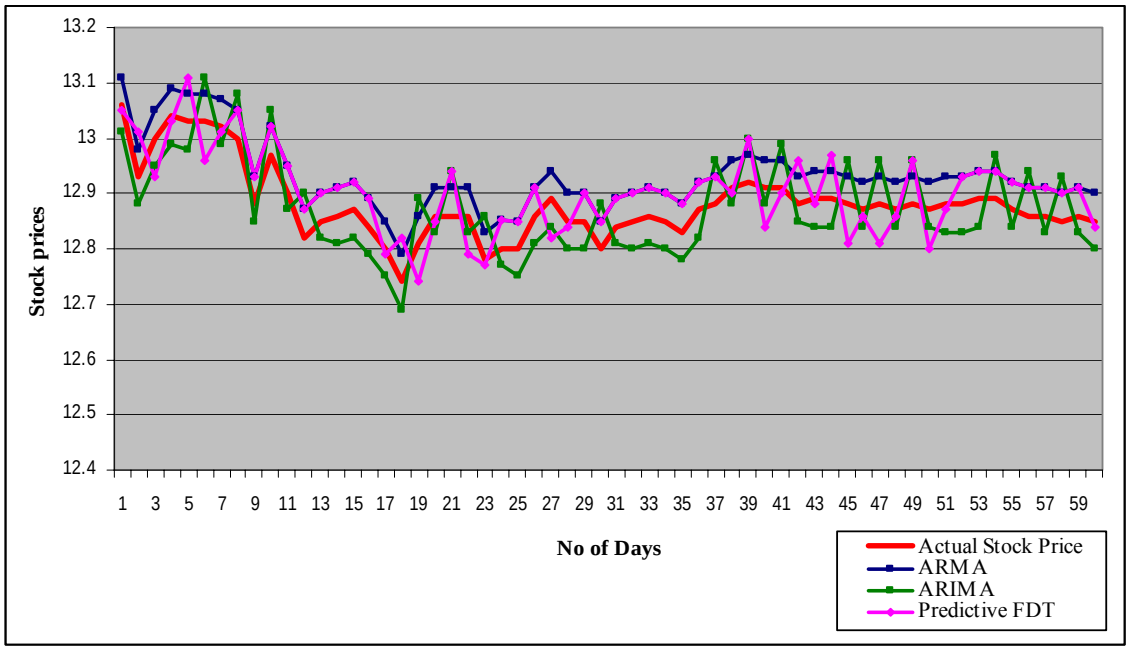


Figure 5.15 (a): Three Months (Jan 1 to March 31, 2005) prediction of Telekom Index from KLSE using random walk and predictive FDT

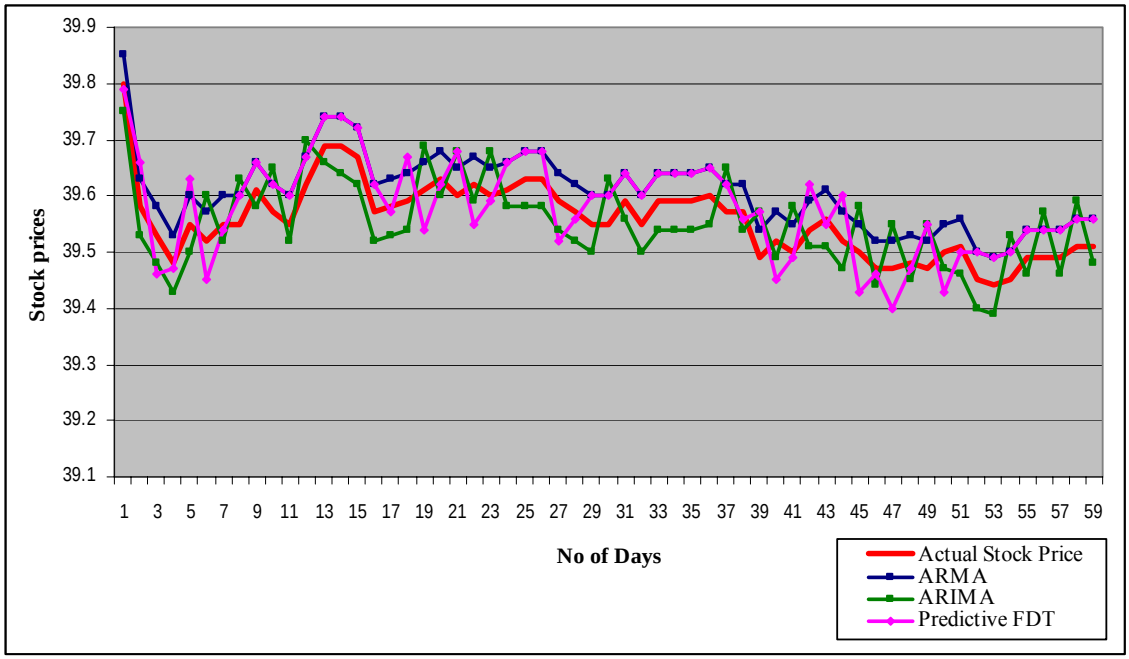


Figure 5.15 (b): Three Months (Jan 1 to March 31, 2005) prediction of DowChemical Index from NYSE using random walk and predictive FDT

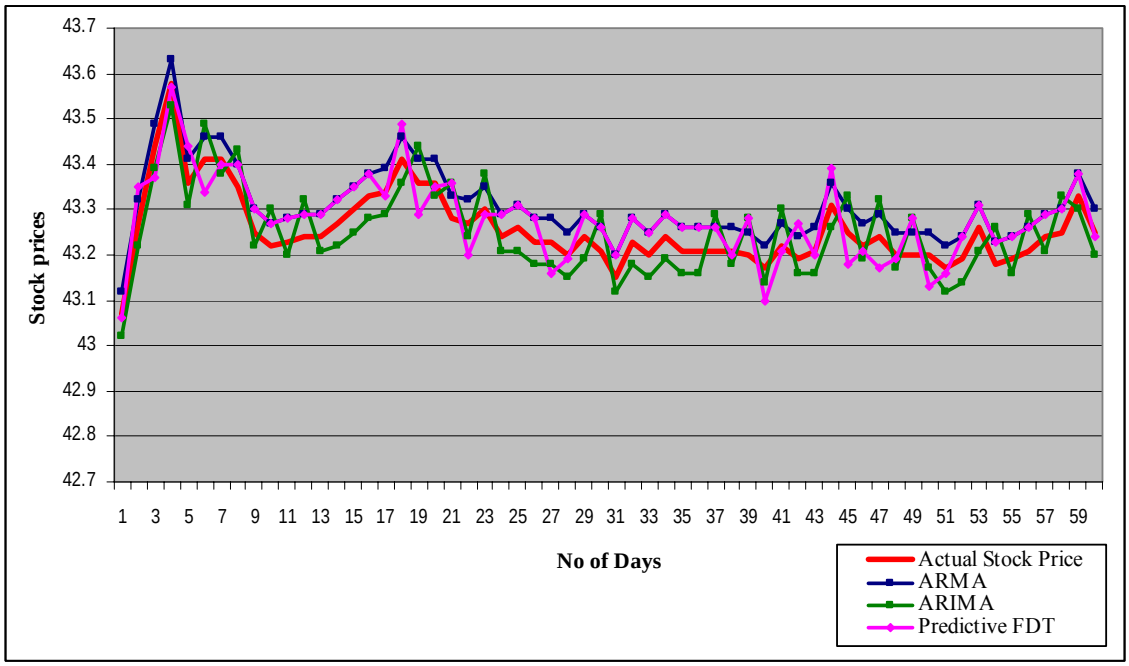


Figure 5.15 (c): Three Months (Jan 1 to March 31, 2005) prediction of ABN-Amro Index from LSE using random walk and predictive FDT

5.7 Summary

Fuzzy production rules (FPRs) are widely used in expert systems to represent fuzzy and uncertainty concepts. FPRs are usually presented in the form of a fuzzy IF-THEN rule in which both the antecedent and the consequent are fuzzy concepts denoted by fuzzy sets. With the increasing complexity of knowledge, it is found that only using the fuzzy sets to represent both the fuzziness and the uncertainty in FPRs is not enough. Therefore, some knowledge parameters such as certainty factor, local weight, threshold value and global weight are incorporated into the FPRs in this chapter. By using these parameters, first the WFPRs are extracted from predictive FDT and then similarity-based fuzzy reasoning method mine the more significant rules for the prediction of stock market data.

In this chapter, all training and testing results are presented that produced by predictive FDT and similarity-based fuzzy reasoning method. In this research, the more focus on the time series stock market prediction therefore in experiment, 5 minute period of stock market data has been used to analyze the whole database by splitting into quarterly (3 months) and predict the time series stock market data. The prediction for more than one quarter may give us inaccurate results because stock market is very sensitive and affected by many factors for long period of time.

CHAPTER 6

STOCK MARKET PREDICTION USING SUPPORT VECTOR REGRESSION

6.1 Introduction

Financial time series forecasting is one of the most challenging applications of modern time series forecasting. Financial time series are inherently noisy, nonstationary and deterministically chaotic (Deboeck, 1994; Yaser, 1996; Huang *et al*., 2005; Cao & Tay, 2003; Kim, 2003). These characteristics suggest that there is no complete information that could be obtained from the past behaviour of financial markets to fully capture the dependency between the future price and that of the past. In other words, the nonstationary characteristic implies that the distribution of financial time series changes over time. In the modeling of financial time series, this will lead to gradual changes in the dependency between the input and output variables. Therefore, the learning algorithm used should take into account this characteristic. Usually, the information provided by the recent data points is given more weight than that provided by the distant data points (Cao & Tay, 2003; Freitas *et al*., 1999; Refenes *et al*., 1997), as in nonstationary financial time series the recent data points could provide more important information than the distant data points.

There are two main categories in financial time series forecasting: univariate analysis and multivariate analysis. In multivariate analysis, any indicator, whether it is related to the output directly or not, can be incorporated as the input variable, while in

univariate analysis, the input variables are restricted to the time series being forecasted. A general univariate model that is commonly used is based on the AutoRegressive Integrated Moving Average (ARIMA) method. Compared to other multivariate models, the performance of ARIMA is not satisfactory because this model is parametric, and additionally, it is developed on the assumption that the time series being forecasted are linear and stationary. These constraints are not consistent with the characteristics of financial time series. Therefore, Artificial Neural Network (ANN) assisted multivariate analysis has become a dominant and popular tool in recent years. The prediction performance is greatly improved by the use of a neural network both in terms of prediction metrics and trading metrics (Abecasis and Lapenta, 1996; Cheng, 1996; Sharda, 1993; Van, 1997; Trafalis and Ince, 2002). Multivariate models can rely on greater information, where not only the lagged time series being forecasted, but also technical indicators, fundamental indicators or inter-market indicators are combined to act as predicators. Moreover, a neural network is more effective in describing the dynamics of non-stationary time series due to its unique non-parametric, non-assumable, noise-tolerant and adaptive properties. Neural networks are universal function approximators that can map any nonlinear function without *a priori* assumptions about the data.

However, a critical issue concerning neural networks is the over-fitting problem. It can be attributed to the fact that a neural network captures not only useful information contained in the given data, but also unwanted noise. This usually leads to a poor level of generalization. The performance of neural networks in terms of generalization for the out-of-sample data–the data that are not used in training the network–is always inferior to that of the training data. Therefore, the development of neural networks and the tasks related to architecture selection, learning parameter estimation and training, require substantial care in order to achieve the desired level of generalization. The significance of good generalization is critical when using neural networks for financial time series forecasting.

The issue of generalization has long been a concern to researchers, who have explored a variety of procedures for enhancing the generalization ability of neural networks. A typical approach uses cross-validation, where the data given is divided into three sub-samples. The first sample is for training, the second one for testing, while the third one for validating. This approach involves a substantial amount of computation that is often referred to as a weakness in theory. Nonparametric probability density estimation is another statistical tool used to improve generalization (Pan, 1998), but its underlying assumption–that the distribution for both patterns and noise remains the same over the model estimation and forecasting period–could often not be satisfied in financial time series. Other techniques of enhancing the generalization ability of neural networks include modifying training algorithms (Kimoto, 1993), pruning the connections or hidden nodes of the network (Dorsey, 1998), using adaptive learning parameters, and selecting significant variables (Abecasis and Lapenta, 1997; Aussem, 1998).

Recently, SVM developed by Vapnik (Vapnik, 1995) have provided another novel approach to improve the generalization property of neural networks. Originally, SVM was developed for pattern recognition problems. Recently, with the introduction of ε-insensitive loss function, SVM has been extended to solve non-linear regression problems. Unlike most of the traditional learning machines that adopt the Empirical Risk Minimization Principle, SVM implements the Structural Risk Minimization Principle, which seeks to minimize an upper bound of the generalization error rather than minimize the training error. This will result in better generalization than that with conventional techniques.

The objective of this chapter is to examine the functional characteristics of SVM in financial forecasting. Here we develop a stock market prediction system using SVM. As day to day stock prices are related, to account for this auto-correlation (i.e., measured values that are close in time and space tend to be highly correlated), weighted kernel-based clustering method incorporating neighborhood constraints is used. Because, in the financial time series, the recent data points could provide more important information

than the distant data points, in the weighted clustering algorithm, the weights are assigned accordingly.

6.2 SVM for Regression Estimation

Compared to other neural network regressors, SVM has three distinct characteristics when it is used to estimate the regression function. First, SVM estimates the regression using a set of linear functions that are defined in a high-dimensional feature space. Second, SVM carries out the regression estimation by risk minimization, where the risk is measured using Vapnik's ε-insensitive loss function. Third, SVM implements the SRM principle which minimizes the risk function consisting of the empirical error and a regularized term.

Given a set of data points $\{(x_1, y_1),....,(x_\ell, y_\ell)\}$, where each $x_i \subset R^n$ denotes the input space of the sample and has a corresponding target value $y_i \subset R$ for i=1,..., l where l corresponds to the size of the training data (Vapnik, 1998; Muller *et al*., 1998). The idea of the regression problem is to determine a function that can approximate future values accurately.

The generic SVR estimating function takes the form:

$$f(x) = (w \cdot \Phi(x)) + b \tag{6.1}$$

where $w \subset R^n$, $b \subset R$ and Φ denotes a non-linear transformation from R^n to high dimensional space. Our goal is to find the value of w and b such that values of x can be determined by minimizing the regression risk:

$$R_{reg}(f) = C\sum_{i=0}^{\ell} \Gamma(f(xi) - yi) + \frac{1}{2}\|w\|^2 \tag{6.2}$$

where $\Gamma(\cdot)$ is a cost function, C is a constant and vector w can be written in terms of data points as:

$$w = \sum_{i=1}^{\ell} (\alpha_i - \alpha_i^*)\Phi(x_i) \tag{6.3}$$

By substituting equation (6.3) into equation (6.1), the generic equation can be rewritten as:

$$f(x) = \sum_{i=1}^{\ell} (\alpha_i - \alpha_i^*)(\Phi(x_i) \cdot \Phi(x)) + b = \sum_{i=1}^{\ell} (\alpha_i - \alpha_i^*)k(x_i, x) + b \tag{6.4}$$

In equation (6.4) the dot product can be replaced with function $k(x_i, x)$, known as the kernel function. Kernel functions enable dot product to be performed in high-dimensional feature space using low dimensional space data input without knowing the transformation Φ. The elegance of using the kernel function is that one can deal with feature spaces of arbitrary dimensionality without having to compute the map explicitly. Any function that satisfies Mercer's condition (Vapnik, 1998) can be used as the kernel function. Common examples of the kernel function are the polynomial kernel and the Gaussian kernel (radial basis function). The radial basis function (RBF) is commonly used as the kernel for regression:

$$K(x_i, x_j) = \exp(-{\|x_i - x_j\|^2}/{2\sigma^2}) \tag{6.5}$$

Some common kernels are shown in Table 6.1.

The ε-insensitive loss function is the most widely used cost function (Muller *et al*., 1998). The function is in the form:

$$\Gamma(f(x) - y) = \begin{cases} |f(x) - y| - \varepsilon, & for \quad |f(x) - y| \geq \varepsilon \\ 0 & otherwise \end{cases} \tag{6.6}$$

By solving the quadratic optimization problem in (6.4), the regression risk in equation (6.2) and the ε-insensitive loss function (6.6) can be minimized:

Table 6.1: Common kernel functions

Kernel	Function	
Linear	$x \cdot y$	
Polynomial	$K(x_i, x_j) =< x_i, x_j >^d$	d is a positive *integer*
Radial Basis Function (RBF)	$K(x_i, x_j) = \exp(-\|x_i - x_j\|^2 / 2\sigma^2)$	σ is a user defined value

$$\frac{1}{2}\sum_{i,j=1}^{\ell}(\alpha_i^* - \alpha_i)(\alpha_j^* - \alpha_j)k(x_i, x_j) - \sum_{i=1}^{\ell}\alpha_i^*(y_i - \varepsilon) - \alpha_i(y_i + \varepsilon)$$

subject to

$$\sum_{i=1}^{\ell}\alpha_i - \alpha_i^* = 0, \quad \alpha_i, \alpha_i^* \in [0, C] \tag{6.7}$$

The Lagrange multipliers, α_i *and* α_i^*, represent solutions to the above quadratic problem that act as forces pushing predictions towards target value y_i. Only the non-zero values of the Lagrange multipliers in equation (6.7) are useful in forecasting the regression line and are known as support vectors. For all points inside the ε-tube, the Lagrange multipliers equal to zero do not contribute to the regression function. Only if the requirement $|f(x) - y| \geq \varepsilon$ (see Figure 6.1) is fulfilled, Lagrange multipliers may be non-zero values and used as support vectors.

The constant *C* introduced in equation (6.2) determines penalties to estimation errors. A large *C* assigns higher penalties to errors so that the regression is trained to minimize error with lower generalization while a small assigns fewer penalties to errors; this allows the minimization of margin with errors, thus higher generalization ability. If *C* goes to infinitely large, SVR would not allow the occurrence of any error and result in

a complex model, whereas when C goes to zero, the result would tolerate a large amount of errors and the model would be less complex.

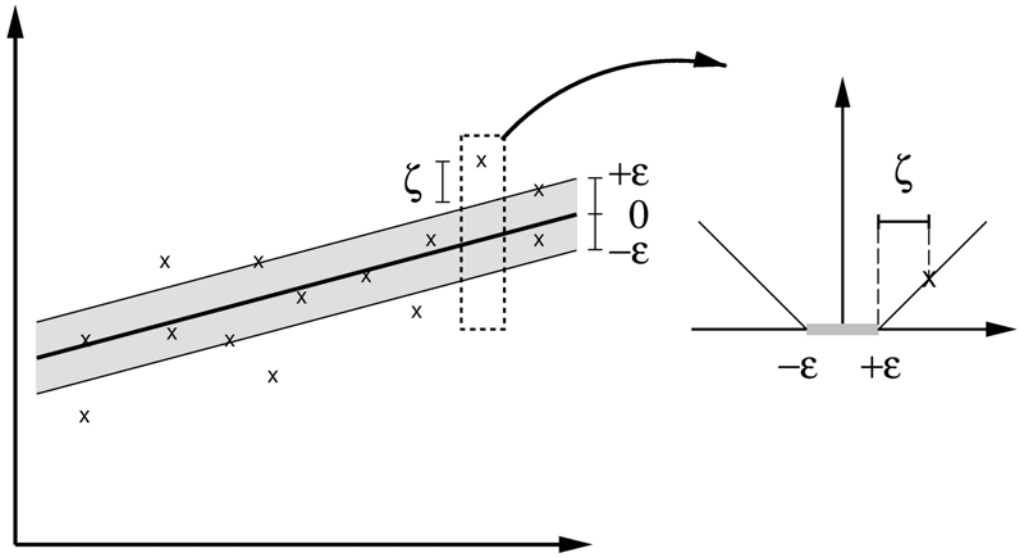


Figure 6.1: ε-insensitive regressor; in SV regression, a tube with radius ε is fitted to the data and positive slack variables ζ_i measuring the points lying outside of the tube.

Now, we have solved the value of w in terms of the Lagrange multipliers. For the variable b, it can be computed by applying Karush-Kuhn-Tucker (KKT) conditions which, in this case, implies that the product of the Lagrange multipliers and constrains has to equal zero:

$$\begin{aligned} &\alpha_i(\varepsilon + \zeta_i - y_i + (w, x_i) + b) = 0 \\ &\alpha_i^*(\varepsilon + \zeta_i^* + y_i - (w, x_i) - b) = 0 \end{aligned} \tag{6.8}$$

and

$$\begin{aligned} &(C - \alpha_i)\zeta_i = 0 \\ &(C - \alpha_i^*)\zeta_i^* = 0 \end{aligned} \tag{6.9}$$

where ζ_i and ζ_i^* are slack variables used to measure errors outside the ε-tube. ε is called the tube size and it is equivalent to the approximation accuracy placed on the training data points. Both C and ε are user-prescribed parameters. Since $\alpha_i, \alpha_i^* = 0$ and $\zeta_i^* = 0$ for $\alpha_i^* \in (0, C)$, b can be computed as follows:

$$b = y_i - (w, x_i) - \varepsilon \quad for\ \alpha_i \in (0, C)$$
$$b = y_i - (w, x_i) + \varepsilon \quad for\ \alpha_i^* \in (0, C) \tag{6.10}$$

Putting it all together, we can use SVM and SVR without knowing the transformation.

From the implementation point of view, training SVM is equivalent to solving the linearly constrained quadratic programming problem (6.6) with the number of variables twice as that of the number of training data points. The sequential minimal optimization (SMO) algorithm extended by Scholkopf and Smola (Smola, 1998; Smola, 1998b) is very effective in training SVM for solving the regression estimation problem.

In section 6.5, we apply our inferential result of SVR based on the general type of ε-insensitive loss function to the regression of financial data, for example, indices and stock prices. By applying regression to the data, we can build a dynamic system to model the data and hence use the system for predicting future prices. However, before applying the support vector regression, we preprocess the data with weighted kernel *k*-means clustering algorithm for getting improved prediction results. This algorithm is described in the next section.

6.3 Proposed Weighted Kernel K-Means with Neighborhood Constraints

Because, in the financial time series, the recent data points could provide more important information than the distant data points, before applying the SVR method, we

preprocess the data using a weighted kernel-based clustering algorithm incorporating neighborhood constraints. The developed algorithm is described in this section.

It has been reviewed in (Awan & Sap, 2006b) that there are some limitations with the *k*-means method, especially for handling complex data. Among these, the important issues/problems that need to be addressed are: i) non-linear separability of data in input space, ii) outliers and noise, iii) auto-correlation in the data, iv) high dimensionality of data. Although kernel methods offer power to deal with non-linearly separable and high-dimensional data but the current methods have some drawbacks. Both (Ben-Hur *et al*., 2001; Camastra & Verri, 2005) are computationally very intensive, unable to handle large datasets and autocorrelation in the data. The method proposed in (Ben-Hur *et al*., 2001) is not feasible to handle high dimensional data due to computational overheads, whereas the convergence of (Camastra & Verri, 2005) is an open problem. With regard to addressing these problems, we propose an algorithm–weighted kernel *k*-means with neighbourhood constraints–in order to handle autocorrelation, noise and outliers present in the data.

Clustering has received a significant amount of renewed attention with the advent of nonlinear clustering methods based on kernels as it provides a common means of identifying structure in complex data (Camastra & Verri, 2005; Dhillon *et al*., 2004; Girolami, 2002; Awan & Sap, 2006a-b; Ben-Hur *et al.,* 2001). We first fix the notation: let $X = \{ x_i \}_{i=1, \dots, n}$ be a data set with $x_i \in \mathrm{R}^{\mathrm{N}}$. We call codebook the set $W = \{ c_j \}_{j=1, \dots, k}$ with $c_j \in \mathrm{R}^{\mathrm{N}}$ and $k << n$. The Voronoi set (V_j) of the codevector c_j is the set of all vectors in X for which c_j is the nearest vector, i.e.

$$V_j = \{x_i \in X \,|\, j = arg \min_{j=1,\dots,k} \|x_i - c_j\|\}$$

For a fixed training set X the quantization error $E(W)$ associated with the Voronoi tessellation induced by the codebook W can be written as:

$$E(W) = \sum_{j=1}^{k} \sum_{x_i \in V_j} \left\| x_i - c_j \right\|^2 \tag{6.11}$$

K-means is an iterative method for minimizing the quantization error $E(W)$ by repeatedly moving all codevectors to the arithmetic mean of their Voronoi sets. In the case of finite data set X and Euclidean distance, the centroid condition reduces to

$$c_j = \frac{1}{|V_j|} \sum_{x_i \in V_j} x_i \tag{6.12}$$

where $|V_j|$ denotes the cardinality of V_j . Therefore, *k*-means is guaranteed to find a local minimum for the quantization error (Gray, 1992; Lloyd, 1982).

The *k*-means clustering algorithm can be enhanced by the use of a kernel function; by using an appropriate nonlinear mapping from the original (input) space to a higher dimensional feature space, one can extract clusters that are non-linearly separable in input space. Usually the extension from *k*-means to kernel *k*-means is realized by expressing the distance in the form of kernel function. The kernel *k*-means algorithm can be generalized by introducing a weight for each point x, denoted by $u(x)$. This generalization would be powerful for making the algorithm more robust to noise and useful for handling auto-correlation in the data. Using the non-linear function ϕ, the objective function of weighted kernel *k*-means can be defined as:

$$E(W) = \sum_{j=1}^{k} \sum_{x_i \in V_j} u(x_i) \left\| \phi(x_i) - c_j \right\|^2 \tag{6.13}$$

where,

$$c_j = \frac{\sum_{x_j \in V_j} u(x_j)\phi(x_j)}{\sum_{x_j \in V_j} u(x_j)} \tag{6.14}$$

Why weighting? As, in the financial time series, the recent data points could provide more important information than the distant data points, in such cases the use of

weighted means instead of means becomes necessary. A 'weight' attributed to a mean or other value usually signifies importance. If weighting is not used, then each value that enters into a calculation of mean would have the same importance. Sometimes this is not a realistic or fair way to calculate a mean. Individual observations often, for various reasons, are of varying importance to the total, or average value.

The Euclidean distance from $\phi(x)$ to center c_j is given by (all computations in the form of inner products can be replaced by entries of the kernel matrix) the following eq.

$$\left\| \phi(x_i) - \frac{\sum_{x_j \in V_j} u(x_j)\phi(x_j)}{\sum_{x_j \in V_j} u(x_j)} \right\|^2 = K(x_i, x_i) - 2\frac{\sum_{x_j \in V_j} u(x_j)K(x_i, x_j)}{\sum_{x_j \in V_j} u(x_j)} + \frac{\sum_{x_j, x_l \in V_j} u(x_j)u(x_l)K(x_j, x_l)}{(\sum_{x_j \in V_j} u(x_j))^2} \tag{6.15}$$

In the above expression, the last term is needed to be calculated once per each iteration of the algorithm, and is representative of cluster centroids. If we write the last term in (6.15) as

$$L_k = \frac{\sum_{x_j, x_l \in V_j} u(x_j)u(x_l)K(x_j, x_l)}{(\sum_{x_j \in V_j} u(x_j))^2} \tag{6.16}$$

With this substitution, eq. (6.15) can be re-written as

$$\left\| \phi(x_i) - \frac{\sum_{x_j \in V_j} u(x_j)\phi(x_j)}{\sum_{x_j \in V_j} u(x_j)} \right\|^2 = K(x_i, x_i) - 2\frac{\sum_{x_j \in V_j} u(x_j)K(x_i, x_j)}{\sum_{x_j \in V_j} u(x_j)} + L_k \tag{6.17}$$

In modelling the financial time series, one key problem is its high noise, or the effect of some data points, called outliers, which differ greatly from others(Cao & Tay, 2003, Yang *et al*., 2004; Huang *et al*., 2005). Learning observations with outliers without awareness may lead to fitting those unwanted data and may corrupt the

approximation function. This will result in the loss of generalization performance in the test phase. Hence, handling the noise, and detecting and removing the outliers are very important. Specific techniques, e.g., a robust SVR network (Chuang *et al*., 2002) and a weighted Least Squares SVM (Suykens *et al*., 2001) have been proposed to enhance the robust capability of SVR. These methods would either involve extensive computation or would not guarantee the global optimal solution.

For increasing the robustness of fuzzy c-means to noise, an approach is proposed in (Ahmed *et al*., 2002). Here we propose a modification to the weighted kernel *k*-means to increase the robustness to noise and to account for autocorrelation in the data. It can be achieved by a modification to eq. (6.13) by introducing a penalty term containing neighborhood information. This penalty term acts as a regularizer and biases the solution toward piecewise-homogeneous labeling. Such regularization is also helpful in finding clusters in the data corrupted by noise. It also gives a smoothing effect. The objective function (6.13) can, thus, be written, with the penalty term, as:

$$E(W) = \sum_{j=1}^{k} \sum_{x_i \in V_j} u(x_i) \left\| \phi(x_i) - c_j \right\|^2 + \frac{\gamma}{N_R} \sum_{j=1}^{k} \sum_{x_i \in V_j} u(x_i) \sum_{r \in N_k} \left\| \phi(x_r) - c_j \right\|^2 \tag{6.18}$$

where N_k stands for the set of neighbors that exist in a window around x_i (for instance, 2 data points on either side of a data point in the time series stock market data) and N_R is the cardinality of N_k. The parameter γ controls the effect of the penalty term. The relative importance of the regularizing term is inversely proportional to the accuracy of clustering results.

The following can be written for kernel functions, using the kernel trick for distances (Scholkopf, 2000)

$$\left\| \phi(x_i) - c_j \right\|^2 = K(x_i, x_i) - 2K(x_i, c_j) + K(c_j, c_j)$$

If we adopt the Gaussian radial basis function (RBF), then $K(x, x) = 1$, so eq. (6.18) can be simplified as

$$E(W) = 2\sum_{j=1}^{k}\sum_{x_i \in V_j} u(x_i)(1 - K(x_i, c_j)) + \frac{\gamma}{N_R}\sum_{j=1}^{k}\sum_{x_i \in V_j} u(x_i)\sum_{r \in N_k}(1 - K(x_r, c_j)) \tag{6.19}$$

The distance in the last term of eq. (6.18), can be calculated as

$$\left\| \phi(x_r) - \frac{\sum_{x_j \in V_j} u(x_j)\phi(x_j)}{\sum_{x_j \in V_j} u(x_j)} \right\|^2 = 1 - 2\frac{\sum_{x_j \in V_j} u(x_j)K(x_r, x_j)}{\sum_{x_j \in V_j} u(x_j)} + \frac{\sum_{x_j, x_l \in V_j} u(x_j)u(x_l)K(x_j, x_l)}{(\sum_{x_j \in V_j} u(x_j))^2} \tag{6.20}$$

As first term of the above equation does not play any role for finding minimum distance, so it can be omitted, however.

$$\left\| \phi(x_r) - \frac{\sum_{x_j \in V_j} u(x_j)\phi(x_j)}{\sum_{x_j \in V_j} u(x_j)} \right\|^2 = 1 - 2\frac{\sum_{x_j \in V_j} u(x_j)K(x_r, x_j)}{\sum_{x_j \in V_j} u(x_j)} + L_k = 1 - \beta_r + L_k \tag{6.21}$$

For RBF, eq. (6.17) can be written as

$$\left\| \phi(x_i) - \frac{\sum_{x_j \in V_j} u(x_j)\phi(x_j)}{\sum_{x_j \in V_j} u(x_j)} \right\|^2 = 1 - 2\frac{\sum_{x_j \in V_j} u(x_j)K(x_i, x_j)}{\sum_{x_j \in V_j} u(x_j)} + L_k \tag{6.22}$$

As first term of the above equation does not play any role for finding minimum distance, so it can be omitted.

We have to calculate the distance from each point to every cluster representative. This can be obtained from eq. (6.18) after incorporating the penalty term containing neighborhood information by using eq. (6.21) and (6.22). Hence, the effective minimum distance can be calculated using the expression:

$$-2\frac{\sum_{x_j \in V_j} u(x_j)K(x_i, x_j)}{\sum_{x_j \in V_j} u(x_j)} + L_k + \frac{\gamma}{N_R}\sum_{r \in N_k}(\beta_r + L_k) \tag{6.23}$$

Now, the algorithm, weighted kernel *k*-means with neighborhood constraints, can be written as in Figure 6.2.

Algorithm WK-means: weighted kernel *k*-means (weighted kernel *k*-means with neighborhood constraints)

WK_means (K, k, u, N, γ, t)

Input: K: kernel matrix, k: number of clusters, u: weights for each point, set $t > 0$ to a very small value for termination, N: information about the set of neighbors around a point (for instance, 2, 3 or 4 data points on either side of a data point in the time series stock market data, depending on the size of the window around a data point), γ : penalty term parameter,

Output: $c_1,...,c_k$: partitioning of the points

1. Initialize the k clusters: $c_1=0, \ldots . , c_k=0$
2. Set $i = 0$.
3. For each cluster, compute $L(k)$ using expression (6.16)
4. For each point x, find its new cluster index as

   $j(x) = arg\ min_j \left\| \phi(x) - c_j \right\|^2$ using expression (6.23),

5. Compute the updated clusters as

   $w_j^{(i+1)} = \{x : j(x)=j\}$

6. Repeat steps 3 to 5 until the following termination criterion is met:

   $\left\| W_{new} - W_{old} \right\| < t$

   where, $W = \{c_1, c_2, c_3, ..., c_k\}$ are the vectors of cluster centroids.

Figure 6.2: Algorithm WK-means: Weighted Kernel *k*-means (weighted kernel *k*-means with neighborhood constraints)

6.3.1 Handling outliers

This section briefly discusses about outliers, i.e., observations or values which appear to be inconsistent with their neighborhoods. Detecting outliers is useful in many applications of information systems and databases, including transportation, ecology, public safety, public health, climatology, location-based services, and severe weather prediction. Since outliers may make it difficult or time-consuming to arrive at an effective solution for the SVM (Cao & Tay, 2003; Yang *et al.*, 2004b), we handle these outliers in the financial time series at this preprocessing stage. They are replaced with the closest centroid values.

We can examine how eq. (6.23) makes the algorithm robust to outliers. As $K(x_i, x_j)$ measures the similarity between x_i and x_j, and when x_i is an outlier, i.e., x_i is far from the other data points, then $K(x_i, x_j)$ will be very small. So, the second term in the above expression will get very low value or, in other words, the weighted sum of data points will be suppressed. The total expression will get higher value and hence results in robustness by not assigning the point to the cluster.

6.3.2 Scalability

The pruning procedure used in (Dhillon *et al*., 2001; Elkan, 2003) can be adapted to speed up the distance computations in the weighted kernel *k*-means algorithm. The acceleration scheme is based on the idea that we can use the triangle inequality to avoid unnecessary computations. According to the triangle inequality, for a point x_i, we can write, $d(x_i, c_j^n) \geq d(x_i, c_j^o) - d(c_j^o, c_j^n)$. The distances between the corresponding new and old centers, $d(c_j^o, c_j^n)$ for all *j,* can be computed. And this information can be stored in a $k \times k$ matrix. Similarly, another $k \times n$ matrix can be kept that contains lower bounds for the distances from each point to each center. The distance from a point to its cluster centre is exact in the matrix for lower bounds. Suppose, after a single iteration, all

distances between each point and each center, $d(x_i, c_j^o)$, are computed. In the next iteration, after the centers are updated, we can estimate the lower bounds from each point x_i to the new cluster center, c_j^n, using $d(c_j^o, c_j^n)$ calculations and the distances from the previous iteration, i.e., we calculate the lower bounds as $d(x_i, c_j^o) - d(c_j^o, c_j^n)$. The distance from x_i to c_j^n is computed only if the estimation is smaller than distance from x_i to its cluster center. This estimation results in sufficient saving in computational time. Once we have computed lower bounds and begin to compute exact distances, the lower bound allows us to determine whether or not to determine remaining distances exactly.

6.4 Experimental Settings for SVR-Forecaster

As already mentioned, the SVM based prediction system is developed as part of the overall Stock Market Prediction System. Therefore, Figure 3.2 is redrawn here as Figure 6.3. A view of the mainmenu window of the SVR-FDT Financial Forecaster is shown in Figure 6.4. A user can select whether to use KnowledgeMiner (based on Fuzzy predictive decision tree) or SVR-Forecaster (based on support vector regression).

In this section, we conduct experiments to illustrate the usefulness of SVM for financial prediction. In our experiment, we use the daily closing prices of Kuala Lumpur Stock Exchange (KLSE) Index from 01 April 2002 up to the end of December 2004. Among these data points, the training data points cover the period from 01 April 2002 up to the end of December 2003, while the data points staring from 01 March 2004 up to the end of December 2004 are used as the test data. A part of sample data is given in appendix E.

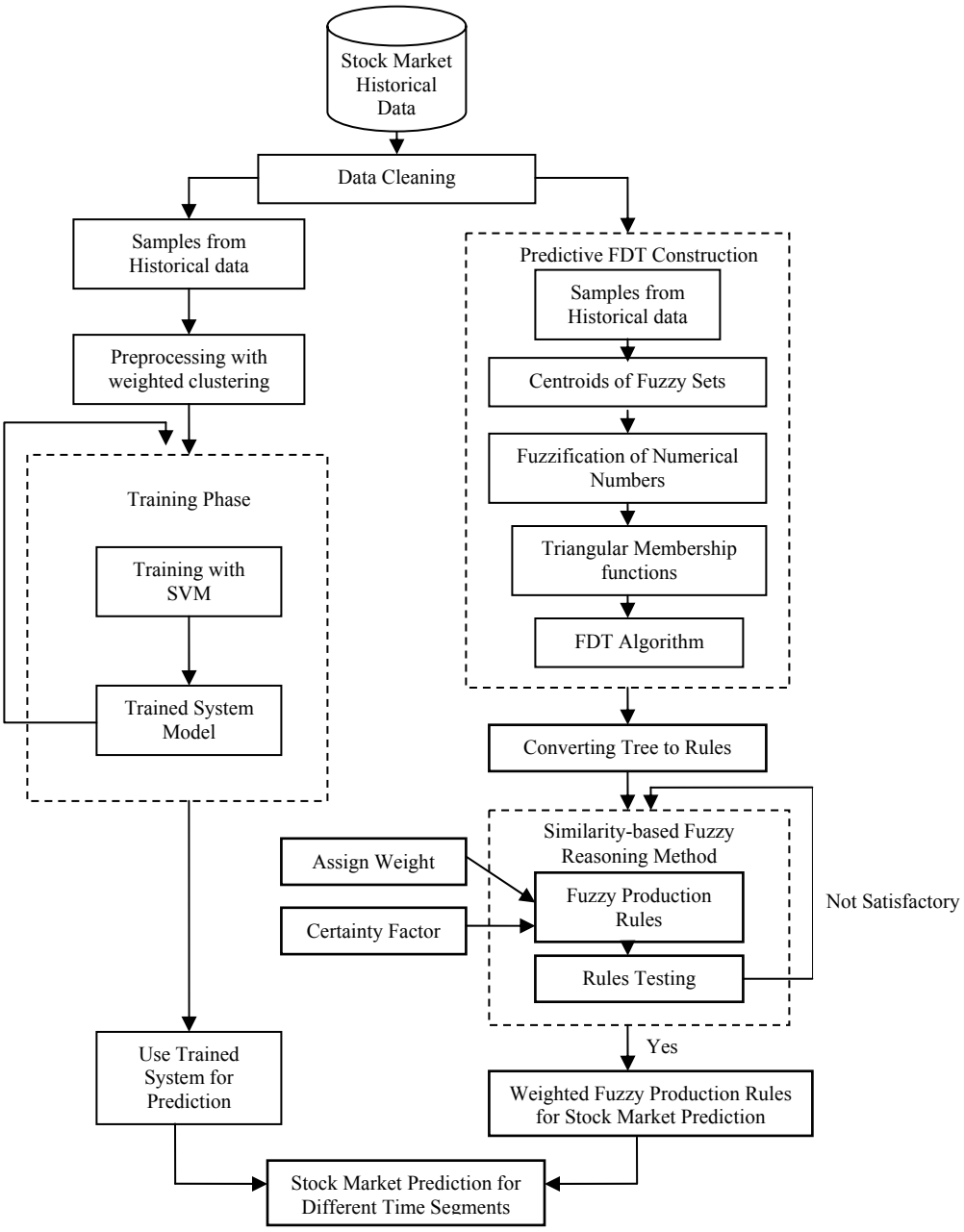


Figure 6.3: System development process

Figure 6.4: MainMenu window of the overall Stock Market Prediction System: A user can select whether to use KnowledgeMiner (based on Fuzzy predictive decision tree) or SVR-Forecaster (based on support vector regression)

6.4.1 Technical Indicators

The financial market is a complex, evolutionary, and non-linear dynamical system (Abu-Mostafa & Atiya, 1996; Huang *et al.,* 2005). The field of financial forecasting is characterized by data intensity, noise, non-stationary, unstructured nature, high degree of uncertainty, and hidden relationships (Hall, 1994; Huang *et al.,* 2005). Many factors interact in finance including political events, general economic conditions, and traders' expectations. Therefore, predicting finance market price movements is quite difficult.

The tendency in the field of financial forecasting is to use state variables which are fundamental or macro economic variables. On the other hand, technical analysis, also known as charting, is used widely in real life. It is based on some technical indicators. These indicators help the investors to buy or sell the underlying stock. These indicators are computed by using stock price and volume overtime, and are vital for better prediction of stock index movement (Huang *et al*., 2005; Cao & Tay, 2003; Kim, 2003).

Choosing a suitable forecasting horizon is the first step in financial forecasting. From the trading aspect, the forecasting horizon should be sufficiently long so that the over-trading resulting in excessive transaction costs could be avoided. From the prediction aspect, the forecasting horizon should be short enough as the persistence of financial time series is of limited duration. As suggested by Thomason (Thomason, 1999), a forecasting horizon of five days is a suitable choice for the daily data. As the precise values of the daily prices is often not as meaningful to trading as its relative magnitude, and also the high-frequency components in financial data are often more difficult to successfully model, the original closing price is transformed into a five-day *relative difference in percentage of price* (RDP). As mentioned by Thomason, there are four advantages in applying this transformation. The most prominent advantage is that the distribution of the transformed data will become more symmetrical and will follow more closely to a normal distribution.

The input variables are determined from four lagged RDP values based on five-day periods (RDP-5, RDP-10, RDP-15, and RDP-20) and one transformed closing price (EMA_{15}) which is obtained by subtracting a 15-day exponential moving average from the closing price. The subtraction is performed to eliminate the trend in price in the data set. The optimal length of the moving day is not critical, but it should be longer than the forecasting horizon of five days (Thomason, 1999). EMA_{15} is used to maintain as much of the information contained in the original closing price as possible since the application of the RDP transform to the original closing price may remove some useful information. In addition to the above mentioned five input variables, the other five input

variables are: moving average convergence divergence, on balance volume, volatility, daily closing price, and the indices obtained from the clustering algorithm described in section 6.3. The calculations for the indicators are given in Table 6.2.

These indicators are explained briefly here.

Exponential Moving Average (EMA)*:* The EMA is used to reduce the lag in simple moving average. Exponential moving averages reduce the lag by applying more weight to recent prices relative to older prices. The weighting applied to the most recent price depends on the length of the moving average (Ince & Trafalis, 2004; Cao & Tay, 2003).

Moving Average Convergence Divergence (MACD)*:* It is one of the simplest and most reliable indicators available. The MACD is defined as the difference between two exponential moving averages, and it is commonly used to predict market trends in financial markets.

Volatility. The volatility denotes the range of the highest and lowest prices in one day, and it is often used as a measure of the market risk.

On Balance Volume (OBV). OBV moves in the same direction as price, i.e. as price increases, the OBV will gain in magnitude. As OBV can relate the volume into price, it is also used here as input.

Relative Strength Index (RSI). RSI acts as a momentum indicator that measures stock's price relative to its performance (Achelis, 1995). It is less affected by sharp rises or drops in the stock's trading activity, and it follows an oscillator that ranges from 0 to 100.

The calculation for all indicators is listed in Table 6.2.

Table 6.2: Independent and dependent variables (input and output variables)

Indicator	Calculation
EMA_{15}	$P_i - EMA_{15,\,i}$
RDP-5	$(P_i - P_{(i-5)})/P_{(i-5)} * 100$
RDP-10	$(P_i - P_{(i-10)})/P_{(i-10)} * 100$
RDP-15	$(P_i - P_{(i-15)})/P_{(i-15)} * 100$
RDP-20	$(P_i - P_{(i-20)})/P_{(i-20)} * 100$
MACD	$EMA_{10,\,i} - EMA_{20,\,i}$
OBV	$P_i \geq P_{(i-1)}\ obv+ = volume(i)$ $P_i \leq P_{(i-1)}\ obv- = volume(i)$
Volatility	$k * sqrt\left(\frac{1}{n} * \sum_{i=1}^{n} \log^2(H_i/L_i)\right) \quad (k = 80)$
RSI	$100 - \frac{100}{1 + (\sum_{i=1}^{n} U_i/n)/(\sum_{i=1}^{n} D_i/n)}$
RDP+5 (output)	$(P_{(i+5)} - P_i) / P_i * 100$ $P_i = EMA_{3,\,i}$

where U means upward price change, and D means downward price change, $EMA_{n,\,i}$ is the n-day exponential moving average of the ith day; P_i, H_i, L_i are the closing, highest and lowest price of the ith day.

6.5 Experimental Results

In this investigation, the Gaussian function is used as the kernel function of the SVM because Gaussian kernels tend to give good performance under general smoothness assumptions. Consequently, they are especially useful if no additional knowledge of the data is available (Smola, 1998b; Trafalis and Ince, 2002). Before running the algorithm, we need to determine some parameters. They are *C*, the cost of error; parameter of kernel function, and the margins. After performing a cross-validation in the first training data, we set values of parameters. Our experiments show that a width value, σ, of the Gaussian function of 10 is found to produce the best possible results. *C* and ε are chosen to be 10 and 10^{-3} respectively, because these values produced the best possible results according to the validation set. The SVR-forecaster adopts the procedural steps as given in Figure 6.5.

The prediction performance is evaluated using the statistical metric: Mean Squared Error (MSE). MSE is a measure of the deviation between actual and predicted values. The smaller the values of MSE, the closer are the predicted time series values to that of the actual values. The experiments are conducted on a Pentium 4, with 1.7 GHZ, 1 GB RAM and WindowsXP. With these configurations, the prediction results are obtained within seconds. Main window of the SVR-Forecaster is shown in Figure 6.6. The user can select the company name, along with starting and ending dates of the past, to see the past behavior of stock price data of the company. Moreover, the user can select for how many days (for instance, up to 10 days) he wants to predict the future trend of stock price of the selected company. The graph in Figure 6.8 is showing past and future predicted stock price values.

Step 1: Obtain a set of training and testing patterns in the form of $\{(x_1, y_1),....,(x_\ell, y_\ell)\}$

Step 2: Scale the data in the range [-1, +1]

Step 3: Apply weighted clustering algorithm

Step 4: Set up support vector regression model by setting values of C (cost of error), and the ε (cost function)

Step 5: Select kernel function; for RBF, set the value of σ and proceed

Step 6: Use cross validation to find best parameters; set number of validation sets

Step 7: Start training by initializing input vectors and parameters

Step 8: Test the training and validation set using a model selected for best parameters

Repeat steps 4 to 8 until the lowest MSE is found

Step 9: Replace the parameters (C and ε) in training data with parameters with lowest MSE

Step 10: Testing data set is used for predictions. For RBF, set new values of σ.

Step 11: Get best SVM model using the training data set with the best parameters.

Step 12: Start prediction with testing data set and the best SVM model obtained in Step 11 (with post-processing)

Step 13: Calculate error between actual and predicted values using Mean Squared Error (MSE).

Figure 6.5: Procedural steps for SVR-Forecaster

The behaviour of the MSE (Mean Squared Error) is illustrated in Figure 6.9 and Figure 6.10. It is evident that the MSE on the training set is very low during the entire training period as shown in Figure 6.9. The graphs of actual and predicted stock price values for the training period of 100 days are almost overlapping. However, the MSE on the test set fluctuates and is higher than the MSE for the training period, as shown in Figure 6.10. It can be seen in Figure 6.10, the graphs of actual and predicted prices for the test period of 101 to 110 days are deviating from each other, whereas as the graphs for the training period until 100 days are overlapping.

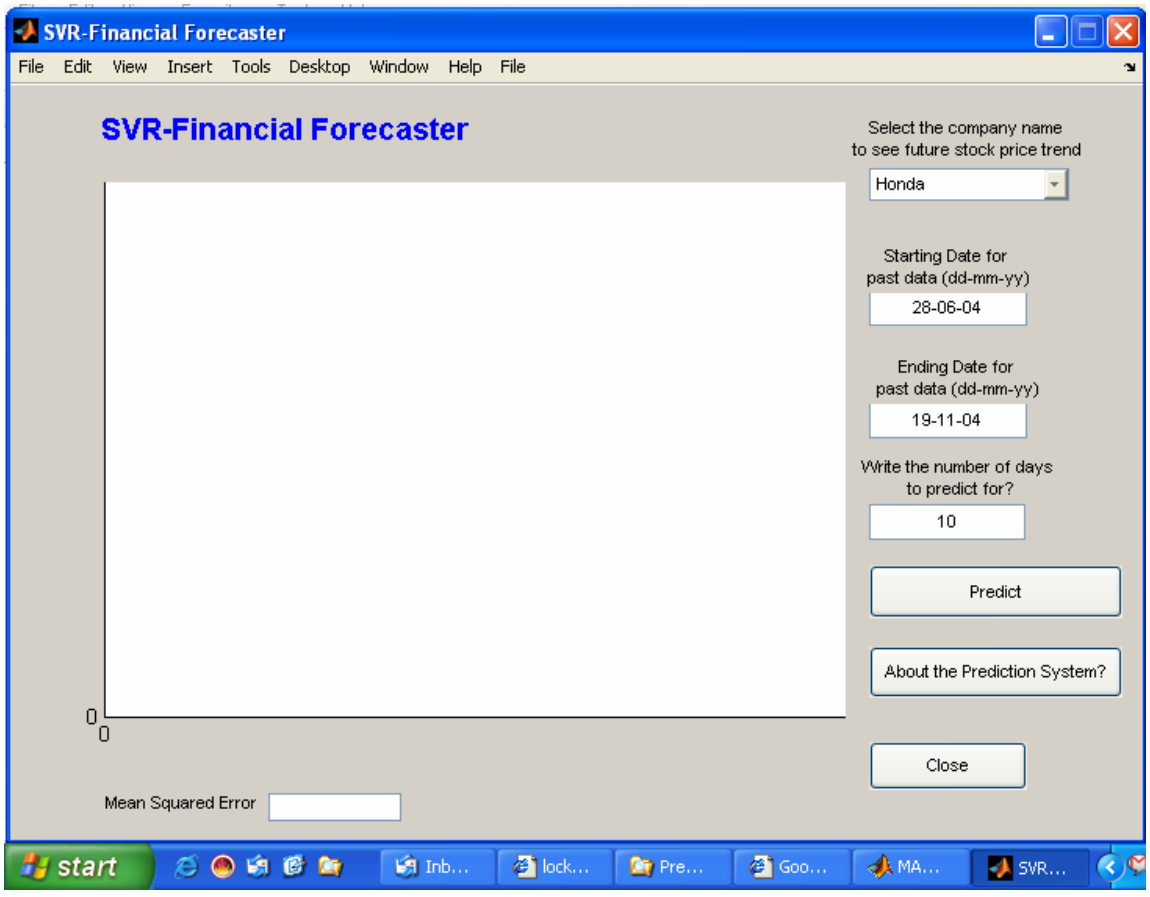


Figure 6.6: Selection window of SVR-Forecaster; the user can select the type of stocks, past data, and the number of days to predict the stock prices

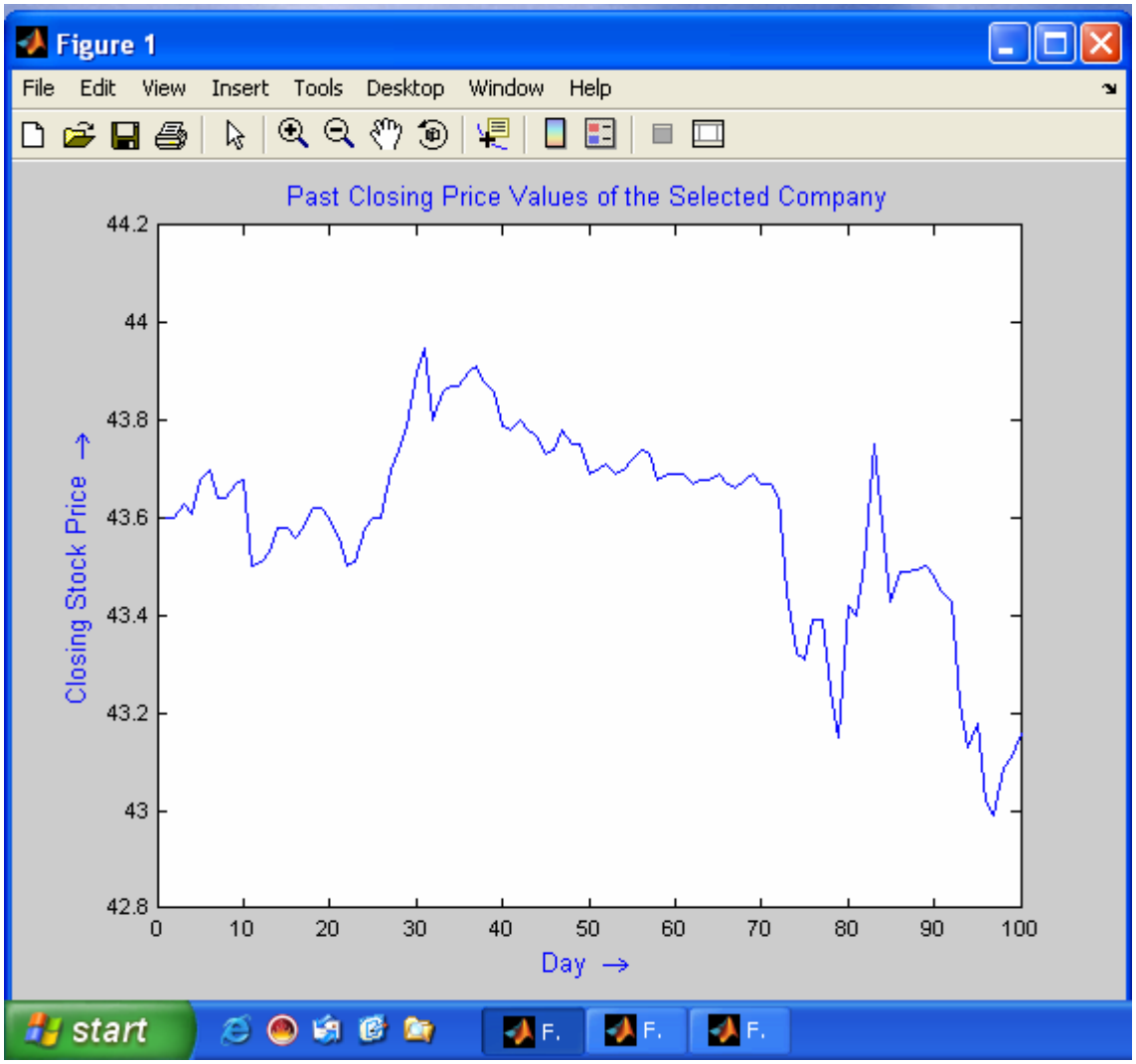


Figure 6.7: Window showing selected past stock data

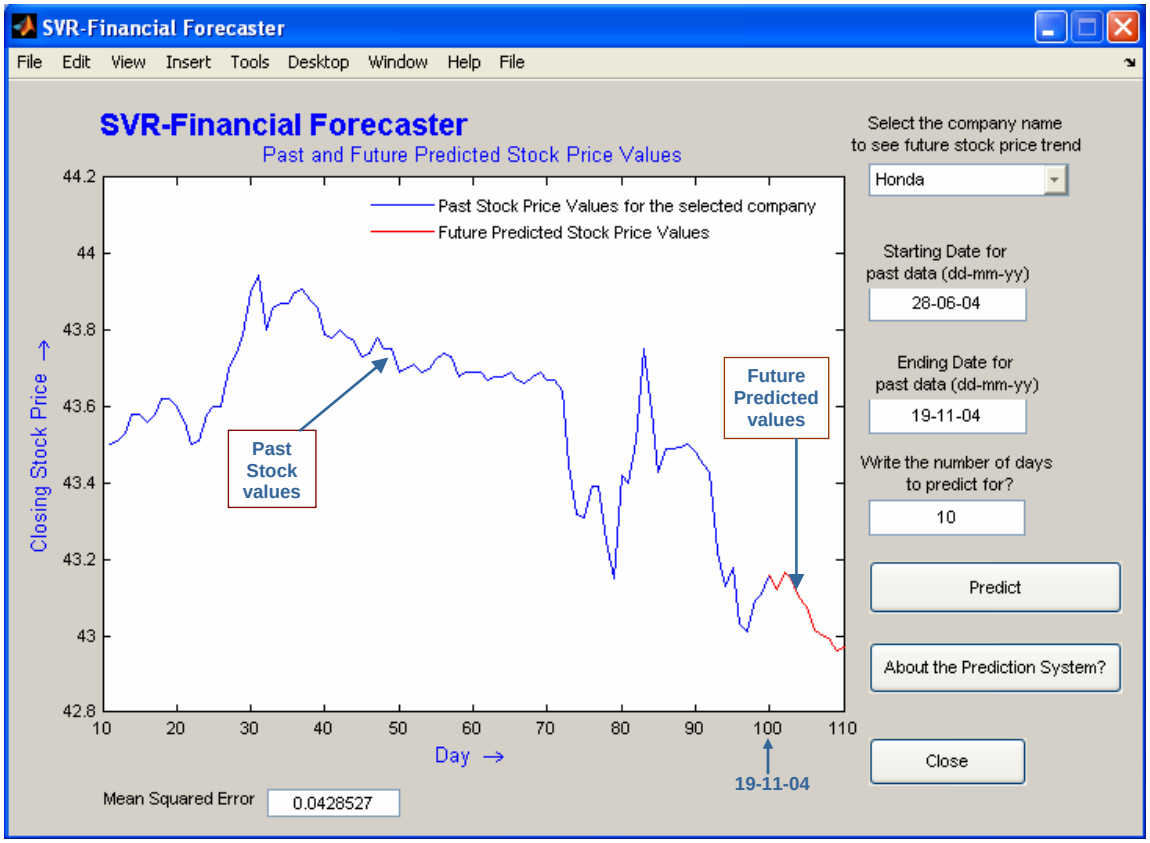


Figure 6.8: Main window of the SVR-Forecaster: user can select the company along with dates to see the past behavior and future trend of its stock price: Graph showing past and future predicted stock price values.

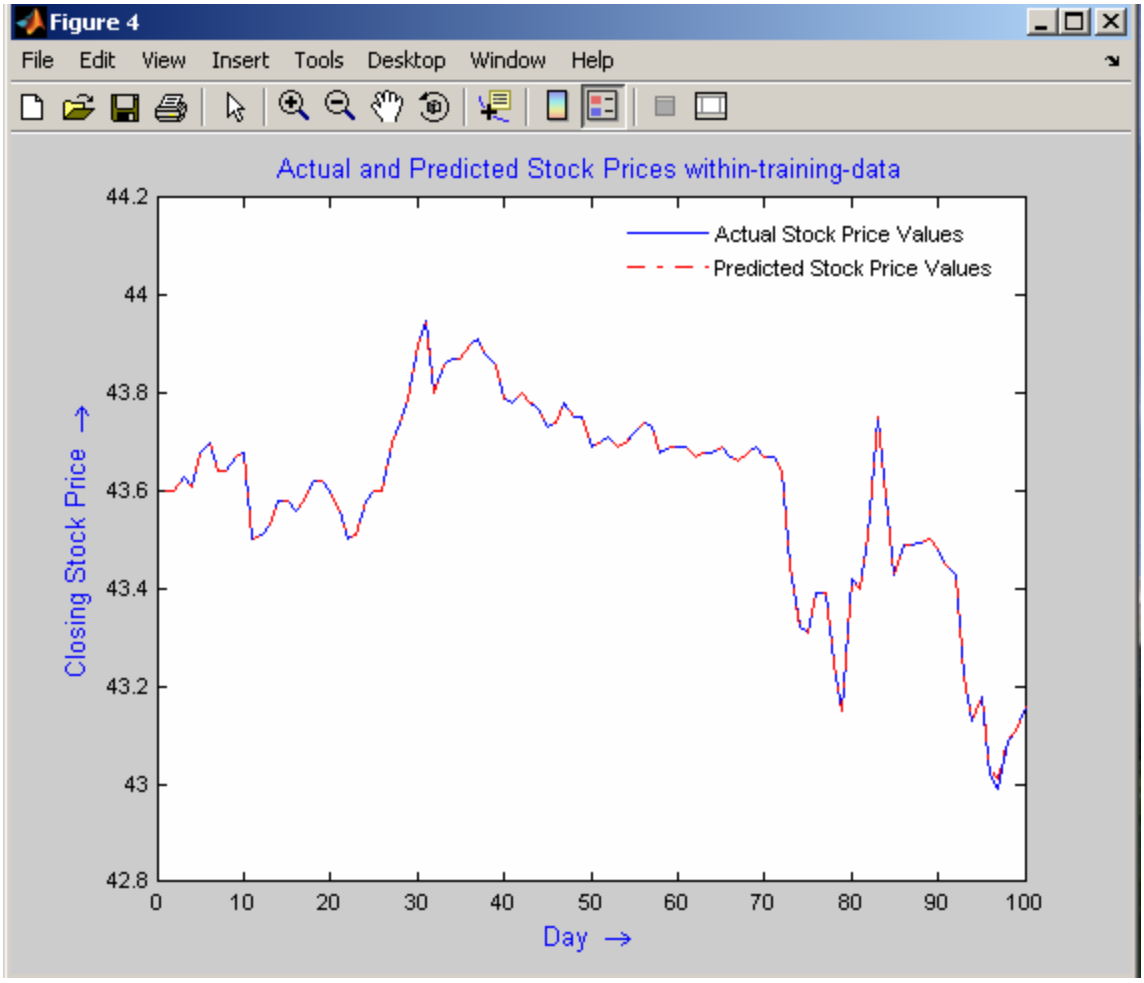


Figure 6.9: Actual and Predicted stock prices for a training period of 100 days

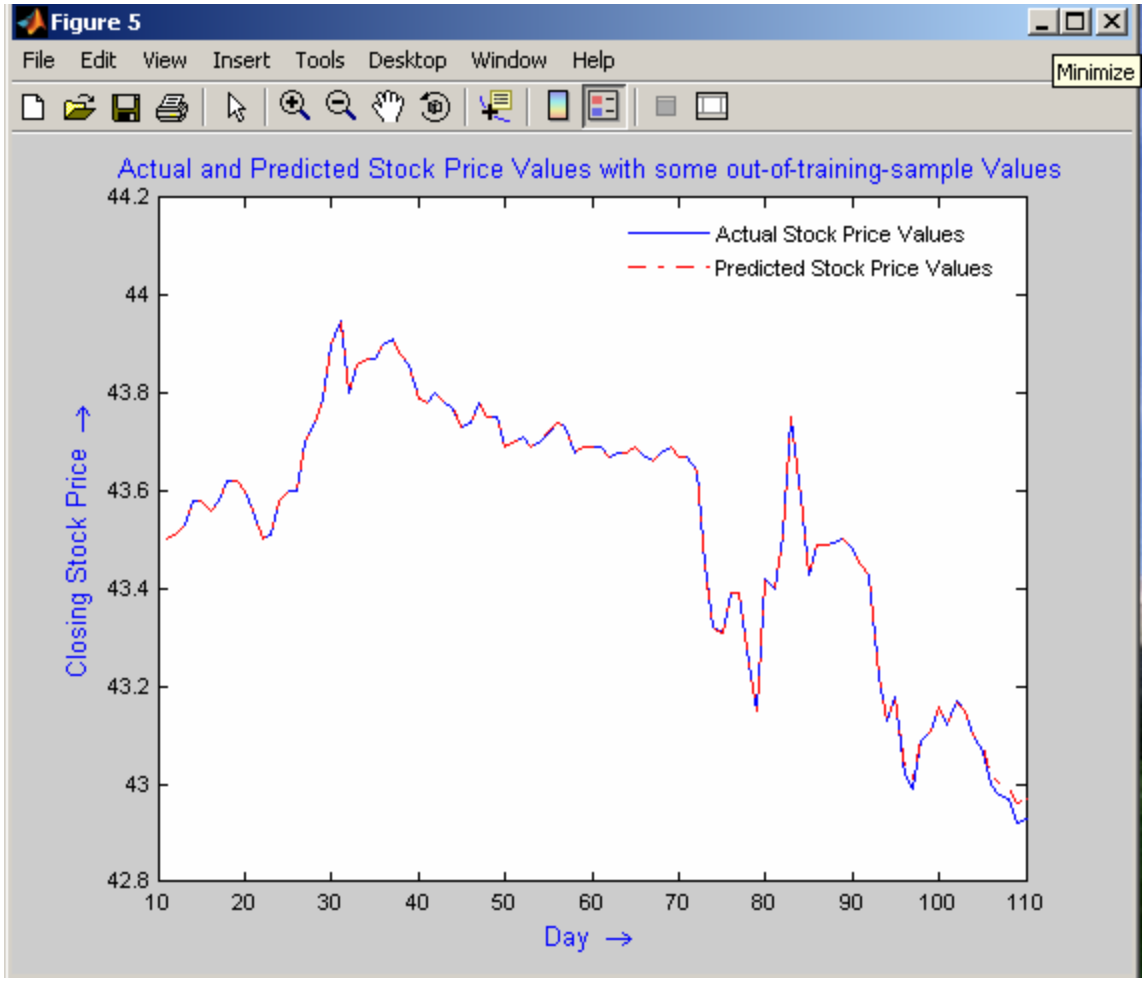


Figure 6.10: Actual and Predicted stock prices for training period of 100 days and test period of day 101-110

From Figure 6.10 it is evident that the predicted values obtained from SVM are closer to the actual values, indicating that there is a smaller deviation between the actual and predicted values. All the above results are found to be consistent with the statistical learning theory.

As already mentioned above, before running the algorithm, we need to determine some parameters. They are *C*, the cost of error; parameter of kernel function, and the

margins. After performing a cross-validation in the first training data, we set values of parameters. Our experiments show that a width value, σ, of the Gaussian function of 10 is found to produce the best possible results. C and ε are chosen to be 10 and 10^{-3} respectively, because these values produced the best possible results according to the validation set. Also, we use different values for the kernel parameter. Specifically, σ = 0.90, 2.5, 5, 7.5, 10, 12.5, 15. As we increase σ, the accuracy of the prediction also increases. This can be seen from Table 6.3.

For comparison of results, a standard three-layer BP neural network is used as a benchmark. There are ten nodes in the input layer, which is equal to the number of indicators (8), and daily closing price and indices from the clustering algorithm. The output node is equal to 1, whereas the number of hidden nodes is determined based on the validation set. This procedure is very simple. The number of hidden nodes is varied from a small value (say, three) to a reasonably big value (say, 25). For each chosen number of hidden nodes, the BP neural network is trained, and the averaged error on the validation sets is estimated. After the above procedure is repeated for every number of hidden nodes, the number of hidden nodes that produces the smallest averaged error on the validation sets is used, which could vary between different data sets. The learning rate is also chosen based on the validation set. The hidden nodes use the sigmoid transfer function and the output node uses the linear transfer function. The stochastic gradient descent method is used for training the BP neural network as it could give better performance than the batch training for large and nonstationary data sets. In the stochastic gradient descent training, the weights and the biases of the BP neural network are immediately updated after one training sample is presented.

So, there are 16 hidden nodes in the network for our data set. We set the learning rate as 0.005 and the momentum term as 0.9 and got the best results with the BP neural network. Comparison of results is given in Table 6.3. It is clear that SVM performed better than BP as the MSE value with SVM is 3.29 which is lower than that given by BP viz. 3.43.

Table 6.3: MSE values, for different values of σ, for SVM, and for BP

SVM (with different values of σ)							BP
0.9	2.5	5	7.5	10	12.5	15	
33.73	21.32	7.11	4.52	3.99	**3.29**	3.87	**3.43**

6.6 Summary

In this chapter, a stock market prediction system using support vector machine is discussed. It is developed in addition to the Fuzzy Decision Tree based system described in chapter 4 and 5. At the preprocessing stage of the SVM-based prediction system, weighted kernel based clustering algorithm with neighborhood constraints is incorporated for getting improved prediction results. Looking at the forecasting results presented by the two methods, i.e. FDT & SVR methods, the trader can confidently decide whether to buy a stock.

The use of SVM in financial forecasting is studied in this chapter. The support vector machine for regression is a robust technique for function approximation. The study has concluded that SVM provides a promising alternative to time series forecasting. In our experimentation, our SVR-based system gave better forecasting results than backpropagation neural networks. The SVM offers the following advantages:

(1) There is a smaller number of free parameters. C and ε are the only two free parameters of SVM if the kernel function has been considered;

(2) SVM forecasts better, as SVM provides a smaller MSE. This is because SVM adopts the Structural Risk Minimization Principle, eventually leading to better generalization than conventional techniques;

(3) Training SVM is faster. The regression function in SVM is only determined by the support vectors, and the number of support vectors is much smaller compared to the number of training samples.

CHAPTER 7

CONCLUSIONS

7.1 Introduction

As described in chapter 1, the aims of this project have been to build an accurate model for stock market prediction by using computational intelligence techniques, like classification, fuzzy sets and support vector machine, in order to make the methods accessible, user-friendly, and prepared for the broader population of economists, analysts, and financial professionals. The main objectives of this research have been to develop and enhance data mining algorithms for extracting the patterns of knowledge from stock market database. In addition, the developed algorithms need to be integrated for designing and building an intelligent knowledge-discovery system for the stock exchange environment. Moreover, assessment of performance of the developed system are also to be performed.

The work described in this report presents a significant advancement towards the aims of the project, and major objectives have been achieved. Financial time series forecasting is one of the most challenging applications of modern time series forecasting. The financial market is a complex, evolutionary, and non-linear dynamical system. The field of financial forecasting is characterized by data intensity, noise, non-stationary, unstructured nature and high degree of uncertainty. Many factors interact in finance including political events, general economic conditions, and traders'

expectations. Therefore, predicting finance market price movements is quite difficult. In this research, a number of algorithms have been developed, adapted, implemented and experimented with.

A working prototype of the stock market prediction system, named FDT-SVR Financial Forecaster, has been developed in this research. The overall stock market prediction system mainly consists of two parts: 1) based on Fuzzy decision tree (FDT); and 2) based on support vector regression (SVR). The FDT based subsystem is described in Chapter 4 and 5, whereas the SVR based subsystem is described in chapter 6.

In the first part of the prediction system, we use predictive FDT and similarity-based fuzzy reasoning method for the prediction of time series stock market data. The predictive FDT is compared with the three standard fuzzy decision tree algorithms and the experimental results given in the previous chapters show the promising performance with regards to comprehensibility (number of rules), and complexity (number of nodes). Similarly, similarity-based fuzzy reasoning method is used for mining the significant WFPRs that are used for the prediction of stock market data.

The second part of the prediction system is developed using support vector machine approach. At the preprocessing stage of the SVM-based prediction system, weighted kernel based clustering algorithm with neighborhood constraints is incorporated for getting improved prediction results.

7.2 Discussion

7.2.1 Fuzzy Decision Tree based system

In this research project, one of our main contributions is the enhancement of data mining algorithms in financial application. Three powerful classification decision trees

are addressed for the classification of stock market index. On a nonleaf node, fuzzy ID3 aims (Umanol *et al*., 1994) on average to select an attribute such that the components of the frequency vector are as close to zero or one as possible, whereas Yuan and Shaw's (1995) heuristic aims on average to select an attribute such that only one component is as close to one as possible and other components are as close to zero as possible. Propositions 2 and 3 in section 2.4.2 (chapter 2) indicate that there is a significant difference between fuzzy ID3 and Yuan and Shaw's method when training examples have much classification uncertainty (ambiguity). Moreover, propositions 1 and 2 show that these two heuristics are gradually consistent when classification ambiguity is small, which gives us such guidelines that ID3 heuristic is better if more than one fuzzy attribute in the consequent part of the generated fuzzy rule is acceptable (e.g., IF A THEN B or C), and Yuan and Shaw's heuristic is better if it is unacceptable. Similarly, Yeung *et al*. (1999) and Wang *et al*. (2001) method selects an attribute which has the most influence on the classification as the expanded attribute, and on average it can arrange the degree of importance of attributes in descending order.

In this project, we extend the Yeung *et al*. (1999) and Wang *et al*. (2001) method and present the performance of predictive FDT with regard to the comprehensibility and complexity. These parameters can be measured by examining the small number of internal and leaf nodes. The main strength of predictive FDT is that the algorithm can generate the weighted fuzzy production rules. A weighted fuzzy production rule means that for each proposition of antecedent of the rule a weight is assigned to indicate the degree of importance of the proposition.

The weight assignment is an important but difficult problem. Usually the weight values are given by domain specialists in terms of their experience. Due to the domain experience, it is difficult to measure the weight values. Yeung *et al*. (1999) and Wang *et al*. (2001) makes an initial attempt to measure the weight by a type of degree of importance. Noting that domain experts usually consider a proposition to be important only if the loss of this proposition will result in many errors, the importance of a proposition specified by domain experts is coherent with the importance of an attribute

value given in definition 1 section 4.6.4. Predictive FDT gives a new way to calculate the weights while generating the predictive fuzzy decision tree (fuzzy rules). That is, for a linguistic term of an expanded attribute, its degree of important (definition 1 section 4.6.4) is regarded as the weight of the corresponding proposition. In this way, while converting the tree into a set of rules, a number of weighted fuzzy rules are derived. As compared with traditional fuzzy rules, weighted fuzzy rules have at least two advantages. One is that weighted fuzzy rules can enhance the representation power of fuzzy rules (Yeung and Tsang, 1997) and the other is that weighted fuzzy rules can improve the learning accuracy due to the use of weights as well as the operator pair (addition, multiplication).

The performance of predictive FDT can be measured by the number of WFPRs, and the number of internal and leaf nodes. The experimental results show that the predictive FDT provides promising performance with regard to comprehensibility and complexity. Also, it can be seen from chapter 5 that the significant WFPRs provide good accuracy for the prediction of time series stock market data.

7.2.1 Support Vector Regression based System

Financial time series are inherently noisy, nonstationary and deterministically chaotic (Deboeck, 1994; Yaser, 1996; Huang, et al., 2005; Cao & Tay, 2003; Kim, 2003). These characteristics suggest that there is no complete information that could be obtained from the past behaviour of financial markets to fully capture the dependency between the future price and that of the past. In the modeling of financial time series, this will lead to gradual changes in the dependency between the input and output variables. Therefore, the learning algorithm used should take into account this characteristic. Usually, the information provided by the recent data points is given more weight than that provided by the distant data points (Cao & Tay, 2003; Freitas *et al.* 1999; Refenes *et al.* 1997), because the recent data points in the financial time series could provide more important information than the distant data points.

Recently, SVM have provided a novel approach to improve the generalization property of neural networks. Unlike most of the traditional learning machines that adopt the Empirical Risk Minimization Principle, SVM implements the Structural Risk Minimization Principle, which seeks to minimize an upper bound of the generalization error rather than minimize the training error. This results in better generalization than that with conventional techniques.

In modeling the financial time series, one key problem is its high noise, or the effect of some data points, called outliers, which differ greatly from others (Cao & Tay, 2003, Yang *et al*., 2004; Huang *et al*., 2005). Learning observations with outliers without awareness may lead to fitting those unwanted data and may corrupt the approximation function. This may result in the loss of generalization performance in the test phase. Hence, handling the noise, and detecting and removing the outliers are very important. Because, in the financial time series, the recent data points could provide more important information than the distant data points, before applying the SVR method, we preprocess the data using a weighted kernel-based clustering algorithm incorporating neighborhood constraints, wherein we assign weights to the data points accordingly. Our algorithm has the mechanism to handle noise and outliers in the data.

In the field of financial forecasting, technical analysis, also known as charting, is used widely in real life. It is based on some technical indicators. These indicators help the investors to buy or sell the underlying stock. These indicators are computed by using stock price and volume overtime, and are vital for better prediction of stock index movement (Huang, et al., 2005; Cao & Tay, 2003; Kim, 2003). We also achieved enhancement in prediction performance by considering the technical indicators, like volatility, relative strength index, moving average convergence divergence, relative difference in percentage, etc. We experimented with the real stock market data and got better prediction performance with SVR-based method than that with backpropagation neural networks.

7.3 Conclusions

In this project, we developed a novel classification data mining approach to predict future trends in financial markets. This classification method consists of two major parts: the first is predictive fuzzy decision tree (predictive FDT); and the second is fuzzy reasoning method. Predictive fuzzy decision tree and fuzzy reasoning method are developed using the combination of data mining and AI techniques, i.e., decision tree classification, fuzzy reasoning, clustering, and rules mining.

Classification data mining approach is used to predict future trends in financial markets. The trader's expectations in stock markets are seriously affected by some uncertainty factors like political situation, oil price, overall world situation, local stock markets, etc. Therefore, in this research these factors are carefully calculated and analyzed on the basis of daily stock market information. The results are obtained over three data sets including KLSE, NYSE and LSE by using proposed predictive FDT algorithm. The different sets of samples are tested by using per day information of 100 different companies from January 1, 1999 to December 30, 2004 for KLSE, NYSE and LSE to build predictive FDT. The performance of predictive FDT can also be compared with other existing methods by using different parameters such as learning accuracy, comprehensibility, and complexity. The results show that the predictive FDT has promising performance of comprehensibility (number of rules), complexity (number of nodes i.e., internal and leaf nodes), and rules with better predictive accuracy. Also, the similarity-based fuzzy predictive reasoning method can mine rules from every quarter among the large number of rules that have better predictive accuracy for stock market prediction.

In addition to the fuzzy decision tree based system, a stock market prediction system using support vector machine is also developed. At the preprocessing stage of the SVR-based prediction system, weighted kernel based clustering algorithm with neighborhood constraints is incorporated for getting improved prediction results. In

experimentation, our SVR-based system gave better results than backpropagation neural networks. So, looking at the forecasting results presented by the two methods, i.e. FDT & SVR methods, the trader can confidently decide whether to buy a stock.

The support vector machine for regression is a robust technique for function approximation. The study has concluded that SVM provides a promising alternative to time series forecasting. The SVM offers the following advantages: 1) There is a smaller number of free parameters. C and ε are the only two free parameters of SVM if the kernel function has been considered; 2) SVM forecasts better, as SVM provides a smaller MSE. This is because SVM adopts the Structural Risk Minimization Principle, eventually leading to better generalization than conventional techniques; 3) Training SVM is faster. The regression function in SVM is only determined by the support vectors, and the number of support vectors is much smaller compared to the number of training samples.

In short, looking at the forecasting results presented by the system, the trader can confidently decide whether to buy a stock. Hence, the intelligent knowledge discovery system developed in this project will facilitate economic activity in the country by assisting the companies and individuals in the process of investing in the financial markets. And, it will prove to be in line with the national thrust of prosperous Malaysia.

BIBLIOGRAPHY

Abecasis S.M., Lapenta, S.L. (1997). Modeling the MERVAL index with neural networks and the discrete wavelet transform. *Journal of Computational Intelligence in Finance*, vol. 5: pp. 15–19.

Abecasis, S.M., Lapenta, E.S. (1996). Modeling multivariate time series with a neural network: comparison with regression analysis. *Proc. of the IX Int. Symposium in Informatic Applications*, Antofagasta, Chile, pp. 18–22.

Abu-Mostafa, Y.S., Atiya, A.F. (1996). Introduction to financial forecasting. *Applied Intelligence*, vol. 6, pp. 205–13.

Achelis, S.B. (1995). *Technical Analysis from A to Z.* Probus Publishing, Chicago.

Agarwal, R., Aggarwal, C., and Prasad, V.V. (2001). A tree projection algorithm for generation of frequent itemsets. *Journal of Parallel and Distributed Computing* (Special Issue on High Performance Data Mining), 61(3): 350–371.

Aggarwal, R. and Demaskey, A. (1997). Using derivates in major currencies for cross-hedging currency risks in Asian emerging markets. *Journal of Futures Markets*, vol. 17, pp. 781–796.

Agrawal, R. and Srikant, R. (1994). Fast algorithms for mining association rules in large databases. *Proc. Int. Conf. Very Large Data Bases,* September 1994, pp. 487–499, Santiago, Chile.

Agrawal, R., Lin, K.-I., Sawhney, H.S., and Shim, K. (1995a). Fast similarity search in the presence of noise, scaling, and translation in time-series databases. *Proc. 21st Int. Conf. Very Large Data Bases* (VLDB '95), Sept. 1995, pp. 490–501.

Agrawal, R., Psaila, G., Wimmers, E.L., and Zait, M. (1995b). Querying shapes of histories. *Proc. 21st Int. Conf. Very Large Data Bases* (VLDB '95), Sept. 1995, pp. 502–514.

Ahmed, M.N., Yamany, S.M., Mohamed, N., Farag, A.A., and Moriarty, T. (2002). A modified fuzzy c-means algorithm for bias field estimation and segmentation of MRI data. *IEEE Trans. Medical Imaging*, vol. 21, pp.193–199, 2002.

Apergis, N. and Eleptheriou, S. (2001). Stock returns and volatility: Evidence from the Athens stock market index. *Journal of Economics and Finance*, 25(1), pp. 50–61.

Asim, M., Malik, M. U., Mohsin, F. and Muttee, A. (2005). Stock market prediction software using recurrent neural networks. Department of Computer Science Lahore University of Management Sciences. Lahore.

Aussem, A., Campbell, J., Murtagh,. F. (1998). Wavelet-based feature extraction and decomposition strategies for financial forecasting. *Journal of Computational Intelligence in Finance*, 6: 5–12.

Awan, A.M. and Sap, M.N.M. (2006a). An intelligent system based on kernel methods for crop yield prediction. As a chapter in *Advances in Knowledge Discovery and Data Mining*, Proc. PAKDD'06, W.K. Ng, M. Kitsuregawa, J. Li (Eds.), Lecture Notes in Computer Science (LNCS), vol. 3918, pp. 841–846, Springer.

Awan, A.M. and Sap, M.N.M. (2006b). A framework for predicting oil-palm yield from climate data. *Int. Journal of Computational Intelligence*, vol. 3, no. 2, pp. 111–118, Apr. 2006.

Awan, A.M., Sap, M.N.M., and Mansur, M.O. (2006). Weighted kernel k-means for clustering spatial data. *WSEAS Trans. on Systems,* vol. 5, issue 6, pp. 1301–1308.

Ayob, M., Nasrudin, M.F., Omar, K., Surip, M. (2001). The effect of return function on individual stock price (KLSE) prediction model using neural networks. *Proc. of the Int. Conf. on Artificial Intelligence,* IC-AI'2001, Las Vegas, USA, pp 409–415 , Jun 2001.

Baillie, R.T. and Bollerslev, T. (1992). Prediction in dynamic models with time-dependent conditional variances. *Journal of Econometrics*, 52 (1992), pp. 91–113.

Balkin, S.D. and Ord, J.K. (2000). Automatic neural network modelling for Univariate time series. *Int. Journal of Forecasting*, 16 (2000), pp. 509–515.

Bao, R. C. and Shiou, F. T. (2002). Forecasting short-term stock price indexes-an integrated predictor vs. neural network predictor. *Proc. of IEEE TENCON'02*, Cheng Shiu Institute of Technology.

Bao, Y., Lu, Y., and Zhang, J. (2004). Forecasting stock price by SVM regression. In C. Bussler and D. Fensel (Eds): AIMSA 2004. Lecture Notes Artificial Intelligence. Berlin: Springer-Verlag. 295–303.

Ben-Hur, A., Horn, D., Siegelman, H., and Vapnik, V. (2001). Support vector clustering. *Journal of Machine Learning Research*, vol. 2(Dec.), pp. 125–137.

Bing, L. and Wynne, H. (1999). Mining association rule with multiple minimum support. *Proc. of Int. Conf. on Knowledge Discovery and Data Mining* (KDD-99), August 15-18.

Boes, D.C. and Salas, J.D. (1978). Nonstationarity of the mean and the Hurst phenomenon. *Water Resources Research*, 14 (1978), pp. 135–143.

Boser, B. E., Guyon, I. M. and Vapnik, V. (1992). A training algorithm for optimal margin classifiers. In D. Haussler, editor, *Proc. of the 5th Annual ACM Workshop on Computational Learning Theory*, Pittsburgh, PA, ACM Press, pp. 144–152.

Bowerman, B.L. and O'Connell, R. T. (1979). *Time Series and Forecasting: An Applied Approach*. Belmont, California: Duxbury Press.

Box, G. and Jenkins, G. (1976). *Time Series Analysis*: *Forecasting and Control*. Holden Day, San Francisco, USA.

Box, G. E. P., Jenkins, G. M., Reinsel, G.C. (1994). *Time Series Analysis, Forecasting and Control*. Ed. Englewood Cliffs: Prentice Hall, 3 edition.

Breidt, F.J. and Nan-Jung, H. (2002). A class of nearly long-memory time series models. *Int. Journal of Forecasting,* vol. 18, issue 2, April-June 2002, pp. 265–281.

Buchanan, B.G. and Shortliffe, E.H. (1984). *Rule-Based Expert Systems*, The MYCIN Experiments of the Stanford Heuristic Programming Projects (Addison-Wesley, Reading, MA, 1984).

Buchananan, W.K., Hodges, P. and Theis, J. (2001). Which way the natural gas price: an attempt to predict the direction of natural gas spot price moments using trader positions. *Energy Economics*, May 2001, vol. 23, no. 3, pp. 279–293.

Burges, C. (1998). A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167.

Burges, C. J. C. (1998). A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167.

Camastra, F. and A. Verri (2005). A novel kernel method for clustering. *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 27, pp. 801–805, May 2005.

Cao, L. J. and Francis, E. H. T. (2003). Support vector machine with adaptive parameters in financial time series forecasting. *IEEE Trans. on Neural Networks*, vol. 14, no. 6.

Cao, L.J. (2003b). Support vector machines experts for time series forecasting. *Neurocompting*, 51:321–339, 2003.

Cao, L.J. and Tay, F.E.H. (2003). Support vector machine with adaptive parameters in financial time series forecasting. *IEEE Trans. on Neural Networks,* vol. 14, No. 6, pp. 1506–1518, Nov 2003.

Chang, CC and Lin, CJ (2001). LIBSVM : a library for support vector machines, 2001. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

Chang, R.L.P. and Pavlidis, T. (1977). Fuzzy decision tree algorithms. *IEEE Trans. on Systems, Man and Cybernetics*, vol.7, pp. 28–35.

Chen, B. J., Chang, M. W. and Lin, C. J. (2004). Load forecasting using support vector machines: a study on EUNITE Competition 2001. *IEEE Trans. on Power Systems*, vol. 19(4): 1821–1830.

Chen, S.M. (1994). A weighted fuzzy reasoning algorithm for medical diagnosis. *Decision Support Syst*., 1994, vol.11, pp. 37–43.

Chen, S.M. (1998). A new approach to handling fuzzy decision-making problems. *IEEE Trans. on Systems, Man and Cybernetics*, Nov/Dec, 1998, vol, 18, pp. 1012–1016.

Cheng, B. and Titterington, D.M. (1994). Neural networks: A review from a statistical perspective. *Statistical Science* 9 1, pp. 2–54

Cheng, W., Wanger, L., Lin, C.H. (1996). Forecasting the 30-year US treasury bond with a system of neural networks. *Journal of Computational Intelligence in Finance*, 4: 10–16

Cherkassky, V. and Mulier, F. (1998). *Learning from data - concepts, theory and methods.* John Wiley & Sons, New York.

Chinn, M.D., LeBlanc, M., and Coibion, O. (2001). The predictive characteristics of energy futures: recent evidence for crude oil, natural gas, gasoline and heating oil. (October 2001). *UCSC Economics Working Paper* No. 490. Available at: http://ssrn.com/abstract=288844 or DOI: 10.2139/ssrn.288844

Chuang, C.-C., Su, S.-F., Jeng, J.-T., and Hsiao, C.-C. (2002). Robust support vector regression networks for function approximation with outliers. *IEEE Trans. on Neural Networks*, vol. 13, pp. 1322 –1330, 2002.

Darbellay, G.A. and Slama, M. (2000). Forecasting the short-term demand for electricity? Do neural networks stand a better chance?. *Int. Journal of Forecasting,* 16 (2000), pp. 71–83.

Das, G., Lin, K., Mannila, H., Renganathan, G., and Smyth, P. (1998). Rule discovery from time series. *Pro. of the 4th Int. Conf. on Knowledge Discovery and Data Mining.* New York, NY, Aug 27-31, pp. 16–22.

Deboeck, G.J. (1994). *Trading on the Edge: Neural, Genetic, and Fuzzy Systems for Chaotic Financial Markets*. New York, Wiley, 1994.

Dhillon, I.S., Fan, J, and Guan, Y. (2001). Efficient clustering of very large document collections. As a chapter in *Data Mining for Scientific and Engineering Applications*, pp.357–381. Kluwer Academic Publishers, 2001.

Dhillon, I.S., Guan, Y., and Kulis, B. (2004). Kernel kmeans, spectral clustering and normalized cuts. *Proc. 10th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining,* pp. 551–556, 2004.

Ding, L., Shen, Z., and Mukaidono, M. (1989). A new method for approximate reasoning. *Proc. 19th Int. Symp. Multiple-Valued Logic,* 1989, pp. 179–185.

Ding, L., Shen, Z., and Mukaidono, M., (1992). Revision principle for approximate reasoning-based on linear revising method. *Proc. 2nd Int. Conf. Fuzzy Logic and Neural Networks*, 1992, pp. 305–308.

Dong, M. and Kothari, R. (2001). Look-ahead based fuzzy decision tree induction. *IEEE Trans. on Fuzzy Systems*, vol. 9(3), June 2001, pp. 461–468.

Dorsey R, Sexton R (1998). The use of parsimonious neural networks for forecasting financial time series. *Journal of Computational Intelligence in Finance*, 6: 24–30

Dwinnell, W. (2002). Data visualization tips for data mining: pattern recognition provides data insight. *PC AI Mag.,* 2002, vol. 16(1), pp. 51–57.

Elkan, C. (2003). Using the triangle inequality to accelerate k-means. *Proc. 20th Int. Conf. Machine Learning*.

Elovici, Y. and Braha, D. (2003). A decision-theoretic approach to data mining. *IEEE Trans. on Systems, Man and Cybernetics*, Part A, vol. 33, issue 1, Jan. 2003, pp. 42–51

Engle, R.F. (1982). Autoregressive conditional heteroskedasticity with estimates of the variance of United Kingdom inflation, *Econometrica* 50 (1982), pp. 987–1007.

Engle, R.F. Lilien, D.M., and Robins, R.P. (1987). Estimating time varying risk premia in the term structure: the ARCH-M model, *Econometrica* 55 (1987), pp. 391–407.

Evgeniou, T., Pontil, M. (2001). Support Vector Machines: Theory and Applications. *Machine Learning and Its Applications*, pp. 249–257.

Fayyad, U., Piatetsky-Shapiro, G., and Smyth, P. (1996). From data mining to knowledge discovery in databases. *AI Magazine*. American Association for Artificial Intelligence, pp. 37–54.

Fayyad, U.M., Piatetsky-Shapiro, G., and Smyth, P. (1995). From data mining to knowledge discovery. In: U. Fayyad., G. Piatetsky-Shapiro., P. Smyth., eds., *From Data Mining to Knowledge Discovery*, AAAI Press/MIT Press.

Fayyad, U.M., Piatetsky-Shapiro, G., Smyth, P. (1996a). From the data mining to knowledge discovery in databases. *AI Magazine*, 1996 vol. 17, pp. 37–54.

Fayyad, U.M., Piatetsky-Shapiro, G., Smyth, P., and Uthurusamy, Eds. (1996b). *Advances in Knowledge Discovery and Data Mining*. Menlo Park, CA: AAAI/MIT Press, 1996.

Ferreira, T.A.E., Vasconcelos, G.C., and Adeodato, P.J.L. (2004). A hybrid intelligent system approach for improving the prediction of real world time series. *Congress on Evolutionary Computation* 2004 (CEC2004), 19-23 June 2004, vol. 1, pp. 736–743.

Fosso, O.B., Gjelsvik, A., Haugstad, A., Mo, B., and Wangensteen, I. (1999). Generation scheduling in a deregulated system: the Norwegian case. *IEEE Trans. on Power Systems,* Feb. 1999, vol. 14, issue 1, pp. 75–81

Freitas, N.D., M. Milo, and P. Clarkson (1999). Sequential support vector machine. *Proc. 1999 IEEE Signal Processing Society Workshop*, 1999, pp. 31–40.

Fritz, M. (1940). *The Stock Market, Credit and Capital Formation*. University of Buffalo. London.

George, H.J. (1997). Enhancements of the data mining process. PhD dissertation, the department of computer science, Stanford University.

Giovanni, C. (2003). A comparison of stock market mechanisms. Department of Science Economic. University of Degli Study Di Salerno. Fisciano.

Girolami, M. (2002). Mercer kernel based clustering in feature space. *IEEE Trans. Neural Networks*, vol. 13(3), pp. 780–784, May 2002.

Gorr, W. (1994). Research prospective on neural network forecasting. *Int. Journal of Forecasting,* 10 (1994), pp. 1–4.

Gray, R.M. (1992), *Vector Quantization and Signal Compression*. Kluwer Academic Press, Dordrecht.

Gross, G. and Galiana, F.D. (1987). Short-term load forecasting. *Proceedings of the IEEE*, December 1987, vol. 75, no. 12, pp. 1558–1573.

Hagan, M.T. and Behr, S.M. (1987). The time series approach to short term load forecasting. *IEEE Trans. Power Syst*., Aug. 1987, vol. 2, pp. 785–791.

Hall, J.W. (1994). Adaptive selection of US stocks with neural nets. In: Deboeck GJ, editor. *Trading on the edge: neural, genetic, and fuzzy systems for chaotic 1nancial markets*. New York: Wiley; 1994, pp. 45–65.

Han, J. and Kamber, M. (2001). *Data Mining: Concept and Techniques*. San Francisco, CA: Morgan Kaufmann, 2001.

Han, J., Gong, W., and Yin, Y. (1999). Efficient mining of partial periodic patterns in time series databases. *Proc. of Int. Conf. on Data Engineering* (ICDE99), Sydney, Australia, March.

Han, J., Pei, J., and Yin, Y. (2000). Mining frequent patterns without candidate generation. *Proc. of the 19th ACM SIGMOD Int. Conf. on Management of data*, pp. 1–12.

Hiemstra, Y. (1994). A stock market forecasting support system based on fuzzy logic. *Proc. of the 27th Annual Hawaii International Conference on System Sciences*, vol 3, pp. 281–288.

Higashi, M. and Klir, G.J. (1993). Measure on uncertainty and information based on possibility distribution. *Int. Journal of General Systems*, vol. 9, pp. 43–58, 1993.

Hill, T., Marquez, L., O'Conner, M., and Remus, W. (1994). Artificial neural network models for neural network forecasting of quarterly accounting earnings. *Int. Journal of Forecasting,* 10 (1994), pp. 5–15.

Hippert, H.S., Pedreire, C.E., and Souza, R.C. (2001). Neural networks for short-term load forecasting: A review and evaluation. *IEEE Trans. Power Systems*, Feb.2001. vol. 16, pp. 44–55.

Hobson, D.G. (1998). Stochastic volatility. In: *Statistics in Finance,* D. Hand and S. Jacka, eds., *Applications of Statistics Series*, Arnold, London (1998).

Hossein, A. (1996). Time series analysis and forecasting techniques. http://obelia.jde.aca.mmu.ac.uk/resdesgn/arsham/opre330Forecast.htm

Huang, H.-J., and Hsu, C.N. (2002). Bayesian classification for data from the same unknown class. Part B, *IEEE Trans. on Systems, Man and Cybernetics,* April 2002, vol. 32 , issue 2, pp. 137–145.

Huang, W., Y. Nakamori and S.Y. Wang (2005). Forecasting stock market movement direction with support vector machine. *Computers and Operations Research,* vol. 32(2005), pp. 2513–2522.

Hurst, H.E. (1957). A suggested statistical model of some time series which occur in nature. *Nature*, 180 (1957), pp. 494.

Ince, H. and T.B. Trafalis (2004). Kernel principal component analysis and support vector machines for stock price prediction. *Proc. IEEE Int. Conf. Neural Networks,* 2004.

Investopedia (2006). Available at: http://www.investopedia.com/university/

Jan, K., Jirí, K. and Martin, V. (2003). Predictive system for multivariate time series. Department of Cybernetics, Applied Research Department, Czech Republic.

Jang, G.S., Lai, F., Jiang, B. W., and Chien, L. H. (1991). An intelligent trend prediction and reversal recognition system using dual-module neural networks. *Proc. First Int. Conf. on Artificial Intelligence Applications on Wall Street*, New York, U.S.A., October 1991. pp. 42–51.

Jing, T. Y. and Hean, L. P. (1997). Forecasting the KLSE index using neural network. National University of Singapore, Singapore.

Johan, A.K.S. (2001). Nonlinear modeling and support vector machines. Department of Electrical Engineering ESAT-SISTA, Belgium.

John, G.H. and Langley, P. (1996). Static versus dynamic sampling for data mining. In: E. Simoudis and J. Han & U. Fayyad, eds., *Proc. 2nd Int. Conf. on Knowledge Discovery and Data Mining*, AAAI Press, Menlo Park, CA, pp. 367–370

Junaida, S. (2005). Experimenting the applicability of support vector machine regression in rainfall prediction. Master Thesis (Computer Science), University of Technology Malaysia.

Kamruzzaman, J., Sarker, R. A. and Ahmad, I. (2003). SVM based models for predicting foreign currency exchange rates. *Proc. of the third IEEE Int. Conf. on Data Mining* (ICDM'03).

Ke, W. and Yu, H., (2000). Mining frequent item sets using support constraint. *Proc. of the 26th VLDB Conf.*, Cairo, Egypt, 2000.

Keogh, E. and Pazzani, M. (1998). An enhanced representation of time series which allows fast and accurate classification, clustering and relevance feedback. *Proc.*

of the 4th Int. Conf. on Knowledge Discovery and Data Mining New York, NY, Aug 27-31, pp. 239–241.

Kim, K.J. (2003). Financial time series forecasting using support vector machines. *Neurocomputing*, vol. 55(2003), pp. 307–319.

Kim, T. W. and Valdes, J. B. (2003). Nonlinear model for drought forecasting based on a conjunction of wavelet transforms and neural networks. *Journal of Hydrologic Engineering*, 8:6 (319–328).

Kimoto, T., Asakawa, K., Yoda, M., Takeoka, M. (1993). Stock market prediction system with modular neural networks. *Neural Networks in Finance and Investing*, 1993; 343–357

Kleme, V. (1974). The Hurst phenomenon: a puzzle?. *Water Resources Research*, 10 (1974), pp. 675–688.

Klemettinen, M., Mannila, H., Ronkainen, P., Toivonen, H., and Verkamo, A.I. (1994). Finding interesting rules from large sets of discovered association rules. *Proc. 3rd Int. Conf. Information and Knowledge Management*, Nov. 1994, pp. 401–408, Gaithersburg, Maryland.

Klir, G.J. and Mariano, M. (1987). On the uniqueness of possibilistic measure of uncertainty and information. *Fuzzy Sets and Systems*, 1987. vol. 24(2), pp. 197–219.

Koenker, R. and Zhao, Q. (1996). Conditional quantile estimation and inference for ARCH models, *Econometric Theory*, 12 (1996), pp. 793–813.

Kohonen, T. (1988). *Self-Organization and Association Memory*. Springer, 1988. Berlin.

Kong, W., Tham, L., Wong, K.Y., Tan, P. (2004). Support vector machine approach for cancer detection using amplified fragment length polymorphism (AFLP) screening method. *Proc. of the second conf. on Asia-Pacific bioinformatics*, vol. 29, pp. 63–66, Dunedin, New Zealand.

Kosko, B. (1992). *Neural Systems and Fuzzy Systems: A Dynamical Systems Approach to Machine Intelligence*. Englewood Cliffs, NJ: Prentice-Hall, 1992.

Langley, P and Simon, H.A. (1995). Applications of machine learning and rule induction. *Communications of the ACM*, 38, November, pp. 55–64

Last, M. and Maimon, O. (2004). A compact and accurate model for classification. *IEEE Trans. on Knowledge and Data Engineering*, vol. 16, issue 2, Feb. 2004, pp. 203– 215.

Lin, D.-I. and Kedem, Z.M. (2002) Pincer-search: an efficient algorithm for discovering the maximum frequent set. *IEEE Trans. on Knowledge and Data Engineering,* vol. 14, issue 3, May-June 2002, pp. 553–566.

Lin, H. T. and Lin, C. J. (2003). A study on sigmoid kernels for support vector machine and the training of non-PSD kernels by SMO-type methods. [Online] http://www.csie.ntu.edu.tw/~cjlin/papers/tanh.pdf

Liu, L.-M, Bhattacharyya, S., Sclove, S., Chen, R., Lattyak, W. (2001). Data mining on time series: an illustration using fast-food restaurant franchise data. *Computational Statistics and Data Analysis*, 37, no.4, pp. 455–476.

Liu, M. (2000). Modeling long memory in stock market volatility. *Journal of Econometrics,* 99 (2000), pp. 139–171.

Lloyd, S.P. (1982). An algorithm for vector quantizer design. *IEEE Trans. Communications*, vol. 28, no. 1, pp. 84–95, 1982.

Lu, W. Z., Wang, W. J., Leung, A. Y. T., Lo, S. M., Yuen, R. K. K., Xu, Z. and Fan, H.Y. (2002). Air pollutant parameter forecasting using support vector machines. *Proc. of the 2002 Int. Joint Conf. on Neural Networks*, IJCNN '02, May 12-17, 2002. IEEE, vol. 1, pp. 630–635.

Ma, X. Pang, H., and Tan, K-L. (2004). Finding constrained frequent episodes using minimal occurrences. *Proc. 4th IEEE Int. Conf. on Data Mining,* 2004 (ICDM 2004), Nov. 2004, pp. 471–474 .

MacQueen, J.B. (1967). Some Methods for classification and Analysis of Multivariate Observations. *Proc. of 5th Berkeley Symposium on Mathematical Statistics and Probability*, Berkeley, University of California Press, 1:281–297

Mark, T., Leung, H.D., and An-Sing, C. (2000). Forecasting stock indices: a comparison of classification and level estimation models. *Int. Journal of Forecasting,* vol. 16, issue 2, April-June 2000, pp. 173–190

Mitra, S., Pal, S.K., and Mitra, P. (2002). Data mining in soft computing framework: a survey Neural Networks. *IEEE Trans. on Neural Networks*, vol. 13 , issue 1, Jan. 2002, pp. 3–14

Morano, C. (2001). A semiparametric approach to short-term oil price forecasting. *Energy Economics*, May 2001. vol. 23, no. 3, pp. 325–338.

Mukaidono, M., Ding, L., and Shen, Z. (1990). Approximate reasoning based on revision principle. *Proc. NAFIPS'90*, vol. 1, 1990, pp. 94–97.

Mukherjee, S., Osuna, E., and Girosi, F. (1997). Nonlinear prediction of chaotic time series using support vector machines. In: J. Principe, L. Giles, N. Morgan, and E. Wilson, eds., *IEEE Workshop on Neural Networks for Signal Processing VII*, pp. 511–519. IEEE Press, 1997.

Muller KR, Smola AJ, Ratch G, Scholkopf B, Kohlmorgen J, and Vapnik V (1998). Using support vector support machines for time series prediction. In: B. Schölkopf, C. Burges, and A. Smola, editors, Advances in Kernel Methods - Support Vector Learning, *Proc. of the NIPS Workshop on Support Vectors*. The MIT Press, Cambridge, MA, 1998.

Muller, K.R., Smola, A., Ratsch, G., Scholkopf, B., Kohlmorgen, J., and Vapnik, V.N. (1997). Predicting time series with support vector machines. *Proc. ICANN,* pp. 999–1004, 1997.

Nicolaisen, J.D., Richter, C.W.Jr., and Sheble, G.B. (2000). Price signal analysis for competitive electric generation companies. *Poc. Conf. Elect. Utility Deregulation*

and Restructuring and Power Technologies, London, U.K., Apr. 4-7, 2000, pp. 66–71.

Nogales, F.J., Contreras, J., Conejo, A.J., and Espinola, R. (2002). Forecasting next-day electricity prices by time series models. *IEEE Trans. Power Systems*, vol. 17, pp. 342–348, May 2002.

NYSE Net: About the NYSE, Capt. About the Exchange. Available at: http://www.nyse.com/about/1088808971270.html

O'Connell, P.E. (1971). A simple stochastic modeling of Hurst's law, *Proc. of Int. Symposium on Mathematical Models in Hydrology*, Warsaw, pp. 327–358.

O'Connor, M., Remus, W., and Griggs, K. (1997). Going up-going down: How good are people at forecasting trends and changes in trends? *Journal of Forecasting*, vol. 16, 165–176.

Okamoto, T., Ueda, Y. and Kunishige, M. (1995). The distributed multimedia learning environment employing gaming/simulation method with expert system in the world of macro economics. *Computer and Artificial Intelligence*, vol. 14, no.4, pp. 395–415.

Osuna, E., Freund, R. and Girosi, F. (1997). Support vector machines: training and applications. *Artificial Intelligence Memo* 1602, MIT.

Osuna, E., Freund, R., and Girosi, F. (1997). Support vector machines: training and applications. *ICANN 2001*, LNCS 2130, pp. 405–415.

Ozaki, T. (1985). Nonlinear time series models and dynamical systems. In: E. J. Hannan and P. R. Krishnaiah, eds., *Hand Books of Statistics,* vol. 5, North Holland, Amsterdam.

Pan, Z.H., Wang X.D. (1998). Wavelet-based density estimator model for forecasting. *Journal of Computational Intelligence in Finance*, vol. 6, pp. 6–13

Parke, W.R. (1999). What is fractional integration?. *Review of Economics and Statistics* 81 (1999), pp. 632–638.

Pasquier, N., Bastide, Y., Taouil, R., Lakhal, L. (1999). Discovering frequent closed itemsets for association rules. In: Beeri, C., Buneman, P., eds., *Proc. 7th Int. Conf. on Database Theory*, Jerusalem, Israel, January 10-12, 1999, vol. 1540 of Lecture Notes in Computer Science. Springer, pp. 398–416

Pawlak, Z. (1991) *Rough Sets, Theoretical Aspects of Reasoning about data*. Dordrecht, Kluwer, 1991.

Pellizzari, P. Pizzi, C. (1997). Fuzzy weighted local approximation for financial time seriesmodelling and forecasting. In: *Computational Intelligence for Financial Engineering* (CIFEr), Proc. of the IEEE/IAFE 1997, pp. 137–143.

Pierre, G. and Sébastien, L. (2004). Modelling daily value-at-risk using realized volatility and ARCH type models. *Journal of Empirical Finance,* vol. 11, issue 3, June 2004, pp. 379–398.

Potter, K.W. (1975). Comment on "The Hurst phenomenon - a puzzle?", by V. Kleme. *Water Resources Research* 11 (1975), pp. 373–374.

Priestley, M.B. (1998). N*on-Linear and Non-Stationary Time Series Analysis*. academic press, London (1988).

Quinlan, J.R. (1986). Induction of decision tree. *Machine Learning*, vol. 1(1), pp.81–106.

Quinlan, J.R. (1993). *C4.5: Programs For Machine Learning*. Morgan Kaufmann.

Rahayu binti Hamid (2004). Bioactivity prediction and compound classification using neural network and support vector machine: a comparison. Master Thesis (Computer Science), University of Technology Malaysia.

Raman, L. (1997). Using neural networks to forecast stock market prices. University of Manitoba: Course project.

Ramirez, L.; Pedrycz, W.; Pizzi, N (2001). Severe storm cell classification using support vector machines and radial basis function approaches. *Proc. Canadian Conf. on Electrical and Computer Engineering*, pp. 87–91.

Raymond, S.T. Lee. (2004). iJADE stock advisor: an intelligent agent based stock prediction system using hybrid RBF recurrent network. *IEEE Trans. on Systems, Man and Cybernetics* (Part A), vol. 34(3), pp. 421–428.

Refenes, A.N., Y. Bentz, D.W. Bunn, A.N. Burgess, and A.D. Zapranis (1997). Financial time series modeling with discounted least squares back-propagation. *Neurocomputing*, vol. 14, pp. 123–138, 1997.

Refenes, A.N., Zapranis, A.D., and Bentz, Y. (1993). Modeling stock returns with neural networks. *Presented at the Workshop on Neural Network Applications and Tools,* London, U.K., 1993.

Ripley, B.D. (1993). Statistical aspects of neural networks. In: Barndorff-Nielsen, O.E., Jensen, J.L., Kendall, W.S. (Eds.), *Networks and Chaos- Statistical and Probabilistic Aspects*. Chapman and Hall, London, pp. 40–123.

Rumelhart, D.E. and McCleland, J.L. (1987). *Parallel Distributed Processing, Explorations in the Microstructure of Cognition.* Vol. 1 & 2, MIT Press Massachusetts (1987).

Saad, E.W. Prokhorov, D.V. Wunsch, D.C. (1996). Advanced neural network training methods for low false alarm stock trend prediction. Proc. IEEE Int. Conf. on Neural Networks, vol. 4, pp. 2021–2026.

Sansom, D. C., Downs, T., and Saha, T. K. (2002). Evaluation of support vector machine based forecasting tool in electricity price forecasting for Australian national electricity market participants. *Journal Elect. Electron. Eng*. Australia., 22: 227–234.

Sap, M.N. Md. and A.M. Awan (2005). Finding spatio-temporal patterns in climate data using clustering. *Proc. IEEE 2005 Int. Conf. on Cyberworlds*, Singapore, 23-25 Nov. 2005, pp. 155–164.

Sarjon, D. and M.N. Md. Sap (2002a). Mining association rules using rough set and association rules methods. *Proc. of Int. Conf. on Artificial Intelligence in*

Engineering and Technology 2002 (ICAIET' 02), Sabah, Malaysia, June 17-18, 2002.

Sarjon, D. and M.N. Md. Sap (2002b). Mining multiple level association rules using rough set and association rule methods. *Proc. of Int. Conf. on artificial intelligence and soft computing* (ASC' 02), July 17-19, 2002, Banff, Canada.

Savnik, I., Lausen, G., Kahle, H.-P., Spiecker, H., Hein, S. (2000). Algorithm for matching sets of time series. In: *Principles of Data Mining and Knowledge Discovery*, pp. 277–288.

Scholkopf, B. (2000). The kernel trick for distances. In: *Advances in Neural Information Processing Systems* 12, S. A Solla, T. K. Leen, and K.-R. Muller (Eds.). MIT Press, 2000, pp. 301–307.

Sharda, R, Patil, R.B. (1993). A connectionsist approach to time series prediction: an empirical test. *Neural Networks in Finance and Investing*, 1993; 451–464

Sharda, R., (1994). Neural networks for the MS/OR analyst: An application bibliography. *Interfaces* 242, pp. 116–130

Smola, A.J. (1998b). Learning with kernels. Ph.D. dissertation, GMD, Birlinghoven, Germany, 1998.

Smola, A.J. and Scholkopf, B. (1998). A tutorial on support vector regression. Royal Holloway College, London, U.K., NeuroCOLT Tech. Rep. TR, 1998.

Subba, R.T. and Gabr, M.M. (1984). Introduction to bispectral analysis and bilinear time series models. *Lecture Notes in Statistics*, vol. 24, Springer, Berlin.

Suykens, J.A.K., J. De Brabanter, L. Lukas, and J. Vandewalle. Weighted least squares support vector machines: robustness and sparse approximation. *Neurocomputing*, 2001.

Szkuta, B.R., Sanabria, L.A., and Dillon, T.S. (1999). Electricity price short-term forecasting using artificial neural networks. *IEEE Trans. Power Systems*, vol. 14, pp. 851–857, Aug.1999.

T. Raicharoen, C. Lursinsap (2003). Application of critical support vector machine to time series prediction. *Proc. of the IEEE Int. Symposium on Circuits and System* (ISCAS'03), vol.5, pp. 741–744, Bangkok Thailand, 25-28 May 2003

Tang, Y, Pan, W.M., and Li, H.M. (2002). Fuzzy Naive Bayes classifier based on fuzzy clustering. *Proc. IEEE Int. Conf. on Systems, Man and Cybernetics*, 6-9 Oct. 2002, vol. 5, pp. 452–458.

Teh, E. L. (2005). Stock price prediction using kohonen network. Degree Thesis, University of Technology Malaysia.

Tetsuji, T. and Ken'ichi, K. (1992). Stock price pattern matching system. C & C Information technology Research Laboratories, NEC Corporation. IEEE.

Theodore, B. T. and Huseyin, I. (2000). Support vector machine for regression and applications to financial forecasting. University of Oklahoma, Norman, Oklahoma.

Thomason, M. (1999). The practitioner methods and tools. *Journal of Computational Intelligence in Finance*, vol. 7, no. 3, pp. 36–45, 1999.

Thomason, M. (1999b). The practitioner methods and tools. *Journal of Computational Intelligence in Finance*, vol. 7, no. 4, pp. 35–45, 1999.

Tong, S. and Chang, E. (2001) Support vector machine active learning for image retrieval. MM'OI. Sept. 30-Oct. 5, 2001. Ottawa, Canada: ACM, pp. 107–118.

Torben, G.A. and Lund, J. (1997). Estimating continuous-time stochastic volatility models of the short-term interest rate. *Journal of Econometrics,* 77 (1997), pp. 343–378.

Toth, E., Brath, A. and Montanari, A. (2000). Comparison of short-term rainfall prediction models for real time flood forecasting. *Journal of Hydrology*, 239: 132–147.

Trafalis, T.B. and Ince, H. (2002). Benders decomposition technique for support vector regression. *Proc. of the 2002 Int. Joint Conf. on Neural Networks*, 12-17 May 2002, vol. 3, pp. 2767–2772.

Turksen, I. B. and Zhong, Z. (1989). An approximate analogical reasoning approach based on similarity measures. *IEEE Trans. Systems, Man and Cybernetics*, vol. 18, pp. 1049–1056.

Turksen, I. B. and Zhong,. (1990). An approximate analogical reasoning scheme based on similarity measures and interval valued fuzzy sets. *Fuzzy Sets and Systems*, vol. 34, pp. 323–346, 1990.

Umanol, M., Okamoto, H., Hatono, I., Tamura, H., Kawachi, F., Umedzu, S., and Kinoshita, J. (1994). Fuzzy decision trees by fuzzy ID3 algorithm and its application to diagnosis systems Fuzzy Systems. *Proc. of the third IEEE Conf. on Computational Intelligence.* 26-29 June 1994, vol. 3, pp. 2113–2118.

Valenzuela, J. and Mazumdar, M. (2001). On the computation of the probability distribution of the spot market price in a deregulated electricity market. *Proc. the 22nd Int. Conf. on Power Industry Computer Applications*, Sydney, Australia, May 2001, pp. 268–271.

Van, E and Robert, J (1997). *The application of neural networks in the forecasting of share prices*. Finance & Technology Publishing, Haymarket, VA, USA, 1997.

Vanajakshi, L. and Rilett, L.R. (2004). A comparison of the performance of artificial neural network and support vector machines for the prediction of traffic speed. 2004 *IEEE Intelligent Vehicles Symposium*, June 14-17, University of Parma, Italy, pp. 194–199.

Vapnik V.P. (1995) *The Nature of Statistical Learning Theory.* Springer-Verlag New York, Inc., 1995.

Vapnik V.P. (1998). *Statistical Learning Theory*. John Wiley & Sons, 1998.

Vapnik, V.N., S. Golowich, and A. Smola. Support vector method for function approximation, regression estimation and signal processing. In: M. Mozer, M. Jordan, and T. Petshe, eds., *NIPS*, vol. 9, pp. 281–287, Cambridge, MA, 1997. MIT Press.

Wang, A.J. and Ramsay, B. (1998). A neural network based estimator for electricity spot- pricing with particular reference to weekend and public holidays. *Neurocomputing,* vol.23, pp. 47–57, 1998.

Wang, X., Phua, P.K.H., and Weidong, L. (2003). Stock market prediction using neural networks: does trading volume help in short-term prediction. *Proc. of the Int. Joint Conf. on Neural Networks*, vol. 4, July 20-24, 2003, pp. 2438–2442.

Wang, X.Z. and Hong, J. R. (1998). On the handling of fuzziness for continuous-valued attributes in decision tree generation, *Fuzzy Sets and Systems*, vol. 99, pp. 283–290, 1998.

Wang, X.-Z., Yeung, D.S., and Tsang, E.C.C. (2001). A comparative study on heuristic algorithms for generating fuzzy decision trees Systems. *IEEE Trans. on Systems, Man and Cybernetics,* Part B, vol. 31 , issue 2 , April 2001, pp. 215–226.

Weigend, A. (1993). Measuring the effective number of dimensions during backpropagation training. *Proc. of the 1993 Connectionist Models Summer School*, pp. 335–342. Morgan Kaufmann, San Francisco, CA.

Weiss, E. (2000). Forecasting commodity prices using ARIMA. *Technical Analysis of Stocks & Commodities*, vol. 18, no. 1, pp. 18–19, 2000.

Widrow, B., Rumelhart, D.E. and Lehr, M.A. (1994). Neural networks: Applications in industry, business and science. *Communications of the ACM*, 37(3), pp. 93–105.

Wu, C. H., Ho, J. M., and Lee, D. T. (2004). Travel time prediction with support vector regression. *IEEE Trans. on Intelligent Transportation Systems*, vol. 5(4): 276–281.

Wu, Y. and Zhang, H. (1997). Forward premiums as unbiased predictors of future currency depreciation: A non-parametric analysis. *Journal of International Money and Finance*, vol. 16, pp. 609–623.

Xu, Y., Yu, J.X., Liu, G., and Lu, H. (2002). From path tree to frequent patterns: a framework for mining frequent patterns. *Proc. IEEE Int. Conf. on Data Mining,* 2002 (ICDM 2002), Dec. 2002, pp. 514–521.

Yang, H., King, I., and Chan, L. (2002). Non-fixed and asymmetrical margin approach to stock market prediction using Support Vector Regression. *Proc. of the 9th Int. Conf. on Neural Information Processing*, 18-22 Nov. 2002, vol. 3, pp. 1398–1402.

Yang, H., Chan, L. and King, I. (2002). Support vector machine regression for volatile stock market prediction. *Proc. of Intelligent Data Engineering and Automated Learning.* H. Yin, N. Allinson, R. Freeman, J. Keane, and S. Hubbard, Eds. LNCS 2412: Springer, 391–396.

Yang, H., Chan, L., and King, I. (2002). Support Vector Machine Regression for Volatile Stock Market Prediction. *IDEAL 2002*, vol. 2412 of *LNCS*, pp. 391–396. Springer, 2002.

Yang, H., Huang, K., Chan, L., King, I., and Lyu M.R. (2004b). Outliers treatment in support vector regression for financial time series prediction. *ICONIP* 2004, LNCS 3316, pp. 1260–1265, Springer.

Yang, H., King, I., Chan, L., and Huang, K. (2004). Financial time series prediction using non-fixed and asymmetrical margin setting with momentum in support vector regression. *Neural Information Processing: Research and Development*, pp. 334–350. Springer.

Yaser, S.A.M., Atiya, A.F. (1996). Introduction to financial forecasting. *Applied Intelligence*, 1996; 6: 205–213

Yeung, D.S. and Tsang, E.C.C. (1994). Improved fuzzy knowledge representation and rule evaluation using fuzzy Petri nets and degree of subsethood. *Intelligent Systems*, vol. 9, no. 12, pp. 1083–110, 1994.

Yeung, D.S. and Tsang, E.C.C. (1995). A weighted fuzzy production rule evaluation method. Fuzzy Systems, 1995. *Proc. of IEEE Int. Joint Conf. on the Fourth IEEE Int. Conf. on Fuzzy Systems and The Second Int. Fuzzy Engineering Symposium*, 20-24 March 1995, vol. 2, pp. 461–468.

Yeung, D.S. and Tsang, E.C.C. (1997a). Weighted fuzzy production rules. *Fuzzy Sets and Systems*, vol. 88, pp. 229–313, 1997.

Yeung, D.S. and Tsang, E.C.C. (1997b). A comparative study on similarity-based fuzzy reasoning methods. *IEEE Trans. on Systems, Man and Cybernetics,* Part B, vol. 27, Issue: 2 , April 1997, pp. 216–227.

Yeung, D.S. and Tsang, E.C.C. (1998). A multilevel weighted fuzzy reasoning algorithm for expert systems. *IEEE Trans. on Systems, Man and Cybernetics*, vol. 28, pp. 149–158, 1998.

Yeung, D.S., Tsang, E.C.C., and Xizhao, Wang. (2002). Fuzzy rule mining by fuzzy decision tree induction based on fuzzy feature subset. *Proc. IEEE Int. Conf. on Systems, Man and Cybernetics*, 6-9 Oct. 2002, vol. 4.

Yeung, D.S., Wang, X.Z., and Tsang, E.C.C. (1999); Learning weighted fuzzy rules from examples with mixed attributes by fuzzy decision trees Systems. *IEEE Int. Conf. on Systems, Man, and Cybernetics,* vol. 3 , 12-15 Oct. 1999, pp. 349–354

Yimin, X and Dit-Yan, Y. (2004). Time series clustering with ARMA mixtures. *Pattern Recognition*, vol. 37, issue 8, August 2004, pp. 1675–1689.

Yuan, Y. and Shaw, M.J. (1995). Induction of fuzzy decision trees. *Fuzzy Sets and Systems*, vol. 69, 1995, pp. 125–139.

Zadeh, L.A. (1965). Fuzzy sets. *Information and Control,* 8, 1965, pp. 338–353

Zadeh, L.A. (1973). Outline of a new approach to the analysis of complex systems and decision processes. *IEEE Trans. Systems, Man and Cybernetics.* vol. 3, pp. 28–44.

Zadeh, L.A. (1987a). The concept of linguistic variables and its application to approximate reasoning-I, II, III. In: *Fuzzy Sets and Applications*: Selected Papers by L. A. Zadeh, R. R. Yager, et al. Ed. New York: Wiley, 1987.

Zadeh, L.A. (1987b). A theory of commonsense knowledge. In: *Fuzzy Sets and Applications*: Selected Papers by L. A. Zadeh, R. R. Yager, et al. Ed. New York: Wiley, 1987, pp. 615–653.

Zadeh, L.A. (1999). Fuzzy Sets as a basis for a theory of possibilistic. *Fuzzy Sets and Systems*, vol. 100, pp. 9–34, 1999.

Zeidler, J. and Schlosser, M. (1996). Continuous valued attributes in fuzzy decision trees. *Proc. of the 8th Int. Conf on Information Processing and Management of Uncertainty in Knowledge-Based System*, pp. 395–400, 1996.

Zhang, D. and Zhou, L. (2004). Discovering Golden Nuggets: Data Mining in Financial Application. *IEEE Trans. on Systems, Man, and Cybernetics*- Part C*: Applications and Reviews : Accepted for future publication* , vol. PP, issue 99, 2004, pp. 1–10.

Zhang, Y-P., Wu, T., and Zhang, L. (2002). A self-adjusting and probabilistic decision-making classifier based on the constructive covering algorithm in neural networks. *Proc. Int. Conf. on Machine Learning and Cybernetics*, 4-5 Nov. 2002, vol. 4, pp. 2171–2174.

Zhu, J.Y., Ren, B., Zhang, H. X. and Deng, Z. T. (2002) Time series prediction via new support vector machines. *Proc. of the First Int. Conf. on Machine Learning and Cybernetics*, November 4-5 2002, Beijing: IEEE, 364–366.

APPENDICES

APPENDIX A

LIST OF PUBLICATIONS

Articles in Journals

1. A. Majid Awan and Mohd. Noor Md. Sap, "A framework for predicting oil-palm yield from climate data," *Int. Journal of Computational Intelligence*, vol. 3, no. 2, pp. 111–118, Apr. 2006.

2. A. Majid Awan, Mohd. Noor Md. Sap and M.O. Mansur, "Weighted kernel k-means for clustering spatial data," *WSEAS Transactions on Systems,* vol. 5, issue 6, pp. 1301–1308, June 2006.

3. Mohd. Noor Md. Sap and A. Majid Awan, "Stock Market Prediction using Support Vector Machine," *J. Information Technology*, Univ. Technology Malaysia, vol. 17 (2), pp. 57–69, Dec 2005.

4. Mohd. Noor Md. Sap and A. Majid Awan, "Developing an intelligent agro-hydrological system using machine learning techniques for predicting palm-oil yield," *J. Information Technology*, Univ. Technology Malaysia, vol. 17(1), pp. 66–77, June 2005.

5. Mohd. Noor Md. Sap and A. Majid Awan, "Weighted kernel k-means algorithm for clustering spatial data," *J. Information Technology*, Univ. Technology Malaysia, vol. 16 (2), pp. 137–156, Dec. 2004.

6. A. Majid Awan and Mohd. Noor Md. Sap, "Spatio-Temporal Object Relationships in Hydrological Data: A Research Perspective," *J. Information Technology*, Univ. Technology Malaysia, vol. 16 (1), pp. 76–89, June 2004.

7. Mohd Noor Md Sap, R.H. Khokhar, "Refinement of generated weighted fuzzy production rules by using fuzzy neural networks for stock market prediction," *J. Information Technology*, Univ. Technology Malaysia, vol. 17(1), pp. 34–53, June 2005.

8. R.H. Khokhar, Mohd Noor Md Sap, "Classification with degree of importance of attributes for stock market data mining," *J. Information Technology*, Univ. Technology Malaysia, vol. 16(2), pp. 21-43, December 2004.

9. Mohd Noor Md Sap, R.H. Khokhar, "Development of dynamic stock trading system based on fuzzy decision tree," *J. Information Technology*, Univ. Technology Malaysia, vol. 16 (1), pp. 09-26, June 2004.

10. R.H. Khokhar, Mohd Noor Md Sap, "Development of a compact linguistic rules-tree (CLR-Tree): The first phase," *J. Information Technology*, Univ. Technology Malaysia, vol. 15(1), pp. 30-40, June 2003.

11. Mohd. Noor Md. Sap, R.H. Khokhar, and A. Majid Awan, "Mining time series stock market data using predictive fuzzy decision tree," Submitted to Journal on Data & Knowledge Engineering, Special issue on Intelligent Data Mining, Elsevier.

12. M.N. Md. Sap, A.M. Awan, "Application of support vector machines in financial forecasting," submitted for publication to *Int. Journal of Information Technology*.

Articles in Conferences

1. A. Majid Awan, Mohd. Noor Md. Sap and M.O. Mansur, "Kernel-based algorithm for clustering spatial data," in *Proc. the 5th WSEAS Int. Conf. on Applied Computer Science*, 16–18 April 2006, Hangzhou, China.

2. Mohd. Noor Md. Sap and A. Majid Awan, "A software framework for predicting oil-palm yield from climate data," in *Proc. Int. Conf. Knowledge Mining,* Prague, 24-26 Feb. 2006.

3. A. Majid Awan and M.N. Md. Sap, "An intelligent system based on kernel methods for crop yield prediction," as a chapter in *Advances in Knowledge Discovery and Data Mining*, Proc. PAKDD'06, W.K. Ng, M. Kitsuregawa, J. Li (Eds.), Lecture Notes in Computer Science (LNCS), vol. 3918, pp. 841–846, Springer, 2006.

4. Mohd. Noor Md. Sap and A. Majid Awan, "Developing an intelligent system using kernel-based learning methods for predicting palm-oil yield," in *Proc. Int. Symp. on Bio-Inspired Computing*, Malaysia, 5-7 Sep. 2005.

5. Mohd. Noor Md. Sap and A. Majid Awan, "Finding spatio-temporal patterns in climate data using clustering," in *Proc. IEEE Int. Conf. on Cyberworlds*, Singapore, 23-25 Nov. 2005, pp. 155–164.

6. Mohd. Noor Md. Sap and A. Majid Awan, "Finding patterns in spatial data using kernel-based clustering method," in *Proc. Int. Conf. Intelligent Knowledge Systems*, Istanbul, Turkey, 06-08 July 2005.

7. A. Majid Awan and Mohd. Noor Md. Sap, "Clustering Spatial Data using a Kernel-based Algorithm," in *Proc. Postgraduate Research Seminar* 2005 (PARS'05), FSKSM, University Technology Malaysia, 17-18 May 2005.

8. Mohd. Noor Md. Sap and A. Majid Awan, "A Kernel-based Algorithm for Clustering Spatial Data," in *Proc. Malaysian Science and Technology Congress* (MSTC 2005), Kuala Lumpur, Malaysia, 18-20 April 2005.

9. Mohd Noor Md Sap, R.H. Khokhar, "Predictive fuzzy reasoning method for time series stock market data mining," *Proc. Data Mining, Intrusion Detection, Information Assurance, and Data Networks Security* 2005, part of the SPIE Defense and Security Symposium, 28 March - 1 April, Florida, USA

10. R.H. Khokhar, Mohd Noor Md Sap, "Fuzzy rules mining from decision tree for stock market prediction," *Proc. 2nd National Conference on Computer Graphics Multimedia* (CoGRAMM'04), 8-10 December 2004, Selangor, Malaysia.

11. Mohd Noor Md Sap, R.H. Khokhar, "Fuzzy decision tree for data mining of time series stock market databases," *Proc. 5th Int. Conf. on the Critical Assessment of Microarray Data Analysis* (CAMDA 2004), November 10-12, 2004, North Caroline, USA.

12. Mohd Noor Md Sap, R.H. Khokhar, " Design and development of neural bayesian approach for unpredictable stock exchange databases" *Proc. Int. Conf. on Cyberworlds* (CW 2003), IEEE Computer Society CW 2003, Marina Mandarin Singapore, 3-5 Dec 2003 Pages:364 – 371.

13. Mohd Noor Md Sap, R.H. Khokhar. "Neuro-pruning method for pruning decision tree in large databases," *Proc. 2nd Int. Conf. on Computational Intelligence, Robotics and Autonomous Systems* (CIRAS 2003), Pan Pacific Hotel, Singapore, 15-18 Dec 2003.

14. R.H. Khokhar, Mohd Noor Md Sap, "Generation linguistic rules from decision tree based on stock exchange databases," Malaysia-Japan Seminar on *Artificial Intelligence Applications in Industry*, AIAI 2003, Kuala Lumpur 24–25 June 2003.

15. R.H. Khokhar, Mohd Noor Md Sap, "Linguistic rules tree for classification of fuzzy data," Malaysia-Japan Seminar on *Artificial Intelligence Applications in Industry*, AIAI 2003, Kuala Lumpur 24–25 June 2003.

16. R.H. Khokhar, Mohd Noor Md Sap, "Design and development of intelligent knowledge discovery system for stock exchange database," Seminar ASITT Johor Malaysia, 7 May 2003.

17. Zamzarina Che Mat @ Mohd Shukor, Mohd Noor Md Sap, "Development of intelligent hybrid learning system using clustering and knowledge-based neural networks for stock exchange database," *Proc. Int. Conf. & Exhibition on Knowledge Management*, 13-15 Feb 2004, Penang, Malaysia.

18. Zamzarina Che Mat @ Mohd Shukor, Mohd Noor Md Sap, "An initial state of design and development of intelligent knowledge discovery system for stock exchange database," *Proc. Int. Conf. & Exhibition on Knowledge Management*, 13-15 Feb 2004, Penang, Malaysia.

APPENDIX B

Algorithm for A-Close method

The advantages of a close algorithm are once a close has discovered all frequent closed itemsets and their support, it can

i) directly determine the frequent itemsets and their support using algorithm as given in Figure B.1. It can reduce the problem of mining association rules to the problem of determining frequent closed itemsets and their support.

1) *Discover all frequent closed itemsets in database, i.e., itemsets that are closed and have support greater than or equal to minimum support.*
2) *Derive all frequent itemsets from the frequent closed itemsets found in phase 1.*

Figure B.1: Generate all frequent itemsets from a database.

ii) directly generate a reduced set of association rules without having to determine all frequent itemsets using algorithm as given in Figure B.2. It has lowering the algorithm computation cost.

1) *Discover all frequent closed itemsets in database*
2) *Determine the exact valid association rules basis: determine the pseudo-closed itemsets in database and then generate all rules r:* $I_1 \rightarrow I_2 \ / I_1 \subset I_2$ *where* I_2 *is a frequent closed itemset and I is a frequent pseudo-closed itemset*

Construct the reduced set of approximate valid association rules: generate rules of the form r: $I_1 \rightarrow I_2 - I_1 \ / I_1 \subset I_2$ *where* I_1 *and* I_2 *are frequent closed itemsets.*

Figure B.2: Generate the reduced set of valid association rules

APPENDIX C

Expended Attribute Selection Criterion for Fuzzy Decision Trees

Proof of Proposition 1

For the first function, one can directly check that, for each $j \in \{1,2,.....,m\}$ and $x_j \in (0,1]$,

$$\frac{\partial^2}{\partial x_j{}^2}\left(-\sum_{j=1}^{m} x_j \log_2 x_j\right) = \frac{1}{x_j \ln 2} < 0$$

which implies the first function is convex concerning each variable on $(0,1]$. A convex function attains its minimum at the extremes hence the first part of Proposition 1 is valid. To prove the second part, we can consider the function $g\ (x_1,.....x_m) = \sum_{j=1}^{m}\ (x_j - x_{j+1})$. In j the area $\{(x_1,.....x_m) \mid 1 = x_1 \geq x_2 \ldots.. \geq x_m \geq x_{m+1} = 0\}$ without losing generality. Noting that the function g can be written as $g\ (x_1,.....x_m) = \sum_{j=2}^{m}\ x_j$. In $(j / j+1)$, it is easy to check that in the considered area g gets its minimum only at (1,0,0,…..0), which completes the proof.

Proof of Proposition 2

The validity of this proposition can be given by solving $(\partial / \partial p_{ij}{}^{(k)})\ Entr_i^{(k)} = 0$ and noting $Ambig_i^{(k)} = \sum_{j=2}^{m}\ \pi_{ij}{}^{(k)}$ in $(j / j+1)$.

Proof of Proposition 3

Consider the learning problem formulated in the section 4.4.2 where the classification is assumed to be crisp. Given a linguistic term $T_j^{(k)}$ of attribute $A^{(k)}$, we examine the degree of importance of $T_j^{(k)}$ and the fuzzy entropy on $T_j^{(k)}$ (as a node). The degree of importance of $T_j^{(k)}$ is determined in terms of the classification change caused by removing $T_j^{(k)}$. For each case e, a fuzzy set defined on $\{e_1, e_2, ...e_N\}$ can be given by $I_e = \{ \lambda_{ip}^{(j)} \mid p = 1,...., N$, fixed j and fixed $i\}$ in which $\lambda_{ip}^{(j)}$ is defined in definition 5 when $i \neq p$ and is 1 when $i = p$. (In the crisp case, I_e denotes the set of cases having the same attribute values as e except $A^{(k)}$. Based on I_e, a frequency vector $(f_1, f_2, ... f_m)$ can be determined by $f_i = M(C_i \cap I_e) / M(I_e)$ where C_i represents the ith class $(i = 1, \ldots, m)$. Let $f_{n1} = \max(f_1, f_2, ... f_m)$ and $f_{n2} = \min(f_1, f_2, ... f_m)$, then for each case e, a pair $(n_1(e), n_2(e))$ is given. This proposition assumes that $\{(n_1(e), n_2(e)) \mid$ all cases $e\}$ is distribution uniformly. Nothing that the degree of important of $T_j^{(k)}$ is $\theta_j^{(k)} = (1/N(N-1))$ $\sum_i \sum_{p \neq i} g^+(\lambda_{ip}^{(k)} - \sigma_{ip})$, thus for each case e, its degree of importance is biggest when its class is $n_1(e)$ and is smallest when its class is $n_2(e)$. Noting the uniform distribution of $(n_1(e), n_2(e))$, the crisp classification and Proposition 2, we have the that either the biggest or the smallest degree of importance of $T_j^{(k)}$ corresponds to the node $T_j^{(k)}$ which has the maximum classification entropy.

If the classification is fuzzy then the frequency vector $(f_1, f_2, ... f_m)$ fails to be a probabilistic distribution but a possibilistic distribution. Similar to the above derivation, we can complete the proof by paying attention to Proposition 2.

APPENDIX D

Entropy and Gain Ratio

Entropy and Gain Ratio

Suppose we have a possible test with n outcomes that partitions the set T of training cases into subsets $T_1, T_2, ...T_n$. If this test is to be evaluated without exploring subsequent divisions of the T_i s, the only information available for guidance is the distribution of classes in T and its subsets. If S is any set of cases, let $freq(C_i, S)$ stand for the number of cases in S that belong to class C_i and $|S|$ denotes the number of cases in set S.

Consider selecting one case at random from a set S if cases and announcing that it belongs to some class C_i. This message has probability

$$\frac{freq(C_i, S)}{|S|}$$

and so the information it conveys is

$$-\log_2\left(\frac{freq(C_i, S)}{|S|}\right)$$

to find the expected information from such a message pertaining to class membership, sum over the classes in proportion to their frequencies in S, giving

$$entropy(S) = \text{info}(S) = -\sum_{j=1}^{k} \left(\frac{freq(C_i, S)}{|S|} \right) \times -\log_2 \left(\frac{freq(C_i, S)}{|S|} \right)$$

when applied to the set of training cases, info(T) measures the average amount of information needed to identify the class of a case in T. (This quantity is also known as the *entropy* of the set S.)

Now consider a similar measurement after T has been partitioned in accordance with the n outcomes of a test X. The expected information requirement can be found as the weighted sum over the subsets, as

$$\text{info}_X(T) = \sum_{i=1}^{n} \frac{|T_i|}{T} \times \text{info}(T_i)$$

The quantity $$gain(X) = \text{info}(T) - \text{info}_X(T)$$

measure the information that is gained by partitioned T in accordance with the test X. The gain criterion then selects a test to maximize this information gain (which is also known as mutual information between the test X and the class).
The bias inherent in the gain criterion can be rectified by a kind of normalization in which the apparent gain attributable to tests with many pertaining to a case that indicates not the class to which the case belongs, but the outcome of the test. By analogy with the definition of info (S), we have

$$split\ \text{info}(X) = \sum_{i=1}^{n} \frac{|T_i|}{T} - \log_2 \left(\frac{|T_i|}{T} \right)$$

This represents the potential information generated by dividing T into n subsets, whereas the information gain measures the information relevant to classification that arises from the same division. Then,

$$gain\ ratio(X) = \text{gain}(X) / \text{split info}(X)$$

expresses the proportion of information generated by the split that is useful, i.e., that appears helpful for classification. If the split is near trivial, split information will be small and this ratio will be unstable. To avoid this, the gain ratio criterion selects a test to maximize the ratio above, subject to the constraint that the information gain must be large-at least as great as the average gain over all tests examined.

APPENDIX E

A Part of Sample Data Used in Some Experiments

Open	Low	High	Close	Volume
43.48	43.45	43.6	43.6	98600
43.48	43.45	43.6	43.6	98600
43.6	43.5	43.64	43.63	32700
43.62	43.47	43.62	43.61	18100
43.61	43.6	43.7	43.68	51900
43.68	43.62	43.74	43.7	19800
43.7	43.64	43.71	43.64	17700
43.64	43.59	43.65	43.64	28700
43.64	43.6	43.7	43.67	59000
43.66	43.64	43.69	43.68	30900
43.68	43.5	43.68	43.5	30600
43.46	43.44	43.53	43.51	34500
43.53	43.5	43.59	43.53	50700
43.51	43.5	43.61	43.58	35000
43.57	43.56	43.6	43.58	28000
43.56	43.54	43.57	43.56	8400
43.52	43.52	43.58	43.58	93000
43.58	43.56	43.64	43.62	37800
43.6	43.59	43.62	43.62	8800
43.61	43.54	43.61	43.6	44500
43.6	43.56	43.62	43.56	15300
43.55	43.44	43.62	43.5	64400
43.46	43.45	43.54	43.51	79100
43.53	43.5	43.59	43.58	41700
43.58	43.58	43.61	43.6	45500
43.6	43.58	43.62	43.6	27600
43.59	43.59	43.7	43.7	16400

Open	Low	High	Close	Volume
43.7	43.69	43.78	43.74	21500
43.75	43.74	43.8	43.79	31900
43.79	43.77	43.9	43.9	98000
43.9	43.9	43.95	43.95	54100
43.95	43.8	43.95	43.8	58200
43.83	43.81	43.88	43.86	31400
43.86	43.83	43.9	43.87	31900
43.91	43.85	43.91	43.87	14300
43.87	43.82	43.91	43.9	47200
43.9	43.87	43.96	43.91	65400
43.93	43.88	43.94	43.88	8000
43.88	43.84	43.9	43.86	12800
43.88	43.79	43.88	43.79	18400
43.79	43.77	43.82	43.78	13800
43.78	43.74	43.8	43.8	57800
43.79	43.77	43.81	43.78	20200
43.8	43.76	43.8	43.77	13200
43.78	43.73	43.79	43.73	13400
43.73	43.73	43.74	43.74	11000
43.74	43.74	43.8	43.78	14900
43.77	43.75	43.8	43.75	11200
43.76	43.71	43.76	43.75	32500
43.75	43.69	43.76	43.69	44200
43.66	43.65	43.71	43.7	22400
43.71	43.7	43.73	43.71	23900
43.7	43.68	43.71	43.69	6300
43.7	43.7	43.71	43.7	14900
43.69	43.69	43.75	43.721	6300
43.74	43.71	43.77	43.74	19000
43.74	43.72	43.75	43.73	6600
43.74	43.68	43.74	43.68	13400
43.68	43.67	43.7	43.69	12200
43.69	43.67	43.7	43.69	21000
43.69	43.67	43.7	43.69	6200

Open	Low	High	Close	Volume
43.69	43.67	43.72	43.67	21900
43.67	43.67	43.69	43.68	20000
43.66	43.65	43.68	43.68	21100
43.69	43.65	43.69	43.69	29200
43.68	43.67	43.69	43.67	35300
43.68	43.65	43.68	43.66	17700
43.66	43.66	43.68	43.68	19400
43.68	43.67	43.69	43.69	22300
43.69	43.66	43.69	43.67	35800
43.66	43.65	43.68	43.67	21000
43.67	43.64	43.67	43.64	13300
43.65	43.45	43.65	43.45	73300
43.45	43.24	43.45	43.32	85400
43.31	43.29	43.38	43.31	43900
43.31	43.31	43.42	43.39	34000
43.39	43.35	43.39	43.39	25300
43.39	43.25	43.39	43.25	51600
43.25	43.13	43.25	43.15	105400
43.45	43.4	43.5	43.42	204200
43.46	43.36	43.48	43.4	18100
43.4	43.4	43.54	43.51	21700
43.51	43.43	43.79	43.75	29800
43.75	43.59	43.75	43.59	26800
43.59	43.43	43.65	43.43	34800
43.43	43.43	43.58	43.49	20800
43.47	43.46	43.49	43.49	15900
43.49	43.48	43.5	43.495	13100
43.5	43.49	43.52	43.5	55400
43.5	43.48	43.5	43.48	14200
43.48	43.42	43.49	43.45	17900
43.45	43.41	43.47	43.43	14900
43.43	43.21	43.43	43.22	33100
43.22	43.13	43.23	43.13	53800
43.18	43.1	43.19	43.18	30900

Open	Low	High	Close	Volume
43.18	43.02	43.19	43.02	65800
43.03	42.91	43.03	42.99	37200
42.99	42.99	43.1	43.09	24300
43.09	43.09	43.12	43.11	39900
43.11	43.11	43.17	43.16	22600
43.17	43.12	43.18	43.12	54900
43.12	43.12	43.28	43.17	196800
43.17	43.15	43.19	43.15	47000
43.15	43.1	43.15	43.1	327000
43.1	43.07	43.1	43.07	51800
43.07	42.99	43.07	43	30600
42.99	42.96	43	42.98	25100
42.98	42.96	42.99	42.97	32000
42.97	42.91	42.97	42.92	21100
42.92	42.92	42.94	42.93	37200
42.92	42.89	42.93	42.9	17000
42.9	42.88	42.9	42.9	17600
42.9	42.89	42.9	42.9	16800
42.9	42.89	42.9	42.89	32100
42.89	42.84	42.9	42.85	26600
42.85	42.82	42.85	42.83	27500
42.82	42.81	42.83	42.83	16300
42.83	42.82	42.85	42.84	19900
42.85	42.79	42.85	42.79	33100
42.79	42.62	42.79	42.68	33700
42.68	42.68	42.77	42.68	31500

APPENDIX F

Extraction of WFPRs from FDT in single experiment of stock market data

Explanation

The 12 extracted weighted fuzzy production rules by using conversion method. As we mentioned in section 5.5.4 for every consequent, we have three possible cases but Table 6.2 gives us only one case (case 2) of weighted fuzzy production rules. The possible explanation about case 2, mean we fire or misfire the rules are as follows

Case 2:

IF x is A_1 AND y is A_2 THEN z is C ($CF = \mu$),

Given: x is A'_1, y is A'_2, threshold values and weights of "x is A_1" and "y is A_2" are λ_{A1}, λ_{A2} and W_{A1} W_{A2} respectively,

What is the consequent C'?

Assume the degree of equality and cardinality between A'_1 and A_1; A'_2 and A_2 are $S(A'_1, A_1) = y_{A1}$ and $S(A'_2, A_2) = y_{A2}$, $AG_w = (S(A'_1, A_1) * W_{A1}/(W_{A1}+W_{A2})) + (S(A'_2, A_2) * W_{A2}/(W_{A1}+W_{A2}))$ respectively.

If either y_{A1} or y_{A2} is less than its threshold value, then the rule cannot be fired.

If $y_{A1} \geq \lambda_{A1}$ and $y_{A2} \geq \lambda_{A2}$ then AG_w and $C^{'} = \min\{1, C/(AG_w * \mu)^{0.5}\}$

Here A_1, A'_1, A_2, A'_2 and C, $C' \in F$ are the fuzzy terms such as "Low", "High", "Tall" etc. which are presented by point valued fuzzy sets.

Rule1 from Figure 4.7 (Chapter 4) is: IF [P_{LSM} = *LowAffect*(4.25) AND C_V = *Med*(3.52) AND P_{OWS} = *Good*(2.50) THEN Primary Signal = *Hold*(0.85)].IF [Change in Open = Low] AND [Change in Close = Low] THEN [Primary Signal = Down], (CF_1=0.95, w_{11}=4.25, w_{12}=6.75)

IF the share index "change in open" s is low AND the share index "change in close" s is low THEN Primary Signal is down s (CF_1 = 0.95)

Given: the share index "change in open" s is *fairly low* and its threshold value and weight are 0.6, 0.9 respectively, the share index "change in close" s is *fairly low* and its threshold value are 0.5, 0.6 respectively

What is the consequent C'?

Here we have:

A_1 = low, A'_1 = *fairly low*, A_2 = low, A'_2 = *fairly low*

C = low and C' = ?,

x = the value of share index "change in open" s

y = the value of share index "change in open" s

z = the value of share index "change in open" s

Assume X = {-0.23, -0.19, -0.14, -0.13, -0.09, -0.09, -0.08, -0.02, -0.02, -0.02} to be the universe of discourse for share index "change in open" s, Y = {-0.21, -0.19, -0.14, -0.11, -0.11, -0.10, -0.09, -0.08, -0.03, -0.01} to be the universe of discourse for share index "change in close" s and Z = {-0.49, -0.48, -0.48, -0.38, -0.36, -0.28, -0.28, -0.27, -0.27, -0.26} to be the universe of discourse for share index "change in open" s. The membership Tables for A_1, A'_1, A_2, A'_2 and C are given by Table D.2(a), D.2(b), D.3(a), D.3(b) and D.4 respectively.

Now we calculate the similarity measure between A_1 and A'_1 using proposed predictive reasoning method as:

$S_{PR}(A'_1, A_1)$ = (min(1.0, 1.0) + min(0.91, 0.98) + min(0.80, 0.92) + min(0.78, 0.87) + min(0.70, 0.78) + min(0.70, 0.75) + min(0.67, 0.74)+ min(0.54, 0.72) + min(0.54, 0.65) + min(0.54, 0.58))/(1.0+0.98+0.92+0.87+0.78+0.75+0.74+0.72+0.65+0.58)

= 7.180/7.990 = 0.899

Table D.2(a) : Membership table of A_1 = low

x	$\mu(x)$
-0.23	1.0
-0.19	0.91
-0.14	0.80
-0.13	0.78
-0.09	0.70
-0.09	0.70
-0.08	0.67
-0.02	0.54
-0.02	0.54
-0.02	0.54

Table D.2(b) : Membership table of A'_1 = fairly low

x	$\mu(x)$
-0.21	1.0
-0.19	0.98
-0.14	0.92
-0.11	0.87
-0.11	0.78
-0.10	0.75
-0.09	0.74
-0.08	0.72
-0.03	0.65
-0.01	0.58

Table D.3(a) : Membership table of A_2 = low

y	$\mu(y)$
-0.21	1.0
-0.19	0.95
-0.14	0.83
-0.11	0.76
-0.11	0.76
-0.10	0.74
-0.09	0.71
-0.08	0.69
-0.03	0.57
-0.01	0.52

Table D.3(b) : Membership table of A'_2 = fairly low

y	$\mu(y)$
-0.21	1.0
-0.19	0.97
-0.14	0.85
-0.11	0.83
-0.11	0.80
-0.10	0.78
-0.09	0.77
-0.08	0.75
-0.03	0.59
-0.01	0.56

$S_{PR}(A'_2, A_2)$ = (min(1.0, 1.0) + min(0.95, 0.97) + min(0.83, 0.85) + min(0.76, 0.83) + min(0.76, 0.80) + min(0.74, 0.78) + min(0.71, 0.77) + min(0.69, 0.75) + min(0.57, 0.59) + min(0.52, 0.56))/(1+0.97+0.85+0.83+0.80+0.78+0.77+0.75+0.59+0.56)

=7.530/7.90 = 0.953

Since 0.899 ≥ threshold value 0.6 and 0.953 ≥ threshold value 0.5 the rule will be fired. Then aggregated weighted average is defining as:

AG_w = (0.899 * 0.90 + 0.953 * 0.60) / (0.90+0.60)

= 0.921

Now rule proposition for membership Table C' can be defined as

$$C' = \min\{1, C/(AG_w * \mu)^{0.5}\}$$

The result is shown in TableD.4 (b), from this Table one can infer that C' is quite down

Table D.4(a) : Membership table of C = down

z	$\mu(z)$
-0.49	0.95
-0.48	0.85
-0.48	0.78
-0.38	0.68
-0.36	0.59
-0.28	0.45
-0.28	0.42
-0.27	0.36
-0.27	0.25
-0.26	0.10

Table D.4(b) : Membership table of consequent C' = quite down

z	$\mu(z)$
-0.49	1.0
-0.48	1.0
-0.48	0.99
-0.38	0.87
-0.36	0.75
-0.28	0.58
-0.28	0.46
-0.27	0.46
-0.27	0.32
-0.26	0.13

APPENDIX G

Case study 1-KLSE

Summary of Experiments for Predictive FDTs and WFPR's from Jan 1, 1999 to Dec 30, 2004 of different stocks from KLSE.

Years & Quarters	Number of nodes	Number of leaves	Number of WFPR's	Training accuracy	Testing accuracy
1999/1	30.65	19.92	19.92	64.12%	48.14%
1999/2	32.65	16.62	16.62	69.98%	45.21%
1999/3	35.62	19.18	19.18	68.29%	59.42%
1999/4	31.74	12.91	12.91	61.48%	59.54%
2000/1	33.54	13.48	13.48	61.99%	54.83%
2000/2	39.65	16.19	16.19	51.98%	40.94%
2000/3	31.69	10.44	10.44	65.43%	51.99%
2000/4	38.62	18.17	18.17	68.89%	53.23%
2001/1	35.62	13.35	13.35	62.48%	57.44%
2001/2	30.55	16.58	16.58	53.44%	46.41%
2001/3	37.11	15.34	15.34	59.14%	57.94%
2001/4	39.19	15.47	15.47	64.89%	59.44%
2002/1	35.91	13.74	13.74	54.49%	44.93%
2002/2	32.44	17.48	17.48	61.27%	58.99%
2002/3	33.52	12.62	12.62	54.27%	49.47%
2002/4	37.37	14.18	14.18	65.74%	53.98%
2003/1	40.95	15.48	15.48	64.12%	57.12%
2003/2	39.62	21.66	21.66	58.89%	54.94%
2003/3	35.92	18.24	18.24	51.77%	42.77%
2003/4	37.09	18.62	18.62	52.44%	43.33%
2004/1	33.81	13.78	13.78	59.44%	42.99%
2004/2	31.30	22.66	22.66	54.12%	48.14%
2004/3	34.92	11.30	11.30	59.98%	45.21%
2004/4	37.17	10.19	10.19	58.29%	49.42%

APPENDIX H

Case study 2-NYSE

Summary of Experiments for Predictive FDTs and WFPR's from Jan 1, 1999 to Dec 30, 2004 of different stocks from NYSE.

Years & Quarters	Number of nodes	Number of leaves	Number of WFPR's	Training accuracy	Testing accuracy
1999/1	25.43	15.48	15.48	51.99%	49.99%
1999/2	22.41	14.55	14.55	49.42%	44.41%
1999/3	26.23	13.19	13.19	44.49%	41.38%
1999/4	29.14	14.31	14.31	63.48%	49.84%
2000/1	23.51	12.90	12.90	62.39%	49.41%
2000/2	29.15	13.66	13.66	51.44%	44.84%
2000/3	21.39	11.11	11.11	54.14%	49.49%
2000/4	28.52	13.74	13.74	49.12%	41.49%
2001/1	25.12	15.48	15.48	54.12%	46.88%
2001/2	20.35	16.96	16.96	54.11%	45.84%
2001/3	27.19	13.84	13.84	58.39%	44.94%
2001/4	29.94	12.32	12.32	53.48%	41.84%
2002/1	25.55	14.19	14.19	54.99%	424.94%
2002/2	22.76	13.06	13.06	59.44%	48.98%
2002/3	23.74	13.44	13.44	49.82%	40.65%
2002/4	27.14	12.10	12.10	53.99%	43.04%
2003/1	20.88	10.18	10.18	44.49%	41.99%
2003/2	29.22	15.32	15.32	57.43%	49.25%
2003/3	25.21	12.41	12.41	58.44%	41.45%
2003/4	27.28	13.31	13.31	58.98%	45.84%
2004/1	23.81	12.77	12.77	51.99%	49.99%
2004/2	21.35	10.16	10.16	49.42%	44.41%
2004/3	24.98	11.34	11.34	44.49%	41.38%
2004/4	27.44	12.94	12.94	53.48%	49.84%

APPENDIX I

Case study 3-LSE

Summary of Experiments for Predictive FDTs and WFPR's from Jan 1, 1999 to Dec 30, 2004 of different stocks from LSE.

Years & Quarters	Number of nodes	Number of leaves	Number of WFPR's	Training accuracy	Testing accuracy
1999/1	31.23	14.54	14.54	60.34%	51.43%
1999/2	33.65	15.24	15.24	64.44%	52.14%
1999/3	35.12	15.81	15.81	60.44%	52.44%
1999/4	37.77	17.43	17.43	64.23%	52.80%
2000/1	32.98	14.76	14.76	61.24%	50.80%
2000/2	31.14	13.67	13.67	63.22%	50.12%
2000/3	36.25	12.12	12.12	60.00%	54.18%
2000/4	38.76	16.76	16.76	58.44%	52.48%
2001/1	32.33	18.28	18.28	60.88%	54.00%
2001/2	34.15	13.80	13.80	62.82%	53.91%
2001/3	38.15	14.44	14.44	51.43%	47.92%
2001/4	34.95	16.75	16.75	58.44%	48.93%
2002/1	32.13	14.33	14.33	54.44%	49.99%
2002/2	38.41	16.87	16.87	54.44%	44.84%
2002/3	34.26	14.18	14.18	58.22%	43.22%
2002/4	37.32	17.87	17.87	53.48%	41.40%
2003/1	41.53	13.81	13.81	54.98%	42.40%
2003/2	36.25	20.27	20.27	61.94%	59.98%
2003/3	32.90	15.43	15.43	58.91%	44.34%
2003/4	34.94	16.33	16.33	53.44%	49.33%
2004/1	31.16	15.82	15.82	50.34%	41.43%
2004/2	36.35	12.62	12.62	54.44%	42.14%
2004/3	31.12	17.02	17.02	50.44%	42.44%
2004/4	33.27	17.29	17.29	54.23%	42.80%

APPENDIX J

Three papers among the published papers

Out of the published papers during the execution of this project, the following papers are attached.

1. Mohd Noor Md Sap, R.H. Khokhar, "Predictive fuzzy reasoning method for time series stock market data mining," *Proc. Data Mining, Intrusion Detection, Information Assurance, and Data Networks Security* 2005, part of the SPIE Defense and Security Symposium, 28 March - 1 April, Florida, USA

2. A. Majid Awan and Mohd. Noor Md. Sap, "A framework for predicting oil-palm yield from climate data," *Int. Journal of Computational Intelligence*, vol. 3, no. 2, pp. 111–118, Apr. 2006.

3. A. Majid Awan, Mohd. Noor Md. Sap and M.O. Mansur, "Weighted kernel k-means for clustering spatial data," *WSEAS Transactions on Systems,* vol. 5, issue 6, pp. 1301–1308, June 2006.

Predictive fuzzy reasoning method for time series stock market data mining

[a]Rashid Hafeez Khokhar, [b]Mohd Noor Md Sap
Faculty of Computer Science and Information System
University Technology of Malaysia, K.B. 791
81310 Skudai, Johor, Malaysia
Tel: (607)-5532419, Fax: (607) 5565044
[a]rashid@siswa.utm.my, [b] mohdnoor@fsksm.utm.my

ABSTRACT

Data mining is able to uncover hidden patterns and predict future trends and behaviors in financial markets. In this research we approach quantitative time series stock selection as a data mining problem. We present another modification of extraction of weighted fuzzy production rules (WFPRs) from fuzzy decision tree by using proposed similarity-based fuzzy reasoning method called predictive reasoning (PR) method. In proposed predictive reasoning method weight parameter can be assigned to each proposition in the antecedent of a fuzzy production rule (FPR) and certainty factor (CF) to each rule. Certainty factors are calculated by using some important variables like effect of other companies, effect of other local stock market, effect of overall world situation, and effect of political situation from stock market. The predictive FDT has been tested using three data sets including KLSE, NYSE and LSE. The experimental results show that WFPRs rules have high learning accuracy and also better predictive accuracy of stock market time series data.

Keywords: Data mining, Time series, Classification, Decision tree, Fuzzy reasoning, Rules mining

1. INTRODUCTION

The stock market is a rather complicated system, and good predictions for its developments are the key to successful trading. Traders must predict stock price movements in order to sell at top range and to buy at bottom range. As stock trading is a very risky business[1], it is necessary to evaluate the risks and benefits before entering into any trading. The key to realize high profits in stock trading is to determine the suitable trading time when the risk of trading should be minimum. Thus, in order to estimate the risk in stock trading correctly, many attempts have been made for meaningful prediction from real time stock market data by using data mining and statistical techniques like Time Series [2, 3], Support Vector Machine [4], Neural Networks [5], Linear and Non-linear models [6], and Classification [7]. But these techniques to predict stock market real time data are yet to be achieved good classifiers (model).

Experts predict the stock using vague, imperfect and uncertain knowledge. As stock market prediction involves imprecise concepts and imprecise reasoning, fuzzy logic is a natural choice for knowledge representation. Fuzzy logic, founded by Zadeh [8], is formalism for reasoning with vague knowledge. Ypke [9] presents a forecasting support system which uses a fuzzy logic model for the prediction of quarterly stock market excess returns. The excess return is the market return minus the risk-free rate of return and is important for asset allocation, the diversification of an investment portfolio among the diversification of an investment portfolio among asset classes like stocks and cash. The basic models [9] shows acceptable performance, however given the great number of design parameters tuning is too complex to follow a trial and error approach.

Pellizzari and Pizzi [10] develop a fuzzy local approach to model and forecast time series. The method appears to be flexible both in modeling nonlinearities and in coping with weak non stationarities. They estimate local linear approximation (LLA) by a fuzzy weighted regression and test the model on data from a simulated noisy chaotic map and on two real financial time series, namely FIAT daily stock returns and USD-LIT exchange return rates. The LLA produces very accurate forecasts and is able to identify the correct order of the chaotic map [10]. However LLA might be more difficult to model when deal large financial datasets because of structural changes and other irregularities.

Some researchers investigate that classification is useful data mining techniques for stock market prediction [11, 12]. In recent years, there have been a growing number of studies looking at the direction or trend of movements of various kinds of financial instruments [13, 14]. However, none of these studies provide a comparative evaluation of different classification techniques regarding their ability to predict the sign of the index return. Given this notion, Mark et al., [11] examine various financial forecasting models based on multivariate classification techniques and compare them with a number of parametric and nonparametric models which forecast the level of the return.

Fuzzy knowledge representation schemes such as the fuzzy production rule (FPR) are usually in a form of fuzzy IF-THEN rule in which fuzzy labels like "tall" or "short" in the antecedent and the consequent part are allowed. If the given fact for an antecedent in a FPR does not match exactly with the antecedent of the rule, the consequent part still is drawn by technique such as fuzzy reasoning. Many existing fuzzy reasoning methods are based on Zadeh's Computational Rule of Inference (CRI) [15], which requires setting up a fuzzy relation between the antecedent and the consequent part. Despite their success in various rule-based system applications, Zadeh's CRI has been criticized to be too complex and it's underlying semantic in unclear [16]. Therefore, similarity-based methods [16, 17, 18] and linear revising methods [19] have also been proposed for reasoning. Among them, the similarity-based fuzzy reasoning methods, which make use of the degree of similarity between a given fact and the antecedent of the rule to draw the conclusion, are well known.

There are many ways to automatically acquire imprecise knowledge. Induction fuzzy decision trees are one of them. This method first generates a fuzzy decision tree and then extracts a set of fuzzy production rules from the tree. Since a fuzzy decision tree generation needs a heuristic to select expanded attributes, most researchers on fuzzy decision trees focus on the selection of expanded attributes by using a set of examples with same type of attributes [20]. So far, the heuristic used in fuzzy decision tree generation is entropy-based. Although the entropy-based heuristic has been proven to be correct for classification in information theory and can generate a relatively small tree, it cannot learn weighted fuzzy production rules which have been considered to be useful and important for representing complex knowledge [21]. It is necessary to propose an alternative heuristic which will help us generate a set of WFPR's with the required knowledge parameters from data with continuous mixed attributes. Specifically, the major contributions of this study are: 1) to demonstrate and verify the predictability of stock index direction using classification models; 2) to compare the performance of various multivariate classification techniques relative to some econometric and artificial intelligence forecasting techniques; and 3) to develop effective trading strategies guided by the predictive fuzzy reasoning method to test the performance of WFPR's.

The reminder of this paper are arranged as: section 2 presents algorithm to handle missing and unclear information, section 3 construction of FDT, section 4 will present Neuro-pruning method for pruning FDT, section 5 extract WFPR's from FDT (experimental results and discussion), and finally conclusion in section 6.

2. SIMILARITY-BASED FUZZY REASONING

The important factors to be considered in domain knowledge representation are the representability, flexibility, integrity, and expendability etc., of the stored knowledge. Weighted fuzzy production rules (WFPRs) representation provides us with such advantages, which is used to capture and represent imprecise vague or fuzzy knowledge as well as the degree of importance of each proposition contributing to a conclusion in a given production rule. In this chapter, fuzzy decision tree will be used to generate and mine WFPRs and the representation power of WFPRs will be enhanced by including several knowledge parameters such as weight and certainty factor. We will assign weight and certainty factor to each proposition of the rule and finally proposed algorithm predicts for a stock is a combination of the predictions made by each rule.

2.1 Weighted Fuzzy Production Rules (WFPRs)

The WFPRs generation from fuzzy decision tree is an extended form of fuzzy production rules (FPR) proposed by [22], WFPRs defined here is similar to the conventional production rules with the exception that fuzzy values such as "fat" or "small" are allowed in the propositions. A weight is assigned to each proposition in the antecedent part, and a certainty factor is also assigned to each rule.

A WFPR is defined as: R: IF a THEN c ($CF = \mu$), Th, w, where $a = \langle a_1, a_2, ...a_n \rangle$ is the antecedent portion which comprises of one or more propositions connected by either "AND" or "OR". Each proposition $a_i (1 \le i \le n)$ can have the format "x is f_{ai}", where f_{ai} is an element of a set of fuzzy sets F = { f_1, f_2,..... f_n }. The consequent of the rule c can be expressed, as "x is f_c", where f_c is also an element of F. The parameter μ is the certainty factor of the rule R and it represents the strength of belief of the rule. The symbol $Th = \langle \lambda_1, \lambda_2, ...\lambda_n \rangle$ represents a set of threshold values specified for the proposition in the antecedent a. The set of weights assigned to the propositions $\langle a_1, a_2, ...a_n \rangle$ is given by $w = \langle w_1, w_2, ...w_n \rangle$. The weight w_i of a proposition a_i shows the degree of importance of a_i contributing to the consequent c when comparing to other proposition a_j, for $j \neq i$. It is obvious that when there is only one proposition in the antecedent of WFPR, the weight w_i is meaningless. The set of weight w assigned to each proposition in the antecedent is referred as local weights. Another important concept called global weight, which could be assigned to each rule in an inference path, is fully explored in [23].

In general WFPR's are categorized into three types, which are defined as follows:

Type 1: A Simple Fuzzy Production Rule
R: IF a THEN c ($CF = \mu$), λ, w, For this type of rule, since there is only one proposition 'a' in the antecedent, the weight w is meaningless.

Type 2: A Composite Fuzzy Conjunction Rule
R: IF a_1 AND a_2 THEN c ($CF = \mu$), $\lambda_1, \lambda_2, w_1, w_2$,

Type 3: A Composite Fuzzy Disjunction Rule
R: IF a_1 OR a_2 THEN c ($CF = \mu$), $\lambda_1, \lambda_2, w_1, w_2$,

For both types 2 and 3, λ_i is the threshold value for a_i and w_i is the weight assigned to a_i. Some authors do not assign a certainty factor to a FPR while others ignore the weight and the threshold value assigned to each proposition in the antecedent. We consider that the capturing of fuzzy knowledge using fuzzy production rule with weights and threshold values plays an important role in real world applications. Hence the weight (degree of importance), the threshold value as well as the certainty factor have been taken into account.

2.2 Weighted Fuzzy Production Rules with Single Antecedent

R: IF A THEN C, ($CF = \mu$), Th, W e.g., if a_1 then C, ($CF = \mu$), $Th = \{\lambda_{a_1}\}$, $W = \{w_1\}$. C is represented as a "concluded disorder" Chen's Diagnosis problem [18] or a consequent in other problems.
Given four cases of facts:

Case 1: $A' = A$
Case 2: $A' =$ very A
Case 3: $A' =$ more or less A
Case 4: $A' =$ not A

What conclusion C' can be drawn? In order to draw the conclusion C', proposed predictive reasoning method will be analyzed all of these possible cases and select the most accurate case which is best suited for classification of stock market prediction.

2.3 Weighted Fuzzy Production Rules with Multiple Antecedents

If the antecedent portion or consequence portion of fuzzy production rule contains "AND" or "OR" connectors, then it is called a composite fuzzy production rule. According to [24], the composite fuzzy production rule can be distinguished into the following rule-types:

Type 1: IF a_{j1} AND, a_{j2} AND…AND a_{jn} THEN a_k ($CF = \mu_i$)

Type 2: IF a_j THEN a_{k1} AND , a_{k2} AND…..AND a_{kn} ($CF = \mu_i$)

Type 3: IF a_{j1} OR, a_{j2} OR…OR a_{jn} THEN a_k ($CF = \mu_i$)

Type 4: IF a_j THEN a_{k1} OR , a_{k2} OR…..OR a_{kn} ($CF = \mu_i$)

In our algorithm, we use multiple propositions connected by "AND"

R: IF A THEN C, ($CF = \mu$), Th, W e.g., if a_1 AND a_2 THEN C, ($CF = \mu$), $Th = \{\lambda_{a1}, \lambda_{a2}\}$, $W = \{w_1, w_2\}$ $A = \langle a_1, a_2 \rangle$, a_1 AND a_2 are connected by "AND" consider the following four cases:

Case 1: $A' = \langle a_1', a_2' \rangle = A = \langle a_1, a_2 \rangle$

Case 2: A' = very $A = \langle$ very a_1, very $a_2 \rangle$ $A \langle a_1', a_2' \rangle \langle a_1', a_2' \rangle \langle a_1', a_2' \rangle$

Case 3: $A' = \langle a_1', a_2' \rangle =$ more or less $A = \langle$ more or less a_1, more or less $a_2 \rangle$

Case 4: $A' = \langle a_1', a_2' \rangle =$ not $A = \langle$ not a_1, not $a_2 \rangle$

What conclusion can be drawn?

2.4 Transformation of WFPRs from FDT

In this paper, we concentrate more one evaluation of WFPR's by using proposed predictive reasoning method that will be present in next section. Therefore, we use the fuzzy rules that we have extracted from predictive fuzzy decision tree [12, 25].

The transformation from the tree to WFPRs is described as follows.

(1) Each path of branch from the root to a leaf can be converted into a rule. The antecedent of the rule represents the attributes on the passing branches from the root to the leaf and the consequent of the rule represents the cluster labeled at the leaf node.

(2) The degree of importance of each attribute value (linguistic term) of the expanded attribute, $\theta_j^{(k)}$, is regarded as the value of the weight w_{ij} of the corresponding proposition of the converted rule.

(3) For the converted rule R_i :

$IF < A_{i1}$ *and* A_{i2} ..... *and* $A_{in} > THEN$ B_i $(CF, w_{i1}, w_{i2}, ...w_{in})$ in which the weight w_{ij} are obtained from step (2) and the certainty factor is defined as

$$CF_i = \sum_{i=1}^{N} (\wedge_{j=1}^{n} (w_{ij} A_{ij}(k) \wedge B_i(k) / \sum_{i=1}^{N} (\wedge_{j=1}^{n} (w_{ij} A_{ij}(k))) \quad (1)$$

where A_{ij} and B_i are fuzzy sets on $\{1,2,.....,N\}$ and $\wedge$ denotes the minimum.

3. Proposed Predictive Reasoning (PR) Method

In order to compare all existing reasoning methods [16, 17, 18] under the same conditions and with a unified rule, we assume that the rule R has CF =1, W ={1}, and the threshold value Th is small enough for the value to be fired for all cases. For Case 4, i.e., A' = not A, we do not expect the rule to be fired, but it is possible that rule miss-firing may result, i.e., the rule is fired with the consequent C' not equal to "not C" due to the assignment of a small threshold value. So the

above rule R is equivalent to a conventional fuzzy production rule "IF A THEN C". The relationship between the fuzzy set and its modifiers such as "very", "more or less" and "not" satisfies the following relationship proposed by [26]:

1) $a_i' = \text{very } a_i = a_i^2$ i.e., $\mu_{very\ a_i}(x) = [\mu_{ai}(x)]^2$

2) $a_i' = \text{more or less } a_i = a_i^{0.5}$ i.e., $\mu_{more,\text{or less}\ a_i}(x) = [\mu_{ai}(x)]^{0.5}$

3) $a_i' = \text{not } a_i = 1 - a_i$ i.e., $\mu_{not\ a_i}(x) = 1 - \mu_{ai}(x)$

In proposed predictive reasoning (PR) method [25] first simple rule case has been considered, where only one observation A' and one simple rule in the form $R: A \to C$ is presented. The basic idea is to modify the consequent C of a simple rule $R: A \to C$ according to the closeness of an observation (fact) A' to the antecedent (pattern) A of the rule R. If they are close (similar) enough in comparison to a threshold, then the rule R can be fired and the consequent can be deduced by some modification technique to be shown later in the sequel.

Formally, their "closeness" is expressed as a similarity measure (SM), which in turn is obtained form a distance measure (DM). More specifically, given as SM we will present a threshold $\lambda_\circ$ of the SM. Once the SM $(A', A) = \lambda$ between A' and A exceeds the threshold $\lambda_\circ$, we fire the rule R, that is we construct a modification function (MF) based on $\lambda: MF = f(\lambda),$ and use this MF to modify the right side C of the rule R to deduce a consequent C'. The selection of $\lambda_\circ$ above 0.5 and closer to 1 assures better similarity A and A'. Hence we believe specification of a $\lambda_\circ$ does improve the behavior of consequents.

3.1 Similarity Measure

The similarity measure between linguistic terms is defining their membership functions. In this section, we propose another similarity-based fuzzy reasoning method, which calculates the similarity between A and A' using equality and cardinality as [17]. In our algorithm certainty factor (CF) is the key point. In previous similarity-based method certainty factor is just assign for strength of every rule but here we first calculate certainty factor by applying some variables in stock market. Let us denote such similarity measure as $S_{PR}(a'_i, a_i)$, which is defined as

$$S_{PR}(a'_i, a_i) = 1 - \frac{f(a'_i \nabla a_i)}{f(a_i)} \tag{2}$$

Where ∇ is the symmetrical difference of A and A' viz. $\forall x \in X, \mu_{A \nabla A'} = | \mu_{A'}(x) - \mu_A(x) |$. Thus the equation can be expressed as

$$S_{PR}(a'_i, a_i) = 1 - \frac{\sum_{x \in X} | \mu_{a'_i}(x) - \mu_{a_i}(x) |}{\sum_{x \in X} \mu_{a_i}(x)} \tag{3}$$

3.2 Aggregated Weighted Average

It is observed that $S_{EC}(a'_i, a_i)$, which is the degree of equality of a'_i in a_i, differs from $S_{EC}(a'_i, a_i)$ and $0 \le S_{EC}(a'_i, a_i) \le 1,$ for instance, if $a_i' = a_i$, then $S_{EC}(a'_i, a_i) = 1$. Again if $S_{EC}(a'_i, a_i) \ge \lambda_{ai}$ for all a_i, the rule can be fired and the aggregated weighted average, AG_w is defined as

$$AG_w = \sum_{i=1}^{n} AG_{wi} = \sum_{i=1}^{n} \left[S_{EC}(a_i', a_i) * \frac{w_i}{\sum_{j=1}^{n} w_j} \right] \qquad (4)$$

After the aggregated weighted average has been calculated, two modification functions are proposed to modify the consequent C'. These two forms are also enhancements of those in [16].

3.3 Modification Functions (MF's)

In the proposed PR method, a rule $R_j : A_j \rightarrow C_j$ is to be fired with the use of an MF that modifies the consequent C_j of the rule R_j. The MF is dependent on SM and its construction is subjective. That is one would if required adjust the form of the MF based on experts experience or historical data so that the system can function as close to the real situation as possible. However this enables us to bypass the matrix operations of CRI (Computational Rule of Inference) it was first proposed by Zadeh [15] to deduce a consequent. There could be many forms of modification functions; we present two as possible examples here.

After the aggregated weighted average has been calculated, two modification functions are proposed to modify the consequent C'.

(1) More or Less Form:

This is an analogy to the definition of linguistic hedge "more or less" (Zadeh's, [15]). The induced consequent C'_j is obtained through

$$C'_j = \min\{1, C_j (AG_w * \mu)^{0.5}\} \qquad (5)$$

Using the similarity transformation SM = $(1+DM)^{-1}$ the "more or less" form of the modification function can be simplified as follows

$$C'_j = \min\{1, C_j * (1 + DM)\} \qquad (6)$$

Where * represents multiplication.

Figure 1 shows the effect of this form of MF. We observe that additional uncertainty is introduced with such a modification function, i.e., large value of distance measure (DM) increase the values of membership. Obviously, at the extreme case, i.e., when $A'_j = A_j$ for the rule R_j, they are indeed exactly matched, i.e., DM=0 and hence the consequent C_j will not be altered based on the proposed schema. This is one of the advantages of similarity based over CRI; recall that the CRI will generally produce a different result other than C_j i.e., $A_j \circ (A_j \rightarrow C_j) = C'_j \neq C_j$.

(2) Membership Value Reduction Form:

In some sense this is an imitation of CRI. We simple multiply the consequent C_j by SM (A'_j, A_j) for a given observation A'_j . i.e.,

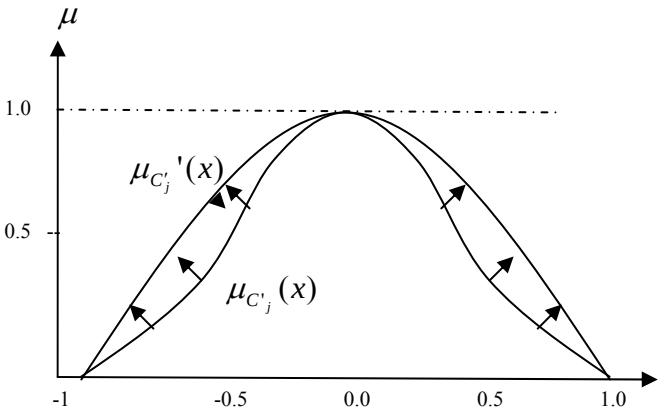


Figure 1: More or Less form

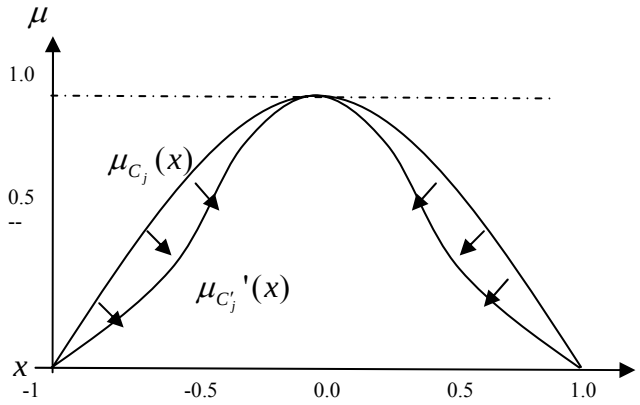


Figure 2: Membership value reduction form

$$C'_j = C_j * (AG_w * \mu)^{0.5}\} \qquad (7)$$

Figure 2 shows the effect of this form of modification function. It gradually reduces the membership value of C_j according to SM. In other words the smaller the SM greater the reduction of membership value of C_j , which imitates the min operator's contraction effect of CRI. However, it is avoids the max operator's expansion. At the extreme case that is when $A'_j = A_j$, we have SM (A'_j , A_j)=1 and hence $C'_j = C_j$. Again this shows an advantage of the proposed PR method over CRI.

3.4 Rules Propositions

The following shows what C'_j can be drawn for each of the three cases:

Case 1: The antecedent A_j has only one proposition $AG_w = S_{EC}(a_1', a_1) = S_{EC}(A', A)$

Since $w_1 / \sum_{j=1}^{n} w_j = 1$ with n=1

If $S_{DS}(a'_1, a_1) \geq \lambda_{a1}$ then $C'_j = \min\{1, C_j / (AG_w * \mu)^{0.5}\}$ or $C'_j = C_j * (AG_w * \mu)^{0.5}$ Depending on whether we want to restrict or dilate the membership value of C_j .

Case2: The antecedent A_j has two or more proposition connected by "AND"

$$AG_w = \sum_{i=1}^{n} (S_{EC}(a_i', a_i) * \frac{w_i}{\sum_{j=1}^{n} w_j}).$$

If $S_{DS}(a'_i, a_i) \geq \lambda_{ai}$ for all $(1 \leq i \leq n)$, then $C'_j = \min\{1, C_j / (AG_w * \mu)^{0.5}\}$ or $C'_j = C_j * (AG_w * \mu)^{0.5}$

Case3: The antecedent A has two or more propositions connected by "OR"
This rule can be split into n simple rules as shown in Case1, i.e.,

$AG_w = S_{EC}(a_i', a_i)$ Because for each $i = 1,....,n, \sum_{j=1}^{n} w_j$ is reduced to a single term w_i, and becomes $w_i / w_i = 1$

If $\exists_i$ s.t. $S_{DS}(a'_i, a_i) \geq \lambda_{ai} \ \forall \ (1 \leq i \leq n)$, then $C'_j = \min\{1, C_j / \max(AG_{w1}, AG_{w2}, ...AG_{wi}) * \mu)^{0.5}\}$
or, $C'_j = C_j * \max(AG_{w1}, AG_{w2}, ...AG_{wi}) * \mu)^{0.5}$ OR
If $S_{DS}(a'_i, a_i) \geq \lambda_{ai} \ \forall \ (1 \leq i \leq n)$, then $C'_j = \min\{1, C_j / \max(AG_{w1}, AG_{w2}, ...AG_{wi}) * \mu)^{0.5}\}$ or,
$C'_j = C_j * \max(AG_{w1}, AG_{w2}, ...AG_{wi}) * \mu)^{0.5}$.
where μ is defined as:

$$\mu = \left| \frac{\gamma + \delta}{2} - \rho - o \right|^{0.5} \tag{8}$$

where γ, δ, ρ, o are active variables that can affect stock market.
IF a_1 is fa_1 AND a_2 is fa_2 THEN C is fc, ($CF = \mu$), $Th = \{\lambda_{a1}, \lambda_{a2}\}$, $W = \{w_1, w_2\}$ e.g., IF Open is low AND Close is low THEN Signal is low

3.5 Fuzzy Reasoning Mechanism

Let $C = \{\lambda_1, \lambda_2, ...\lambda_n\}$ be an object to be classified, F be a group of WFPRs extracted from the tree, and there are k samples. The initial state of C with respect to classification is set to be $(x_1, x_2, ..., x_k) = (0,0,.....0)$.

Step 1:

From the group F, select a rule $R: IF \ A \ THEN \ B \ [CF, Th, Lw]$ where the antecedent A is supposed to be $(A_1, A_2, ...A_n)$; the consequent B to be $(b_1, b_2, ...b_n)$; and the local weight Lw to be $(Lw_1, Lw_2, ...Lw_n)$.

Step 2:

Compute the membership degree of λ_j belonging to $A_j : SM_j = A_j(\lambda_j)$ where the membership function of the fuzzy set A_j is denoted by itself.

Step 3:

Let $Th = (Th_1, Th_2, ...Th_n)$ be the threshold. If for each proposition A_j the inequality $SM_j \geq Th_j$ holds, then the rules are executed. Computed the overall weighted average SM_w of similarity measures as $SM_w = \sum_{j \in T} (Lw_j / \sum_{i \in T} Lw_i) SM_j$ where $T = \{j : SM_j \geq Th_j\}$.

Step 4:

Modify the matching-consequent according to one of the beforehand given modification strategies:

4a) more or less form: $B^* = \min\{1, B / SM_w\}$;

4b) membership-value reduction form: $B^* = B_* SM_w$;

4c) keeping the consequent of the rule unchanged (no modification): $B^* = B$.

Step 5:

Compute the certainty degree of B^* as $CF_{B^*} = CF * SM_w$;

Step 6:

Put $x_j = \max(CF_{B^*}, x_j)$ where j is a number corresponding to the sample of the consequent B^*.

Repeat the above six steps until each rule within the group F has been applied to the object C.

4. EXPERIMENTS

Fuzzy decision tree algorithms have found fast greedy search classification data mining method for stock market predictions. It searches through the space of fuzzy production rules to find those likely to the useful for classifying items in a future stock market database. Some examples of weighted fuzzy production rules found define by single experiment of predictive fuzzy decision tree are shown in Figure Table 1. For example, the first rule reads, "IF [Change in Open (strength = 0.0245) == Low] AND [Change in Close (strength = 0.0345) == Low] THEN [Primary Signal (-0.2564) == Down], (CF_1 =0.95, w_{11} =4.25, w_{12} =6.75)" then predict unexceptional (Down) return with some strength. We will discuss in detail about fuzzy rules extraction in next sections.

In single day experiment, we use the 78 information (share stock information from 9:30 to 3:55) on 100 different companies during the period October 1, 1999 to October 31, 2004 from KLSE, NYSE and LSE to build predictive FDT. In this research we focus on every movement occur during 5 minutes of period, therefore for every single day experiment the total size of information is 78 and the size of equal set of pattern in single experiment is 30. The developed predictive FDT for single experiment shows in Figure 3. We remove the first information and add the next information for every next single experiment of size 30. Repeat the recursive process until we will have reached the 78th information of single day stock market and approximately we have 43 continuous experiments. This process will remove the discontinuity that is important for prediction of time series stock market data. However on every experiment of the above process, we will mine the best accurate weighted fuzzy production rules as test information by giving different parameters to find the best predicted sets of rules for prediction of next information of stock market. We will repeat the same process for one quarter (66 days) and during quarterly experiment we may have approximately 2838 experiments. Similarly, for whole experiments we may have 42570 experiments for 20 quarters (period October 1, 1999 to October 31, 2004 from KLSE, NYSE and LSE). We will discuss in next subsection that how we mine the best rules from the predictive fuzzy decision trees on every quarter.

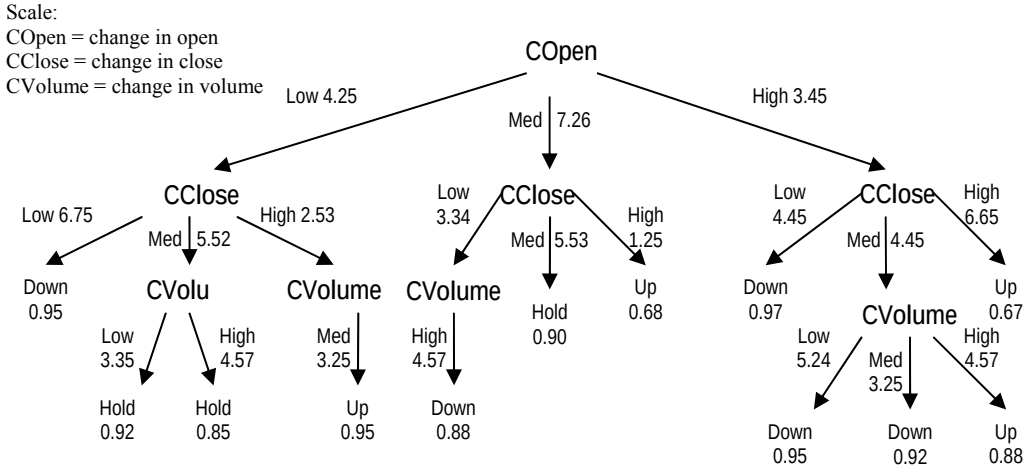


Figure 3: Predictive FDT by using proposed algorithm to train sets of patterns in single experiment

4.1 Extraction of Weighted Fuzzy Production Rules from Predictive Fuzzy Decision Tree

In Figure 3 each path of branches from the root to a leaf can be converted into a rule. The antecedent of the rule represents the attributes on the passing branches from the root to the leaf and the consequent of the rule represents the labeled at the leaf node. Table 1 shows the result as some weighted fuzzy production rules by using the proposed conversion method for extraction of rules from predictive fuzzy decision tree for single experiment. Similarly, we extract the weighted fuzzy production rules for 43 experiments in single day and 2838 experiments in one quarter (66 days). One quarter may have many rules, our task is to mine rules that have highest accuracy and also we repeat the same process for 20 quarters form October 1, 1999 to October 31, 2004.

We apply the best accurate rules after mining from 20 quarters October 1, 1999 to October 31, 2004. Figure 4 shows the results of 30 core rules points after mining the best accurate rules for next one month prediction. In Figure 4, every core point presents the rules for one day prediction and also these rules compare with the actual stock market prediction for one month prediction. We use these core points of rules as optimal signals of stock prices for different stocks. We can use these rules to calculate the optimal signals for 3 days, 5 days and one month stock price prediction. Optimal signals are actually that signals we calculate for final stock market prediction on the bases of above predicted rules.

Table 1: An example of 12 extracted WFPRs from FDT in single experiment for stock market prediction

Rule No	Weighted fuzzy production rules
1	IF [Change in Open = Low] AND [Change in Close = Low] THEN [Primary Signal = Down], (CF_1 =0.95, W_{11} =4.25, W_{12} =6.75)
2	IF [Change in Open = Low] AND [Change in Close = Med] AND [Volume = Low] THEN [Primary Signal = Hold], (CF_2 =0.92, W_{21} =4.25, w_{22} =5.52, W_{23} =3.35)
3	IF [Change in Open = Low] AND [Change in Close = Med] AND [Volume = High] THEN [Primary Signal = Hold], (CF_3 =0.85, W_{31} =4.25, w_{32} =5.52, W_{33} =4.57)
4	IF [Change in Open = Low] AND [Change in Close = High] AND [Volume = Med] THEN [Primary Signal = Up], (CF_4 =0.95, W_{41} =4.25, w_{42} =2.53, W_{43} =3.25)
5	IF [Change in Open = Med] AND [Change in Close = Low] AND [Volume = High] THEN [Primary Signal = Down], (CF_5 =0.88, W_{51} =7.26, w_{52} =3.34, W_{53} =4.57)
6	IF [Change in Open = Med] AND [Change in Close = Med] THEN [Primary Signal = Hold], (CF_6 =0.90, W_{61} =7.26, w_{62} =5.53)
7	IF [Change in Open = Med] AND [Change in Close = High] THEN [Primary Signal = Up], (CF_7 =0.68, W_{71} =7.26, w_{72} =1.25)
8	IF [Change in Open = High] AND [Change in Close = Low] THEN [Primary Signal = Down], (CF_8 =0.97, W_{81} =3.45, w_{82} =4.45)
9	IF [Change in Open = High] AND [Change in Close = Med] AND [Volume = Low] THEN [Primary Signal = Down], (CF_9 =0.95, W_{91} =3.45, w_{92} =4.45, W_{93} =5.24)
10	IF [Change in Open = High] AND [Change in Close = Med] AND [Volume = Med] THEN [Primary Signal = Down], (CF_{10} =0.92, W_{101} =3.45, w_{102} =4.45, W_{103} =3.25)
11	IF [Change in Open = High] AND [Change in Close = Med] AND [Volume = High] THEN [Primary Signal = Up], (CF_{11} =0.88, W_{111} =3.45, w_{112} =4.45, W_{113} =4.57)
12	IF [Change in Open = High] AND [Change in Close = High] THEN [Primary Signal = Up], (CF_{12} =0.67, W_{121} =3.45, w_{122} =6.65)

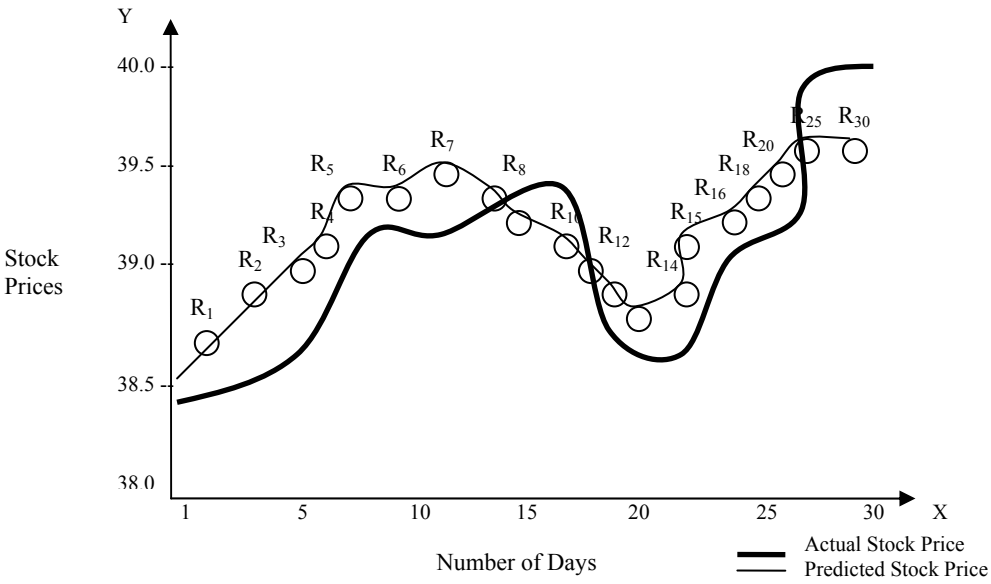


Figure 4: Results for core points of rules for one month prediction

5. CONCLUSIONS

In this paper, data mining techniques have been used to uncover hidden patterns and predict future trends and behaviors in financial markets. In data mining, fuzzy decision tree classification has been developed to extract weighted fuzzy production rules for stock market prediction. In similarity-based fuzzy predictive reasoning method, we analyze WFPR's which are extracted from FDT. The analysis is based on the result of consequent drawn for different given facts (e.g. variables that can affect stock market) of the antecedent. We present the results obtained with the predictive reasoning method in this research over three data sets including KLSE, NYSE and LSE. We test the experimental results by using per day information on 100 different companies during the period October 1, 1999 to October 31, 2004 (20 Quarters) from KLSE, NYSE and LSE to build predictive fuzzy decision trees. The experimental results show that among the large number of rules fuzzy predictive reasoning method can mine rules from every quarter that have high learning predictive accuracy of time series stock market prediction. We used our classifier to rank the stocks according to its certainty that they would exhibit exceptional returns during the one quarter and predict for the future stock prices.

REFERENCES

1. G. A. Torben, and J. Lund. Estimating continuous-time stochastic volatility models of the short-term interest rate. *Journal of Econometrics* 77 (1997), pp. 343–378.
2. R. Agrawal, Lin, K.-I., H.S. Sawhney, K. Shim. Fast Similarity Search in the Presence of Noise, Scaling, and Translation in Time-Series Databases,º Proc. 21st Int'l Conf. Very Large Data Bases(VLDB '95), Sept. 1995, pp. 490±501.
3. J. Han, W. Gong, Y. Yin. Efficient mining of partial periodic patterns in time series databases. *Proc. Of International Conference of Data Engineering (ICDE99)*, Sydney, Australia, Mar.
4. A. Fan, M. Palaniswami. Stock selection using support vector machines. *Proceedings. IJCNN '01. International Joint Conference on Neural Networks, 2001,* Volume: 3 , 15-19 July 2001 Pages:1793 - 1798 vol.3
5. Xiaohua, Wang., Paul, Kang., Hoh, Phua., Weidong, Lin. (2003). Stock market prediction using neural networks: does trading volume help in short-term prediction? Neural Networks, 2003. *Proceedings of the International Joint Conference on Neural Networks* ,Volume: 4 , July 20 - 24, 2003 Pages:2438 – 2442
6. M. Chinn, Coibion. O. LeBlanc. The predictive characteristics of energy futures: Recent evidence for crude oil, natural gas, gaoline and heating oil. *Presented at UCSC working* paper # 409.

7. R. Agarwal, C. Aggarwal, V. V. V. Prasad. A tree projection algorithm for generation of frequent itemsets. *In Journal of Parallel and Distributed Computing (Special Issue on High Performance Data Mining)*, (to appear), 2000.
8. L. A. Zadeh's. Fuzzy sets. *Information and Control,* 8, 1965, pp. 338-353
9. Ypke Hiemstra. A Stock Market Forecasting Support System Based on Fuzzy Logic. HICSS (3) 1994: 281-288
10. P. Pellizzari, and C. Pizzi. Fuzzy-like conditional density estimation in time series outliers detection, *submitted to Technometrics.*
11. T. Mark, Leung, Hazem. Daouk., An-Sing, Chen. Forecasting stock indices: a comparison of classification and level estimation models • MISCELLANEOUS *International Journal of Forecasting, Volume 16, Issue 2, April-June 2000, Pages 173-190*
12. R. H. Khokhar, and Mohd. Md. Sap Noor. Design and Development of Neural Bayesian Approach for Unpredictable Stock Exchange Databases. *IEEE Computer Society CW 2003, 2nd International Conference on Cyberworlds (CW 2003)*, Marina Mandarin Singapore, 3-5 Dec 2003 Pages:364 – 371.
13. Y. Wu, and H. Zhang. Forward premiums as unbiased predictors of future currency depreciation: A non-parametric analysis. *Journal of International Money and Finance* 16, 609-623.
14. M. O'Connor, W. Remus, K. Griggs. Going up-going down: How good are people at forecsting trends and changes in trends? *Journal of Forecasting* 16, 165-176.
15. L. A. Zadeh's. Outline of a new approach to the analysis of complex systems and decision processus. *IEEE Trans. Syst., Man, Cybern.*, vol SMC-3, pp. 28-44, 1973.
16. I. B. Turksen, and Zhong. An approximate analogical reasoning approach based on similarity measures. *IEEE Trans. Syst., Man, Cybern.*, vol 18, pp. 1049-1056, 1989
17. D. S. Yeung, and E. C. C. Tsang. A comparative study on similarity-based fuzzy reasoning methods. Systems Man and Cybernetics, Part B, IEEE Transactions on , Volume: 27 , Issue: 2 , April 1997 Pages:216 – 227
18. S. M. Chen. A weighted fuzzy reasoning algorithm for medical diagnosis. *Decision Support Syst.*, 1994, vol.11, pp 37-43.
19. M. Mukaidono, L. Ding, Z. Shen. Approximate reasoning based on revision principle. *In Proc. NAFIPS'90*, vol. 1, 1990, pp. 94-97.
20. X. Z. Wang, and J. R. Hong. On the handling of fuzziness for continuous-valued attributes in decision tree generation, *Fuzzy Sets and Systems*, vol. 99, pp. 283-290, 1998
21. D. S. Yeung and E. C. C. Tsang. A multilevel weighted fuzzy reasoning algorithm for expert systems. *IEEE Transactions on SMC*, vol. 28, pp. 149-158, 1998.
22. D. S. Yeung, and E. C. C. Tsang. Improved fuzzy knowledge representation and rule evaluation using fuzzy Petri nets and degree of subsethood. *Intell. Syst.*, vol. 9, no. 12, pp. 1083-110, 1994.
23. D. S. Yeung, and E. C. C. Tsang. A weighted fuzzy production rule evaluation method Fuzzy Systems. 1995. International Joint Conference of the Fourth IEEE International Conference on Fuzzy Systems and The Second International Fuzzy Engineering Symposium., Proceedings of 1995 IEEE International Conference on, Volume: 2, 20-24 March 1995 Pages: 461 - 468 vol.2
24. C. G. Looney. Fuzzy controls via Boolean rule matrix transformations," IEEE Trans, Syst., Man, Cyber., vol. SMC-17, no. 6 , pp. 1077-1082, Nov/Dec, 1987.
25. Mohd. Md. Sap. Noor, and R. H. Khokhar. Development of Dynamic Stock Trading System Based on Fuzzy Decision Tree" Journal Technology Maklumat (Journal of Information Technology), Jilid 16, Bil.1 09-27, June, 2004.
26. L. A. Zadeh's. A theory of commonsense knowledge. *In Fuzzy Sets and Applications: Selected Paper by L. A. Zadeh, R. R. Yager, et al. Ed.* New York: Wiley, 1987, pp. 615-653.

A Framework for Predicting Oil-Palm Yield from Climate Data

A. Majid Awan and Mohd. Noor Md. Sap

***Abstract*—**Intelligent systems based on machine learning techniques, such as classification, clustering, are gaining wide spread popularity in real world applications. This paper presents work on developing a software system for predicting crop yield, for example oil-palm yield, from climate and plantation data. At the core of our system is a method for unsupervised partitioning of data for finding spatio-temporal patterns in climate data using kernel methods which offer strength to deal with complex data. This work gets inspiration from the notion that a non-linear data transformation into some high dimensional feature space increases the possibility of linear separability of the patterns in the transformed space. Therefore, it simplifies exploration of the associated structure in the data. Kernel methods implicitly perform a non-linear mapping of the input data into a high dimensional feature space by replacing the inner products with an appropriate positive definite function. In this paper we present a robust weighted kernel k-means algorithm incorporating spatial constraints for clustering the data. The proposed algorithm can effectively handle noise, outliers and auto-correlation in the spatial data, for effective and efficient data analysis by exploring patterns and structures in the data, and thus can be used for predicting oil-palm yield by analyzing various factors affecting the yield.

***Keywords*—**Pattern analysis, clustering, kernel methods, spatial data, crop yield

I. Introduction

THE contribution of agriculture in the economic growth of Malaysia can be substantially improved through better management practices. Oil-palm has become an important crop in Malaysia. However, oil-palm production potential is reduced when trees are exposed to stressful weather conditions. Low moisture is the most common stressful condition oil-palm faces, so monitoring rainfall and other related parameters (e.g. temperature, pressure, soil moisture, sun-shine duration, humidity, etc.) is useful in predicting oil palm yield levels. The lagged effect of weather in Malaysia has implications for global vegetable oil prices in general and for the palm oil market in particular. Moreover, not enough is known about the daily patterns of rainfall or sunshine illumination levels to determine what may mitigate the expected negative effects of the heavy or below-normal rainfall [1]. Keeping this importance in view, this study is aimed at investigating the impacts of hydrological and meteorological conditions on oil-palm plantation using computational machine learning techniques. Resulting improved understanding of the factors affecting oil-palm yield would not only help in accurately predicting yield levels but would also help substantially in looking for mitigating solutions.

Clustering and classification are very useful machine learning techniques which can capture meaningful patterns in the agro-hydrological data. However, complexity of geographic data precludes the use of general purpose pattern discovery and data analysis techniques. Data clustering, a class of unsupervised learning algorithms, is an important and applications-oriented branch of machine learning. Its goal is to estimate the structure or density of a set of data without a training signal. It has a wide range of general and scientific applications such as data compression, unsupervised classification, image segmentation, anomaly detection, etc. There are many approaches to data clustering that vary in their complexity and effectiveness, due to the wide number of applications that these algorithms have. While there has been a large amount of research into the task of clustering, currently popular clustering methods often fail to find high-quality clusters. Clustering has received a renewed attention with the advent of nonlinear clustering methods based on kernels as it provides a common means of identifying structure in complex data.

A number of kernel-based learning methods have been proposed in recent years [2–9]. However, much research effort is being put up for improving these techniques and in applying these techniques to various application domains. Generally speaking, a kernel function implicitly defines a non-linear transformation that maps the data from their original space to a high dimensional space where the data are expected to be more separable. Consequently, the kernel methods may achieve better performance by working in the new space. While powerful kernel methods have been proposed for supervised classification and regression problems, the development of effective kernel method for clustering, aside from a few tentative solutions [2, 5, 10], needs further investigation.

Finding good quality clusters in spatial data (e.g., temperature, precipitation, pressure, etc.) is more challenging because of its peculiar characteristics such as auto-correlation (i.e., measured values that are close in time and space tend to be highly correlated, or similar), non-linear separability,

The authors are with the Faculty of Computer Science & Information Systems, University Technology Malaysia, Skudai 81310, Johor, Malaysia (Phone:+60-75532542, Fax:+60-75565044, email: awanmajid@hotmail.com; mohdnoor@fsksm.utm.my, respectively).

outliers, noise, high-dimensionality, and when the data have clusters of widely differing shapes and sizes [11–13]. The popular clustering algorithms, like k-means, have some limitations for this type of data [13, 14]. Therefore, we present a weighted kernel k-means clustering algorithm incorporating spatial constraints, bearing spatial neighborhood information, in order to handle spatial auto-correlation, outliers and noise in the spatial data.

This paper is organized as follows. In the next section, a brief overview of the problem area and methodology is given. In section III, it is pointed out how kernel-based methods can be useful for clustering non-linearly separable and high-dimensional spatial (climate) data. In section IV, the proposed algorithm–a weighted kernel k-means algorithm with spatial constraints–is presented which could be useful for handling noise, outliers and auto-correlation in the spatial data. Some experimental results of the proposed clustering algorithm on rainfall data are given in section V. In section VI, some related work on using clustering approach for studying the effect of climate on vegetation is given. Finally the paper concludes in section VII.

II. Application Area and Methods

This work focuses on clustering spatial data, e.g. for finding patterns in rainfall, temperature, pressure data, so that their impact on other objects like vegetation (specifically oil-palm yield) could be explored. A very simplified view of the problem domain looks as shown in Figure 1. The data consist of a sequence of snapshots of the earth areas taken at various points in time. Each snapshot consists of measurement values for a number of variables, e.g., temperature, pressure, precipitation, crop yield, etc. All attribute data within a snapshot are represented using spatial frameworks, i.e., a partitioning of the study region into a set of mutually disjoint divisions which collectively cover the entire study region. This way we would be dealing with spatial time series data.

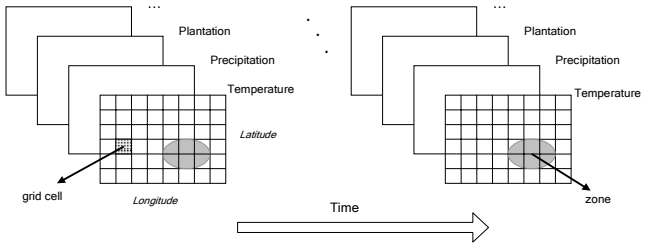


Fig. 1 A simplified view of the problem domain

Clustering, often better known as spatial zone formation in this context, segments land into smaller pieces that are relatively homogeneous in some sense. A goal of the work is to use clustering to divide areas of the land into disjoint regions in an automatic but meaningful way that enables us to identify regions of the land whose constituent points have similar short-term and long-term characteristics. Given relatively uniform clusters, we can then identify how various phenomena or parameters, such as precipitation, influence the climate and oil-palm produce (for example) of different areas.

The spatial and temporal nature of our target data poses a number of challenges. For instance, such type of data are noisy, and display autocorrelation. In addition, such data have high dimensionality (for example, if we consider monthly precipitation values at 1000 spatial points for 12 years, then each time series would be 144 dimensional vector), clusters of non-convex shapes, outliers.

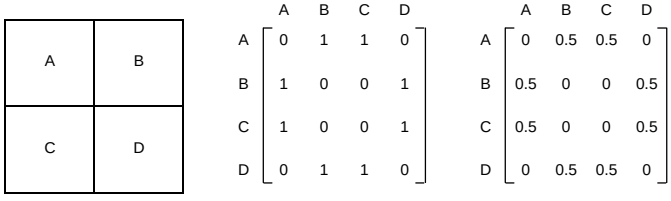
A. Problems with classical data analysis techniques for spatial data

Classical statistical data analysis algorithms often make assumptions (e.g., independent, identical distributions) that violate the first law of Geography, which says that everything is related to everything else but nearby things are more related than distant things. Ignoring spatial autocorrelation may lead to residual errors that vary systematically over space exhibiting high spatial autocorrelation [12]. The models derived may not only turn out to be biased and inconsistent, but may also be a poor fit to the dataset [13].

One way of modeling spatial dependencies is to add a spatial autocorrelation term in the regression equation. This term contains a neighborhood relationship contiguity matrix. Such spatial statistical methods, however, are computationally expensive due to their reliance on contiguity matrices that can be larger than the spatial datasets being analyzed [12]. A brief description of the contiguity matrix and spatial autoregression model is given below (for detail, pls. refer to [12]).

The spatial relationship among locations in a spatial framework is often modeled via a contiguity matrix. A simple contiguity matrix may represent a neighborhood relationship defined using adjacency, Euclidean distance, etc. Example definitions of neighborhood using adjacency include a four-neighborhood and an eight-neighborhood.

Figure 2-a shows a girded spatial framework with four locations, A, B, C, and D. A binary matrix representation of a four-neighborhood relationship is shown in Figure 2-b. The row-normalized representation of this matrix is called a contiguity matrix, as shown in Figure 2-c. Other contiguity matrices can be designed to model neighborhood relationships based on distance. The essential idea is to specify the pairs of locations that influence each other along with the relative intensity of interaction.



a. Spatial framework b. Neighbor relationship c. Contiguity matrix

Fig. 2 A spatial framework and its four-neighborhood contiguity matrix

In the spatial autoregression model, the spatial dependencies of the error term, or, the dependent variable, are directly modeled in the regression equation. If the dependent values y_i are related to each other, then the regression equation can be modified as

$$y = \rho Wy + X\beta + \varepsilon$$

Here W is the neighborhood relationship contiguity matrix and ρ is a parameter that reflects the strength of the spatial dependencies between the elements of the dependent variable. After the correction term ρWy is introduced, the components of the residual error vector ε are then assumed to be generated from independent and identical standard normal distributions.

B. System overview and architecture

For this application, the yield values of oil-palm plantation areas over a span of time constitute time series. The analysis of these and other time series (e.g., precipitation, temperature, pressure, etc.) can be conducted using clustering technique. If we apply a clustering algorithm to cluster time series associated with points on the land, we obtain clusters that represent land regions with relatively homogeneous behavior. The centroids of these clusters are time series that summarize the behavior of these land areas, and can be represented as indices. Given relatively uniform clusters we can then identify how various parameters, such as precipitation, temperature, etc., influence the climate and oil-palm produce of different areas using correlation. This way clustering can better help in detailed analysis of our problem.

A simplified architecture of the agro-hydrological system is shown in Figure 3:

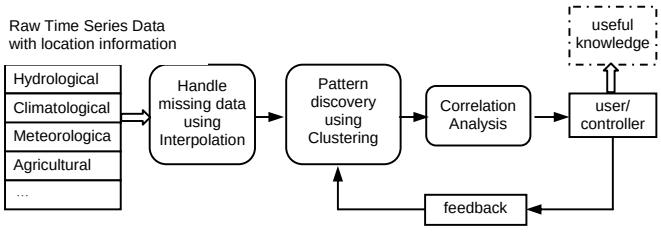


Fig. 3 A simplified architecture for the system

It is worth mentioning that we use Rough sets approach for handling missing data in the preprocessing stage of the system. However, the main component of the system, namely clustering algorithm, is elaborated in this paper.

III. KERNEL-BASED METHODS

The kernel methods are among the most researched subjects within machine-learning community in recent years and have been widely applied to pattern recognition and function approximation. Typical examples are support vector machines [15–17], kernel Fisher linear discriminant analysis [18], kernel principal component analysis [10], kernel perceptron algorithm [19], just to name a few. The fundamental idea of the kernel methods is to first transform the original low-dimensional inner-product input space into a higher dimensional feature space through some nonlinear mapping. The complex nonlinear problems in the original low-dimensional space can more likely be linearly treated and solved in the transformed space according to the well-known Cover's theorem. However, usually such mapping into high-dimensional feature space will undoubtedly lead to an exponential increase of computational time, i.e., so-called curse of dimensionality. Fortunately, adopting kernel functions to substitute an inner product in the original space, which exactly corresponds to mapping the space into higher-dimensional feature space, is a favorable option. Therefore, the inner product form leads us to applying the kernel methods to cluster complex data [5, 17, 20].

Support vector machines and kernel-based methods: Support vector machines (SVM), having its roots in machine learning theory, utilize optimization tools that seek to identify a linear optimal separating hyperplane to discriminate any two classes of interest [18, 21]. When the classes are linearly separable, the linear SVM performs adequately.

There are instances where a linear hyperplane cannot separate classes without misclassification, an instance relevant to our problem domain. However, those classes can be separated by a nonlinear separating hyperplane. In this case, data may be mapped to a higher dimensional space with a nonlinear transformation function. In the higher dimensional space, data are spread out, and a linear separating hyperplane may be found. This concept is based on the Cover's theorem on the separability of patterns. According to the Cover's theorem, an input space made up of nonlinearly separable patterns may be transformed into a feature space where the patterns are linearly separable with high probability, provided the transformation is nonlinear and the dimensionality of the feature space is high enough. Figure 4 illustrates that two classes in the input space may not be separated by a linear separating hyperplane, a common property of spatial data, e.g. rainfall patterns in a green mountain area might not be linearly separable from those in the surrounding plain area. However, when the two classes are mapped by a nonlinear transformation function, a linear separating hyperplane can be found in the higher dimensional feature space.

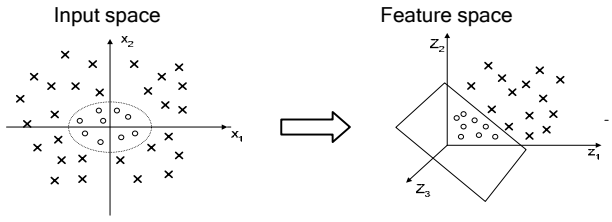


Fig. 4 Mapping nonlinear data to a higher dimensional feature space where a linear separating hyperplane can be found, e.g., via the nonlinear map $\Phi(x) = (z_1, z_2, z_3) = \left([x]_1^2, [x]_2^2, \sqrt{2}[x]_1[x]_2\right)$

Let a nonlinear transformation function ϕ maps the data into a higher dimensional space. Suppose there exists a function K, called a kernel function, such that,

$$K(x_i, x_j) = \phi(x_i) \cdot \phi(x_j)$$

A kernel function is substituted for the dot product of the transformed vectors, and the explicit form of the transformation function ϕ is not necessarily known. In this way, kernels allow large non-linear feature spaces to be explored while avoiding curse of dimensionality. Further, the use of the kernel function is less computationally intensive. The formulation of the kernel function from the dot product is a special case of Mercer's theorem [7].

Examples of some well-known kernel functions are given in Table I.

TABLE I
SOME WELL-KNOWN KERNEL FUNCTIONS

Polynomial	$K(x_i, x_j) = < x_i, x_j >^d$	d is a positive *integer*
Radial Basis Function (RBF)	$K(x_i, x_j) = \exp(-\|x_i - x_j\|^2 / 2\sigma^2)$	σ is a user defined value
Sigmoid	$K(x_i, x_j) = \tanh(\alpha < x_i, x_j > + \beta)$	α, β are user defined values

IV. PROPOSED WEIGHTED KERNEL K-MEANS WITH SPATIAL CONSTRAINTS

It has been reviewed in [22] that there are some limitations with the k-means method, especially for handling spatial and complex data. Among these, the important issues/problems that need to be addressed are: i) non-linear separability of data in input space, ii) outliers and noise, iii) auto-correlation in spatial data, iv) high dimensionality of data. Although kernel methods offer power to deal with non-linearly separable and high-dimensional data but the current methods have some drawbacks. Both [15, 2] are computationally very intensive, unable to handle large datasets and autocorrelation in the spatial data. The method proposed in [15] is not feasible to handle high dimensional data due to computational overheads, whereas the convergence of [2] is an open problem. With regard to addressing these problems, we propose an algorithm–weighted kernel k-means with spatial constraints– in order to handle spatial autocorrelation, noise and outliers present in the spatial data.

Clustering has received a significant amount of renewed attention with the advent of nonlinear clustering methods based on kernels as it provides a common means of identifying structure in complex data [2, 3, 5, 6, 15, 22–24]. Before elaborating the weighted kernel k-means algorithm, we fix the notation as: let $X = \{ x_i \}_{i=1, \ldots, n}$ be a data set with $x_i \in R^N$. We call codebook the set $W = \{ w_j \}_{j=1, \ldots, k}$ with $w_j \in R^N$ and $k << n$. The Voronoi set (V_j) of the codevector w_j is the set of all vectors in X for which w_j is the nearest vector, i.e.,

$$V_j = \{x_i \in X \,|\, j = arg \min_{j=1,\ldots,k} \|x_i - w_j\|\}$$

For a fixed training set X, the quantization error $E(W)$ associated with the Voronoi tessellation induced by the codebook W can be written as:

$$E(W) = \sum_{j=1}^{k} \sum_{x_i \in V_j} \|x_i - w_j\|^2 \quad (1)$$

K-means is an iterative method for minimizing the quantization error $E(W)$ by repeatedly moving all codevectors to the arithmetic mean of their Voronoi sets. In the case of finite data set X and Euclidean distance, the centroid condition reduces to

$$w_j = \frac{1}{|V_j|} \sum_{x_i \in V_j} x_i \quad (2)$$

where $|V_j|$ denotes the cardinality of V_j. Therefore, k-means is guaranteed to find a local minimum for the quantization error [25, 26].

The k-means clustering algorithm can be enhanced by the use of a kernel function; by using an appropriate nonlinear mapping from the original (input) space to a higher dimensional feature space, one can extract clusters that are non-linearly separable in input space. Usually the extension from k-means to kernel k-means is realized by expressing the distance in the form of kernel function [7]. The kernel k-means algorithm can be generalized by introducing a weight for each point x, denoted by $u(x)$. This generalization would be powerful for making the algorithm more robust to noise and useful for handling auto-correlation in the spatial data. Using the non-linear function ϕ, the objective function of weighted kernel k-means can be defined as:

$$E(W) = \sum_{j=1}^{k} \sum_{x_i \in V_j} u(x_i) \|\phi(x_i) - w_j\|^2 \quad (3)$$

where,
$$w_j = \frac{\sum_{x_j \in V_j} u(x_j) \phi(x_j)}{\sum_{x_j \in V_j} u(x_j)} \quad (4)$$

Why weighting? As the weather station readings or agriculture related attribute values do not often represent equal areas, in such cases the use of weighted means instead of means becomes necessary. A 'weight' attributed to a mean or other value usually signifies importance. If weighting is not used, then each value that enters into a calculation of mean would have the same importance. Sometimes, this is not a realistic or fair way to calculate a mean. Individual observations often, for various reasons, are of varying importance to the total, or average value.

The Euclidean distance from $\phi(x)$ to center w_j is given by (all computations in the form of inner products can be replaced by entries of the kernel matrix) the following eq.

$$\left\| \phi(x_i) - \frac{\sum_{x_j \in V_j} u(x_j)\phi(x_j)}{\sum_{x_j \in V_j} u(x_j)} \right\|^2 = K(x_i, x_i) - 2\frac{\sum_{x_j \in V_j} u(x_j)K(x_i, x_j)}{\sum_{x_j \in V_j} u(x_j)} + \frac{\sum_{x_j, x_l \in V_j} u(x_j)u(x_l)K(x_j, x_l)}{(\sum_{x_j \in V_j} u(x_j))^2} \tag{5}$$

In the above expression, the last term is needed to be calculated once per each iteration of the algorithm, and is representative of cluster centroids. If we write

$$C_k = \frac{\sum_{x_j, x_l \in V_j} u(x_j)u(x_l)K(x_j, x_l)}{(\sum_{x_j \in V_j} u(x_j))^2} \tag{6}$$

With this substitution, (5) can be re-written as

$$\left\| \phi(x_i) - \frac{\sum_{x_j \in V_j} u(x_j)\phi(x_j)}{\sum_{x_j \in V_j} u(x_j)} \right\|^2 = K(x_i, x_i) - 2\frac{\sum_{x_j \in V_j} u(x_j)K(x_i, x_j)}{\sum_{x_j \in V_j} u(x_j)} + C_k \tag{7}$$

For increasing the robustness of fuzzy c-means to noise, an approach is proposed in [27]. Here we propose a modification to the weighted kernel k-means to increase the robustness to noise and to account for spatial autocorrelation in the spatial data. It can be achieved by a modification to (3) by introducing a penalty term containing spatial neighborhood information. This penalty term acts as a regularizer and biases the solution toward piecewise-homogeneous labeling. Such regularization is also helpful in finding clusters in the data corrupted by noise. The objective function (3) can, thus, be written, with the penalty term, as:

$$E(W) = \sum_{j=1}^{k} \sum_{x_i \in V_j} u(x_i) \left\| \phi(x_i) - w_j \right\|^2 + \frac{\gamma}{N_R} \sum_{j=1}^{k} \sum_{x_i \in V_j} u(x_i) \sum_{r \in N_k} \left\| \phi(x_r) - w_j \right\|^2 \tag{8}$$

where N_k stands for the set of neighbors that exist in a window around x_i and N_R is the cardinality of N_k. The parameter γ controls the effect of the penalty term. The relative importance of the regularizing term is inversely proportional to the accuracy of clustering results.

The following can be written for kernel functions, using the kernel trick for distances [28]

$$\left\| \phi(x_i) - w_j \right\|^2 = K(x_i, x_i) - 2K(x_i, w_j) + K(w_j, w_j)$$

If we adopt the Gaussian radial basis function (RBF), then $K(x, x) = 1$, so (8) can be simplified as

$$E(W) = 2\sum_{j=1}^{k} \sum_{x_i \in V_j} u(x_i)(1 - K(x_i, w_j)) + \frac{\gamma}{N_R} \sum_{j=1}^{k} \sum_{x_i \in V_j} u(x_i) \sum_{r \in N_k} (1 - K(x_r, w_j)) \tag{9}$$

The distance in the last term of (8), can be calculated as

$$\left\| \phi(x_r) - \frac{\sum_{x_j \in V_j} u(x_j)\phi(x_j)}{\sum_{x_j \in V_j} u(x_j)} \right\|^2 = 1 - 2\frac{\sum_{x_j \in V_j} u(x_j)K(x_r, x_j)}{\sum_{x_j \in V_j} u(x_j)} + \frac{\sum_{x_j, x_l \in V_j} u(x_j)u(x_l)K(x_j, x_l)}{(\sum_{x_j \in V_j} u(x_j))^2} \tag{10}$$

If we write the second term on the right side of (10) as β_r, we get:

$$\left\| \phi(x_r) - \frac{\sum_{x_j \in V_j} u(x_j)\phi(x_j)}{\sum_{x_j \in V_j} u(x_j)} \right\|^2 = 1 - 2\frac{\sum_{x_j \in V_j} u(x_j)K(x_r, x_j)}{\sum_{x_j \in V_j} u(x_j)} + C_k = 1 - \beta_r + C_k \tag{11}$$

For RBF, (7) can be written as

$$\left\| \phi(x_i) - \frac{\sum_{x_j \in V_j} u(x_j)\phi(x_j)}{\sum_{x_j \in V_j} u(x_j)} \right\|^2 = 1 - 2\frac{\sum_{x_j \in V_j} u(x_j)K(x_i, x_j)}{\sum_{x_j \in V_j} u(x_j)} + C_k \tag{12}$$

As first terms on the right sides of (11) and (12) do not play any role for finding minimum distance, so it can be omitted.

For finding clusters, we have to calculate the distance from each point to every cluster representative. This can be obtained from (8) after incorporating the penalty term containing spatial neighborhood information by using (11) and (12). Hence, the effective minimum distance can be calculated using the following expression:

$$-2\frac{\sum_{x_j \in V_j} u(x_j)K(x_i, x_j)}{\sum_{x_j \in V_j} u(x_j)} + C_k + \frac{\gamma}{N_R} \sum_{r \in N_k} (\beta_r + C_k) \tag{13}$$

Now, the algorithm, weighted kernel k-means with spatial constraints, can be written as follows.

Algorithm SWK-means: Spatial Weighted Kernel k-means
(weighted kernel k-means with spatial constraints)

SWK-means (K, k, u, N, γ, ε)
Input: K: kernel matrix, k: number of clusters, u: weights for each point, set $\varepsilon > 0$ to a very small value for termination, N: information about the set of neighbors around a point, γ: penalty term parameter,
Output: $w_1, \ldots, w_k$: partitioning of the points

1. Initialize the k clusters: $w_1 = 0, \ldots, w_k = 0$
2. Set $i = 0$,
3. For each cluster, compute $C(k)$ using (6),
4. For each point x, find its new cluster index as

   $j(x) = arg\ min_j \left\| \phi(x) - w_j \right\|^2$ using (13),

5. Compute the updated clusters as

   $w_j^{(i+1)} = \{x : j(x)=j\}$

6. Repeat steps 3 to 5 until the following termination criterion is met:

   $$\left\| W_{new} - W_{old} \right\| < \varepsilon$$

   where, $W = \{w_1, w_1, w_1, \ldots, w_k\}$ are the vectors of cluster centroids.

A. Handling Outliers

This subsection briefly discusses about spatial outliers, i.e., observations which appear to be inconsistent with their neighborhoods. Detecting spatial outliers is useful in many applications of geographic information systems and spatial databases, including transportation, ecology, public safety, public health, climatology, location-based services, and severe weather prediction. Informally, a spatial outlier is a local instability (in values of non-spatial attributes) or a spatially referenced object whose non-spatial attributes are extreme relative to its neighbors, even though the attributes may not be significantly different from the entire population.

We can examine how (13) makes the algorithm robust to outliers. As $K(x_i, x_j)$ measures the similarity between x_i and x_j, and when x_i is an outlier, i.e., x_i is far from the other data points, then $K(x_i, x_j)$ will be very small. So, the second term in the above expression (13) will get very low value or, in other words, the weighted sum of data points will be suppressed. The total expression will get higher value and hence results in robustness by not assigning the point to the cluster.

B. Scalability

The pruning procedure used in [29, 30] can be adapted to speed up the distance computations in the weighted kernel k-means algorithm. The acceleration scheme is based on the idea that we can use the triangle inequality to avoid unnecessary computations. According to the triangle inequality, for a point x_i, we can write, $d(x_i, w_j^n) \geq d(x_i, w_j^o) - d(w_j^o, w_j^n)$. The distances between the corresponding new and old centers, $d(w_j^o, w_j^n)$ for all j, can be computed. And this information can be stored in a $k \times k$ matrix. Similarly, another $k \times n$ matrix can be kept that contains lower bounds for the distances from each point to each center. The distance from a point to its cluster centre is exact in the matrix for lower bounds. Suppose, after a single iteration, all distances between each point and each center, $d(x_i, w_j^o)$, are computed. In the next iteration, after the centers are updated, we can estimate the lower bounds from each point x_i to the new cluster center, w_j^n using $d(w_j^o, w_j^n)$ calculations and the distances from the previous iteration, i.e., we calculate the lower bounds as $d(x_i, w_j^o) - d(w_j^o, w_j^n)$. The distance from x_i to w_j^n is computed only if the estimation is smaller than the distance from x_i to its cluster center. This estimation results in sufficient saving in computational time. Once we have computed lower bounds and begun to compute exact distances, the lower bound allows us to determine whether or not to determine the remaining distances exactly.

V. Experimental Results

The system is implemented in C++. We get very hopeful results regarding analyzing various factors impacting oil-palm yield. Here we mention the results in a brief manner, especially of the clustering algorithm as the SWK algorithm is at the core of the overall system.

Given a data matrix, whose rows consists of time series from various points on the land (rainfall stations), the objective is to discover temporal and/or spatial patterns in the data. If we apply clustering algorithm to the rainfall time series associated with points on the land (surroundings of rainfall stations), we obtain clusters that represent land regions with relatively homogeneous behavior. The centroids of these clusters are time series that summarize the behavior of those land areas.

For experimentation we selected 24 rainfall stations. A 12-month moving average is used for removing seasonality from the data. For monthly rainfall values for 5 years, we get a data matrix of 24×60. SWK-means partitioned it into 2 clusters. We also applied the algorithm to the monthly average rainfall values of this period, for easy visualization of results. Its results are shown in Figure 5. As the locations of rainfall stations are known, the clustering results can be easily mapped on the physical locations on the map. Actually the clusters will summarize the time series associated with relevant regions, and when results are plotted for a longer period, some contiguous regions will be formed.

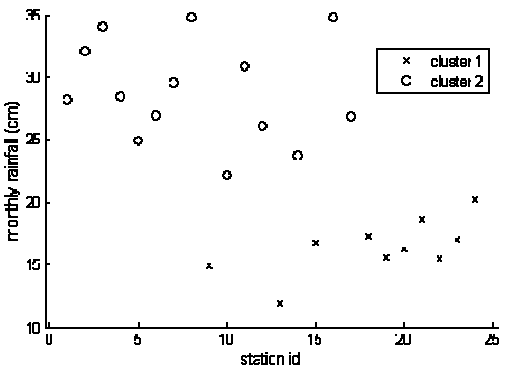


Fig. 5 Clustering results of SWK-means algorithm showing two clusters of monthly rainfall (average) of 24 stations

Since the kernel matrix is symmetric, we only keep its upper triangular matrix in the memory. For the next five year periods of time for the selected 24 rainfall stations, we may get data matrices of 48×60, 72×60 and so on. The algorithm proportionally partitioned the data into two clusters. The corresponding results are shown in Table II (a record represents 5-year monthly rainfall values taken at a station). It also validates proper working of the algorithm.

TABLE II
RESULTS OF SWK-MEANS ON RAINFALL DATA AT 24 STATIONS FOR 5, 10, 15, 20, 25, 30, 35 YEARS

No. of Records	No. of records in cluster 1	No. of records in cluster 2
24	10	14
48	20	28
72	30	42
96	40	56
120	50	70
144	60	84
168	70	98

For the overall system, the information about the landcover areas of oil palm plantation is gathered. The yield values for these areas over a span of time constitute time series. The analysis of these and other time series (e.g., precipitation, temperature, pressure, etc.) is conducted using clustering technique. Clustering is helpful in analyzing the impact of various hydrological and meteorological variables on the oil palm plantation. It enables us to identify regions of the land whose constituent points have similar short-term and long-term characteristics. Given relatively uniform clusters, we can then identify how various parameters, such as precipitation, temperature, etc., influence the climate and oil-palm produce of different areas using correlation. Our initial study shows that the rainfall patterns alone affect oil-palm yield after 6-7 months. This way we are able to predict oil-palm yield for the next 1-3 quarters on the basis of analysis of present plantation and environmental data.

VI. RELATED WORK

In this section we discuss some of the related work for illustrating issues involved in clustering our application domain data.

In [31], a mixture model approach is used to identify the cluster structure in atmospheric pressure data. (Mixture models assume that the data are generated probabilistically from a mixture of Gaussian distributions and use the data to estimate the parameters of these distributions). This approach is related to k-means [32], but has two advantages. First, it assigns a "membership" probability to each data point and each cluster. These probabilities provide a measure of the uncertainty in cluster membership. Second, it is sometimes possible to estimate the most appropriate choice for k [31]. (It is also possible to estimate the best k for k-means by plotting the overall error or similarity for different values of k and looking for the knee in the plot).

Another possible approach to clustering, particularly in spatially oriented domains, is to use "region growing." Starting with individual points as clusters, each cluster is grouped with the most similar, physically adjacent cluster, until there is only one cluster. (Sometimes various criteria are applied to prevent clusters from being merged if the resulting cluster is too "poor"). This approach can be viewed as a form of hierarchical clustering which has the constraint that clusters can only be merged if the resulting cluster is contiguous, i.e., not split into disconnected sets of points [33].

However, it is sometimes desirable to have clusters that are "piecewise contiguous," i.e., consist of points which are similar, but not all in one contiguous region. An example of such an approach is presented in [34] and was applied to the problem of land use classification based on spectral image data. The technique, Recursive Hierarchical Image Segmentation, consists of alternating steps in which similar, adjacent regions are merged (a region growing step) and similar, non-adjacent regions are merged (a spectral clustering step). For land use classification, this allows the grouping of points, which may represent the same type of land cover, but which are in disconnected regions.

A related work [35] introduces ACTS (Automatic Classification of Time Series), a clustering method for remote sensing time series. (The data considered are NDVI, the Normalized Difference Vegetation Index, or greenness index [36]). The goal of this work is to use clustering as an initial step for deriving continental-scale to global-scale vegetation maps. After the removal of components with a period of one year or less, clustering is also used to group points that had similar patterns of inter-annual variation in NDVI. However, there was no investigation of the relationships between different regions of the land and the climate factors.

Another related work is reported in [13]. The goal of this work is to use climate variables, such as long term sea level pressure (SLP) and sea surface temperature (SST), to discover interesting patterns relating changes in NPP (Net Primary Production–plant growth) to land surface climatology and global climate. (NPP is the net assimilation of atmospheric carbon dioxide into organic matter by plants). In this work, ocean climate indices (OCIs) and rainfall indices are found from the relevant time series using shared nearest neighbor clustering technique. This way clustering is used to find interesting connections between land and ocean regions for the discovery of interesting ecosystem patterns.

While there has been considerable research into hierarchical clustering and spatial clustering [11], many issues still remain. Some of the new issues of zone formation are zonal formation over time, the multi-scale nature of the data, and constrained zone formation.

VII. CONCLUSIONS

It is highlighted how computational machine learning techniques, like clustering, can be effectively used in analyzing the impacts of various hydrological and meteorological factors on vegetation. Then, a few challenges especially related to clustering spatial data are pointed out. Among these, the important issues/problems that need to be addressed are: i) non-linear separability of data in input space, ii) outliers and noise, iii) auto-correlation in the spatial data, iv) high dimensionality of data.

The kernel methods are helpful for clustering complex and high dimensional, and non-linearly separable data. Consequently for developing a system for oil-palm yield prediction, an algorithm, weighted kernel k-means incorporating spatial constraints, is presented which is a central part of the system. The proposed algorithm has the mechanism to handle spatial autocorrelation, noise and outliers in the spatial data. We get promising results on our test data sets. It is hoped that the algorithm would prove to be robust and effective for spatial (climate) data analysis, and it would be very useful for oil-palm yield prediction.

REFERENCES

[1] P. Provance, "Malaysia: Below-normal rainfall may limit palm oil yields late 2004-early 2005," US Department of Agriculture, Production Estimates & Crop Assessment Div., Aug. 2004. Available: http://www.fas.usda.gov/pecad/highlights/2004/08/maypalm/index.htm

[2] F. Camastra, A. Verri, "A novel kernel method for clustering," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 27, pp. 801-805, May 2005.

[3] I.S. Dhillon, Y. Guan, and B. Kulis, "Kernel kmeans, spectral clustering and normalized cuts," in *Proc. 10th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining,* pp. 551–556, 2004.

[4] C. Ding and X. He, "K-means clustering via principal component analysis," in *Proc. 21st Int. Conf. Machine Learning*, July 2004, pp. 225–232.

[5] M. Girolami, "Mercer kernel based clustering in feature space," *IEEE Trans. Neural Networks*, vol. 13(3), pp. 780–784, May 2002.

[6] A.M. Awan and M.N. Md. Sap, "An intelligent system based on kernel methods for crop yield prediction," in *Advances in Knowledge Discovery and Data Mining*, W.K. Ng, M. Kitsuregawa, J. Li (Eds.). Lecture Notes in Computer Science (LNCS), vol. 3918, pp. 841–846, Springer, 2006.

[7] B. Scholkopf and A. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, 2002.

[8] F. Camastra, "Kernel methods for unsupervised learning," PhD thesis, University of Genova, Italy, 2004.

[9] L. Xu, J. Neufeld, B. Larson, and D. Schuurmans, "Maximum margin clustering," in *Proc. Advances in Neural Information Processing Systems* (NIPS2004)*,* 2004.

[10] B. Scholkopf, A. Smola, and K. R. Müller, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural Computation*, vol. 10, no. 5, pp. 1299–1319, 1998.

[11] J. Han, M. Kamber and K. H. Tung, "Spatial clustering methods in data mining: a survey," in *Geographic Data Mining and Knowledge Discovery*, Harvey J. Miller and Jiawei Han (eds.). Taylor and Francis, 2001.

[12] S. Shekhar, P. Zhang, Y. Huang, and R. Vatsavai, "Trends in spatial data mining," as a chapter in *Data Mining: Next Generation Challenges and Future Directions*, H. Kargupta, A. Joshi, K. Sivakumar, and Y. Yesha (eds.). MIT Press, 2003.

[13] P. Zhang, M. Steinbach, V. Kumar, S. Shekhar, P-N Tan, S. Klooster, and C. Potter, "Discovery of patterns of earth science data using data mining," as a Chapter in *Next Generation of Data Mining Applications*, J. Zurada and M. Kantardzic (eds.). IEEE Press, 2003.

[14] M. Steinbach, P-N. Tan, V. Kumar, S. Klooster and C. Potter, "Data mining for the discovery of ocean climate indices," in *Proc. 5th Workshop on Scientific Data Mining* at *2nd SIAM Int. Conf. on Data Mining,* 2002.

[15] A. Ben-Hur, D. Horn, H. Siegelman, and V. Vapnik, "Support vector clustering," *J. Machine Learning Research*, vol. 2(Dec.), pp. 125–137, 2001.

[16] N. Cristianini and J.S.Taylor, *An Introduction to Support Vector Machines*. Cambridge Academic Press, 2000.

[17] V.N. Vapnik, *Statistical Learning Theory*. John Wiley & Sons, 1998 .

[18] V. Roth and V. Steinhage, "Nonlinear discriminant analysis using kernel functions," in *Advances in Neural Information Processing Systems* 12, S. A Solla, T. K. Leen, and K.-R. Muller (Eds.), pp. 568–574. MIT Press, 2000.

[19] J. H. Chen and C. S. Chen, "Fuzzy kernel perceptron," *IEEE Trans. Neural Networks*, vol. 13, pp. 1364–1373, Nov. 2002.

[20] D.S. Satish and C.C. Sekhar, "Kernel based clustering for multiclass data," in *Proc. Int. Conf. on Neural Information Processing*, Kolkata, India, Nov. 2004.

[21] V.N. Vapnik, *The Nature of Statistical Learning Theory*. Springer-Verlag, New York, 1995.

[22] A.M. Awan, M.N Md. Sap and M.O. Mansur, "Weighted kernel k-means for clustering spatial data," *WSEAS Trans. on Systems,* vol. 5, issue 6, pp. 1301–1308, June 2006.

[23] M.N. Md. Sap, and A. Majid Awan, "Developing an intelligent agro-hydrological system using machine learning techniques for predicting palm-oil yield," *J. Information Technology*, Univ. Technology Malaysia, vol. 17(1), pp. 66–77, Jun. 2005.

[24] M.N. Md. Sap, and A. Majid Awan, "Finding spatio-temporal patterns in climate data using clustering," in *Proc. IEEE Int. Conf. on Cyberworlds*, Singapore, 23-25 November 2005, pp. 155–164.

[25] R.M. Gray, *Vector Quantization and Signal Compression*. Kluwer Academic Press, Dordrecht, 1992.

[26] S.P. Lloyd, "An algorithm for vector quantizer design," *IEEE Trans. Communications*, vol. 28, no. 1, pp. 84–95, 1982.

[27] M.N. Ahmed, S.M. Yamany, N. Mohamed, A.A. Farag and T. Moriarty, "A modified fuzzy c-means algorithm for bias field estimation and segmentation of MRI data," *IEEE Trans. Medical Imaging*, vol. 21, pp.193–199, 2002.

[28] B. Scholkopf, "The kernel trick for distances," in *Advances in Neural Information Processing Systems* 12, S. A Solla, T. K. Leen, and K.-R. Muller (Eds.), pp. 301–307. MIT Press, 2000.

[29] I.S. Dhillon, J. Fan, and Y. Guan, "Efficient clustering of very large document collections," as a chapter in *Data Mining for Scientific and Engineering Applications*, pp.357–381. Kluwer Academic Publishers, 2001.

[30] C. Elkan, "Using the triangle inequality to accelerate k-means," in *Proc. 20th Int. Conf. Machine Learning*, 2003.

[31] P. Smyth, K. Ide, and M. Ghil, "Multiple regimes in northern hemisphere height fields via mixture model clustering," *J. Atmospheric Science*, 56, 3704–3723. 1999.

[32] T. Mitchell, *Machine Learning*. McGraw Hill, 1997.

[33] F. Murtagh, "Contiguity-constrained hierarchical clustering," in *Partitioning Data Sets*, I.J. Cox, P. Hansen and B. Julesz, (eds.), DIMACS workshop, AMS (American Mathematical Society), 1995, pp. 143–152.

[34] J. C. Tilton, "Image segmentation by region growing and spectral clustering with a natural convergence criterion," in *Proc. Int. Geoscience and Remote Sensing Symp.*, Seattle, WA , 1998, pp. 1766–1768.

[35] N. Vivoy, "Automatic classification of time series (ACTS): a new clustering method for remote sensing time series," *Int. J. Remote Sensing*, vol. 21, pp. 1537–1560, 2000.

[36] http://earthobservatory.nasa.gov/Library/

Weighted Kernel K-Means for Clustering Spatial Data

[1]A. MAJID AWAN, [2]MOHD. NOOR MD. SAP, [3]M. OMAR MANSUR

Faculty of Computer Science and Information Systems
University Technology Malaysia
Skudai 81310, Johor
MALAYSIA
[1]awanmajid@hotmail.com, [2]mohdnoor@fsksm.utm.my, [3]mansurukm2@hotmail.com

Abstract: - This paper presents a method for unsupervised partitioning of data for finding spatio-temporal patterns in climate data using kernel methods which offer strength to deal with complex data non-linearly separable in input space. This work gets inspiration from the notion that a non-linear data transformation into some high dimensional feature space increases the possibility of linear separability of the patterns in the transformed space. Therefore, it simplifies exploration of the associated structure in the data. Kernel methods implicitly perform a non-linear mapping of the input data into a high dimensional feature space by replacing the inner products with an appropriate positive definite function. In this paper we present a robust weighted kernel k-means algorithm incorporating spatial constraints for clustering climate data. The proposed algorithm can effectively handle noise, outliers and auto-correlation in the spatial data, for effective and efficient data analysis.

Keywords: - Clustering, K-means, Kernel methods, spatial data, unsupervised learning

1 Introduction

Data clustering, a class of unsupervised learning algorithms, is an important and applications-oriented branch of machine learning. Its goal is to estimate the structure or density of a set of data without a training signal. It has a wide range of general and scientific applications such as data compression, unsupervised classification, image segmentation, anomaly detection, etc. There are many approaches to data clustering that vary in their complexity and effectiveness, due to the wide number of applications that these algorithms have. While there has been a large amount of research into the task of clustering, currently popular clustering methods often fail to find high-quality clusters.

A number of kernel-based learning methods have been proposed in recent years [3, 7, 8, 9, 16, 21]. However, much research effort is being put up for improving these techniques and in applying these techniques to various application domains. Generally speaking, a kernel function implicitly defines a non-linear transformation that maps the data from their original space to a high dimensional space where the data are expected to be more separable. Consequently, the kernel methods may achieve better performance by working in the new space. While powerful kernel methods have been proposed for supervised classification and regression problems, the development of effective kernel method for clustering, aside from a few tentative solutions [4, 9, 17], needs further investigation.

Finding good quality clusters in spatial data (e.g, temperature, precipitation, pressure, etc) is more challenging because of its peculiar characteristics such as auto-correlation, non-linear separability, outliers, noise, high-dimensionality, and when the data has clusters of widely differing shapes and sizes [11, 18, 22]. With this in view, the intention of this paper is, firstly, to analyze selective kernel-based clustering techniques in order to identify how further improvement can be made especially for spatial data clustering. Finally, we present a weighted kernel k-means clustering algorithm incorporating spatial constraints bearing spatial neighborhood information in order to handle spatial auto-correlation and noise in the spatial data.

This paper is organized as follows. In the next section, it is described how kernel-based methods can be useful for clustering non-linearly separable and high-dimensional spatial (climate) data. Two currently proposed kernel-based algorithms are briefly reviewed in section 3. In section 4, a weighted kernel k-means algorithm with spatial constraints is presented which can be useful for handling noise, outliers and auto-correlation in the spatial data. Some experimental results are given in

section 5. Finally in section 6, brief discussion and conclusion are given.

2 Kernel-based methods

The kernel methods are among the most researched subjects within machine-learning community in recent years and have been widely applied to pattern recognition and function approximation. Typical examples are support vector machines [2, 6, 20], kernel Fisher linear discriminant analysis [14], kernel principal component analysis [17], kernel perceptron algorithm [5], just to name a few. The fundamental idea of the kernel methods is to first transform the original low-dimensional inner-product input space into a higher dimensional feature space through some nonlinear mapping where complex nonlinear problems in the original low-dimensional space can more likely be linearly treated and solved in the transformed space according to the well-known Cover's theorem. However, usually such mapping into high-dimensional feature space will undoubtedly lead to an exponential increase of computational time, i.e., so-called curse of dimensionality. Fortunately, adopting kernel functions to substitute an inner product in the original space, which exactly corresponds to mapping the space into higher-dimensional feature space, is a favorable option. Therefore, the inner product form leads us to applying the kernel methods to cluster complex data [9, 15].

2.1 Support vector machines and kernel-based methods

Support vector machines (SVM), having its roots in machine learning theory, utilize optimization tools that seek to identify a linear optimal separating hyperplane to discriminate any two classes of interest [19, 20]. When the classes are linearly separable, the linear SVM performs adequately.

There are instances where a linear hyperplane cannot separate classes without misclassification, an instance relevant to our problem domain. However, those classes can be separated by a nonlinear separating hyperplane. In this case, data may be mapped to a higher dimensional space with a nonlinear transformation function. In the higher dimensional space, data are spread out, and a linear separating hyperplane may be found. This concept is based on Cover's theorem on the separability of patterns. According to Cover's theorem on the separability of patterns, an input space made up of nonlinearly separable patterns may be transformed into a feature space where the patterns are linearly separable with high probability, provided the transformation is nonlinear and the dimensionality of the feature space is high enough. Figure 1 illustrates that two classes in the input space may not be separated by a linear separating hyperplane, a common property of spatial data, e.g. rainfall patterns in a green mountain area might not be linearly separable from those in the surrounding plain area. However, when the two classes are mapped by a nonlinear transformation function, a linear separating hyperplane can be found in the higher dimensional feature space.

Let a nonlinear transformation function ϕ maps the data into a higher dimensional space. Suppose there exists a function K, called a kernel function, such that,

$$K(x_i, x_j) = \phi(x_i) \cdot \phi(x_j)$$

A kernel function is substituted for the dot product of the transformed vectors, and the explicit form of the transformation function ϕ is not necessarily known. In this way, kernels allow large non-linear feature spaces to be explored while avoiding curse of dimensionality. Further, the use of the kernel function is less computationally intensive. The formulation of the kernel function from the dot product is a special case of Mercer's theorem [16].

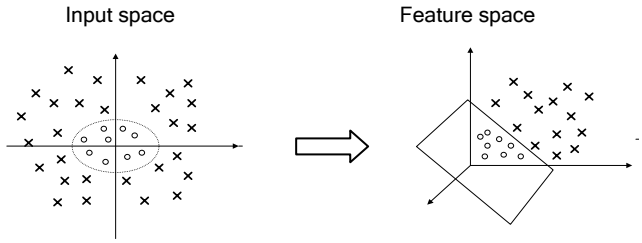


Fig. 1. Mapping nonlinear data to a higher dimensional feature space where a linear separating hyperplane can be found, eg, via the nonlinear map
$\Phi(x) = (z_1, z_2, z_3) = \left([x]_1^2, [x]_2^2, \sqrt{2}[x]_1[x]_2\right)$

Examples of some well-known kernel functions are given below in table 1.

TABLE 1. Some well-known kernel functions

Polynomial	$K(x_i, x_j) = < x_i, x_j >^d$	d is a positive *integer*
Radial Basis Function (RBF)	$K(x_i, x_j) = \exp(-\|x_i - x_j\|^2 / 2\sigma^2)$	σ is a user defined value
Sigmoid	$K(x_i, x_j) = \tanh(\alpha < x_i, x_j > + \beta)$	α, β are user defined values

3 K-means and kernel methods for clustering

Clustering has received a significant amount of renewed attention with the advent of nonlinear clustering methods based on kernels as it provides a common means of identifying structure in complex data [2, 4, 9, 15]. Before discussing two kernel-based algorithms [2, 4] here, the popular k-means algorithm is described in the next subsection, which is used as predominant strategy for final partitioning of the data.

3.1 K-means

First we briefly review k-means [12] which is a classical algorithm for clustering. We first fix the notation: let $X = \{ x_i \}_{i=1,\ldots,n}$ be a data set with $x_i \in R^N$. We call codebook the set $W = \{ w_j \}_{j=1,\ldots,k}$ with $w_j \in R^N$ and $k << n$. The Voronoi set (V_j) of the codevector w_j is the set of all vectors in X for which w_j is the nearest vector, i.e.

$$V_j = \{x_i \in X \,|\, j = arg \min_{j=1,\ldots,k} \|x_i - w_j\|\}$$

For a fixed training set X the quantization error $E(W)$ associated to the Voronoi tessellation induced by the codebook W can be written as:

$$E(W) = \sum_{j=1}^{k} \sum_{x_i \in V_j} \|x_i - w_j\|^2 \quad (1)$$

K-means is an iterative method for minimizing the quantization error $E(W)$ by repeatedly moving all codevectors to the arithmetic mean of their Voronoi sets. In the case of finite data set X and Euclidean distance, the centroid condition reduces to

$$w_j = \frac{1}{|V_j|} \sum_{x_i \in V_j} x_i \quad (2)$$

where $|V_j|$ denotes the cardinality of V_j . Therefore, k-means is guaranteed to find a local minimum for the quantization error. However, the k-means does not have mechanism to deal with issues such as:

- Outliers; one of the drawbacks of k-means is lack of robustness with respect to outliers, this problem can be easily observed by looking at the effect of outliers in the computation of the mean in eq. (2).
- non-linear separability of data in input space,
- auto-correlation in spatial data,
- noise, and high dimensionality of data.

3.2 One class SVM

Support vector clustering (*SVC*) [2], also called one-class SVM, is an unsupervised kernel method based on support vector description of a data set consisting of positive examples only. In SVC, data points are mapped from data space to a high dimensional feature space using a Gaussian kernel. In feature space, SVC computes the smallest sphere that encloses the image of the input data. This sphere is mapped back to data space, where it forms a set of contours, which enclose the data points. These contours are interpreted as cluster boundaries.

The clustering level can be controlled by changes in the width parameter of the Gaussian kernel (σ). The SVC algorithm can also deal with outliers by employing a soft margin constant that allows the sphere in feature space not to enclose all points.

Since SVC is using a transformation to an infinite dimension space, it can handle clusters of practically any shape, form or location in space. This is probably its most important advantage. However, the algorithm has the following drawbacks:

- One problem with the algorithm is its extreme dependence on σ. Finding the right value of σ is time-consuming and very delicate.
- Another disadvantage of the algorithm is its complexity. The separation of the sphere to different clusters and determining the adjacency matrix is extremely complicated.
- As the number of dimensions increases, the running time of the algorithm grows dramatically. For a large number of attributes, it is practically not feasible to use this algorithm.

3.3 Mercer kernel k-means

In [4], F. Camastra and A. Verri report on extending the SVC algorithm. The kernel k-means algorithm [4] uses k-means like strategy in the feature space using one class support vector machine. The algorithm can find more than one clusters. Although the algorithm [4] gives nice results and can handle outliers but it has some drawbacks:

- The convergence of this procedure is not guaranteed and is an open problem. The algorithm does not aim at minimizing the

quantization error because the Voronoi sets are not based on the computation of the centroids.

- The algorithm requires the solution of a quite number of quadratic programming problems, so takes heavy computation time.
- Because of the computational overheads, the algorithm might become unstable for high-dimensional data.
- Moreover, there is no mechanism for handling spatial auto-correlation in the data.

4 Weighted kernel k-means with spatial constraints

As we have illustrated above, there exist some problems in the k-means method, especially for handling spatial and complex data. Among these, the important issues/problems that need to be addressed are: i) non-linear separability of data in input space, ii) outliers and noise, iii) auto-correlation in spatial data, iv) high dimensionality of data. Although kernel methods offer power to deal with non-linearly separable and high-dimensional data but the current methods have some drawbacks as identified in section 3. Both [2, 4] are computationally very intensive, unable to handle large datasets and autocorrelation in the spatial data. The method proposed in [2] is not feasible to handle high dimensional data due to computational overheads, whereas the convergence of [4] is an open problem. With regard to addressing these problems, we propose an algorithm—weighted kernel k-means with spatial constraints, in order to handle spatial autocorrelation, noise and outliers present in the spatial data.

The k-means clustering algorithm can be enhanced by the use of a kernel function; by using an appropriate nonlinear mapping from the original (input) space to a higher dimensional feature space, one can extract clusters that are non-linearly separable in input space. Usually the extension from k-means to kernel k-means is realised by expressing the distance in the form of kernel function [16]. The kernel k-means algorithm can be generalized by introducing a weight for each point x, denoted by $u(x)$ [7]. This generalization would be powerful for making the algorithm more robust to noise and useful for handling auto-correlation in the spatial data. Using the non-linear function ϕ, the objective function of weighted kernel k-means can be defined as:

$$E(W) = \sum_{j=1}^{k} \sum_{x_i \in V_j} u(x_i) \left\| \phi(x_i) - w_j \right\|^2 \tag{3}$$

$$\text{where,} \quad w_j = \frac{\sum_{x_j \in V_j} u(x_j)\phi(x_j)}{\sum_{x_j \in V_j} u(x_j)} \tag{4}$$

Why weighting? As the weather station readings or agriculture related attribute values do not often represent equal areas, in such cases the use of weighted means instead of means becomes necessary. A 'weight' attributed to a mean or other value usually signifies importance. If weighting is not used, then each value that enters into a calculation of mean would have the same importance. Sometimes this is not a realistic or fair way to calculate a mean. Individual observations often, for various reasons, are of varying importance to the total, or average value.

The Euclidean distance from $\phi(x)$ to center w_j is given by (all computations in the form of inner products can be replaced by entries of the kernel matrix) the following eq.

$$\left\| \phi(x_i) - \frac{\sum_{x_j \in V_j} u(x_j)\phi(x_j)}{\sum_{x_j \in V_j} u(x_j)} \right\|^2 = K(x_i, x_i) - 2\frac{\sum_{x_j \in V_j} u(x_j)K(x_i, x_j)}{\sum_{x_j \in V_j} u(x_j)} + \frac{\sum_{x_j, x_l \in V_j} u(x_j)u(x_l)K(x_j, x_l)}{(\sum_{x_j \in V_j} u(x_j))^2} \tag{5}$$

In the above expression, the last term is needed to be calculated once per each iteration of the algorithm, and is representative of cluster centroids. If we write

$$C_k = \frac{\sum_{x_j, x_l \in V_j} u(x_j)u(x_l)K(x_j, x_l)}{(\sum_{x_j \in V_j} u(x_j))^2} \tag{6}$$

With this substitution, eq (5) can be re-written as

$$\left\| \phi(x_i) - \frac{\sum_{x_j \in V_j} u(x_j)\phi(x_j)}{\sum_{x_j \in V_j} u(x_j)} \right\|^2 = K(x_i, x_i) - 2\frac{\sum_{x_j \in V_j} u(x_j)K(x_i, x_j)}{\sum_{x_j \in V_j} u(x_j)} + C_k \tag{7}$$

For increasing the robustness of fuzzy c-means to noise, an approach is proposed in [1]. Here we propose a modification to the weighted kernel k-means to increase the robustness to noise and to account for spatial autocorrelation in the spatial data. It can be achieved by a modification to eq. (3) by introducing a penalty term containing spatial neighborhood information. This penalty term acts as

a regularizer and biases the solution toward piecewise-homogeneous labeling. Such regularization is also helpful in finding clusters in the data corrupted by noise. The objective function (3) can, thus, be written as:

$$E(W) = \sum_{j=1}^{k}\sum_{x_i \in V_j} u(x_i)\left\|\phi(x_i) - w_j\right\|^2 + \frac{\gamma}{N_R}\sum_{j=1}^{k}\sum_{x_i \in V_j} u(x_i)\sum_{r \in N_k}\left\|\phi(x_r) - w_j\right\|^2 \quad (8)$$

where N_k stands for the set of neighbors that exist in a window around x_i and N_R is the cardinality of N_k . The parameter γ controls the effect of the penalty term. The relative importance of the regularizing term is inversely proportional to the accuracy of clustering results.

For kernel functions, the following can be written

$$\left\|\phi(x_i) - w_j\right\|^2 = K(x_i, x_i) - 2K(x_i, w_j) + K(w_j, w_j)$$

If we adopt the Gaussian radial basis function (RBF), then $K(x, x) = 1$, so eq. (8) can be simplified as

$$E(W) = 2\sum_{j=1}^{k}\sum_{x_i \in V_j} u(x_i)(1 - K(x_i, w_j)) + \frac{\gamma}{N_R}\sum_{j=1}^{k}\sum_{x_i \in V_j} u(x_i)\sum_{r \in N_k}(1 - K(x_r, w_j)) \quad (9)$$

The distance in the last term of eq. (8), can be calculated as

$$\left\|\phi(x_r) - \frac{\sum_{x_j \in V_j} u(x_j)\phi(x_j)}{\sum_{x_j \in V_j} u(x_j)}\right\|^2 = 1 - 2\frac{\sum_{x_j \in V_j} u(x_j)K(x_r, x_j)}{\sum_{x_j \in V_j} u(x_j)} + \frac{\sum_{x_j, x_l \in V_j} u(x_j)u(x_l)K(x_j, x_l)}{(\sum_{x_j \in V_j} u(x_j))^2} \quad (10)$$

As first term of the above equation does not play any role for finding minimum distance, so it can be omitted, however.

$$\left\|\phi(x_r) - \frac{\sum_{x_j \in V_j} u(x_j)\phi(x_j)}{\sum_{x_j \in V_j} u(x_j)}\right\|^2 = 1 - 2\frac{\sum_{x_j \in V_j} u(x_j)K(x_r, x_j)}{\sum_{x_j \in V_j} u(x_j)} + C_k \quad = 1 - \beta_r + C_k \quad (11)$$

For RBF, eq. (5) can be re-written as

$$\left\|\phi(x_i) - \frac{\sum_{x_j \in V_j} u(x_j)\phi(x_j)}{\sum_{x_j \in V_j} u(x_j)}\right\|^2 = 1 - 2\frac{\sum_{x_j \in V_j} u(x_j)K(x_i, x_j)}{\sum_{x_j \in V_j} u(x_j)} + C_k \quad (12)$$

As first term of the above equation does not play any role for finding minimum distance, so it can be omitted.

We have to calculate the distance from each point to every cluster representative. This can be obtained from eq. (8) after incorporating the penalty term containing spatial neighborhood information by using eq. (11) and (12). Hence, the effective minimum distance can be calculated using the expression:

$$-2\frac{\sum_{x_j \in V_j} u(x_j)K(x_i, x_j)}{\sum_{x_j \in V_j} u(x_j)} + C_k + \frac{\gamma}{N_R}\sum_{r \in N_k}(\beta_r + C_k) \quad (13)$$

Now, the algorithm, weighted kernel k-means with spatial constraints, can be written as in Figure 2.

Algorithm SWK-means: spatial weighted kernel k-means (weighted kernel k-means with spatial constraints)

SWK_means (K, k, u, N, γ , ε)

Input: K: kernel matrix, k: number of clusters, u: weights for each point, set $\varepsilon > 0$ to a very small value for termination, N: information about the set of neighbors around a point, γ : penalty term parameter,

Output: w_1, ...,w_k: partitioning of the points

1. Initialize the k clusters: w_1=0, ... , w_k=0
2. Set $i = 0$.
3. For each cluster, compute $C(k)$ using expression (6)
4. For each point x, find its new cluster index as

   $j(x) = arg\,min_j \left\|\phi(x) - w_j\right\|^2$ using expression (13),
5. Compute the updated clusters as

   $w_j^{(i+1)} = \{x : j(x)=j\}$
6. Repeat steps 3-4 until the following termination criterion is met:

   $$\left\|W_{new} - W_{old}\right\| < \varepsilon$$

   where, $W = \{w_1, w_1, w_1, \ldots, w_k\}$ are the vectors of cluster centroids.

Fig. 2. Algorithm SWK-means: Spatial Weighted Kernel k-means (weighted kernel k-means with spatial constraints)

4.1 Handling outliers

This section briefly discusses about spatial outliers, i.e., observations which appear to be inconsistent with their neighborhoods. Detecting spatial outliers is useful in many applications of geographic information systems and spatial databases, including transportation, ecology, public

safety, public health, climatology, location-based services, and severe weather prediction. Informally, a spatial outlier is a local instability (in values of non-spatial attributes) or a spatially referenced object whose non-spatial attributes are extreme relative to its neighbors, even though the attributes may not be significantly different from the entire population.

We can examine how eq. (13) makes the algorithm robust to outliers. As $K(x_i, x_j)$ measures the similarity between x_i and x_j, and when x_i is an outlier, i.e., x_i is far from the other data points, then $K(x_i, x_j)$ will be very small. So, the second term in the above expression will get very low value or, in other words, the weighted sum of data points will be suppressed. The total expression will get higher value and hence results in robustness by not assigning the point to the cluster.

4.2 Scalability

The pruning procedure used in [23, 24] can be adapted to speed up the distance computations in the weighted kernel *k*-means algorithm. The acceleration scheme is based on the idea that we can use the triangle inequality to avoid unnecessary computations. According to the triangle inequality, for a point x_i, we can write, $d(x_i, w_j^n) \geq d(x_i, w_j^o) - d(w_j^o, w_j^n)$. The distances between the corresponding new and old centers, $d(w_j^o, w_j^n)$ for all *j*, can be computed. And this information can be stored in a $k \times k$ matrix. Similarly, another $k \times n$ matrix can be kept that contains lower bounds for the distances from each point to each center. The distance from a point to its cluster centre is exact in the matrix for lower bounds. Suppose, after a single iteration, all distances between each point and each center, $d(x_i, w_j^o)$, are computed. In the next iteration, after the centers are updated, we can estimate the lower bounds from each point x_i to the new cluster center, w_j^n, using $d(w_j^o, w_j^n)$ calculations and the distances from the previous iteration, i.e., we calculate the lower bounds as $d(x_i, w_j^o) - d(w_j^o, w_j^n)$. The distance from x_i to w_j^n is computed only if the estimation is smaller than distance from x_i to its cluster center. This estimation results in sufficient saving in computational time. Once we have computed lower bounds and begin to compute exact distances, the lower bound allows us to determine whether or not to determine remaining distances exactly.

5 Experimental Results

Given a data matrix, whose rows consists of time series from various points on the land (rainfall stations), the objective is to discover temporal and/or spatial patterns in the data. If we apply clustering algorithm to the rainfall time series associated with points on the land (surroundings of rainfall stations), we obtain clusters that represent land regions with relatively homogeneous behaviour. The centroids of these clusters are time series that summarize the bahaviour of those land areas.

For experimentation we selected 24 rainfall stations. A 12-month moving average is used for removing seasonality from the data. For monthly rainfall values for 5 years, we get a data matrix of 24×60. SWK-means partitioned it into 2 clusters. We also applied the algorithm to the monthly average rainfall values of this period, for easy visualization of results. Its results are shown in Figure 3. As the locations of rainfall stations are known, the clustering results can be easily mapped on the physical locations on the map. Actually the clusters will summarize the time series associated with relevant regions, and when results are plotted for a longer period, it will form some contiguous regions.

Since the kernel matrix is symmetric, we only keep its upper triangular matrix in the memory. For the next five year periods of time for the selected 24 rainfall stations we may get data matrices of 48×60, 72×60 and so on. The algorithm proportionally partitioned the data into two clusters. The corresponding results are shown in table 2 (a record represents 5-year monthly rainfall values taken at a station). It validates proper working of the algorithm.

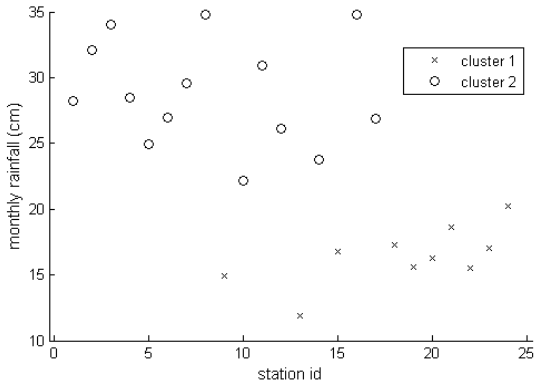


Fig. 3. Clustering results of SWK-means algorithm showing two clusters of monthly rainfall of 24 stations

TABLE 2. Results of SWK-means on rainfall data at 24 stations for 5, 10, 15, 20, 25, 30, 35 years

No. of Records	No. of records in cluster 1	No. of records in cluster 2
24	10	14
48	20	28
72	30	42
96	40	56
120	50	70
144	60	84
168	70	98

We use the clustering algorithm as a part of a software system for analyzing the impact of various hydrological and meteorological variables on the oil palm plantation [25]. It enables us to identify regions of the land whose constituent points have similar short-term and long-term characteristics. Given relatively uniform clusters we can then identify how various parameters, such as precipitation, temperature etc, influence the climate and oil-palm produce of different areas using correlation.

6 Discussion and conclusions

In this paper, a few challenges especially related to clustering spatial data are pointed out. There exist some problems that k-means method cannot tackle, especially for dealing with spatial and complex data. Among these, the important issues/problems that need to be addressed are: i) non-linear separability of data in input space, ii) outliers and noise, iii) auto-correlation in spatial data, iv) high dimensionality of data.

The strengths of kernel methods are outlined, which are helpful for clustering complex and high dimensional data that is non-linearly separable in input space. Two of the currently proposed kernel based algorithms are reviewed and the related research issues are identified. Both [2, 4] are computationally very intensive, unable to handle large datasets and have no mechanism to deal with autocorrelation in the spatial data. The method proposed in [2] is not feasible to handle high dimensional data due to computational overheads, whereas the convergence of [4] is an open problem. With regard to addressing these problems, we presented weighted kernel k-means incorporating spatial constraints. The proposed algorithm has the mechanism to handle spatial autocorrelation, noise and outliers in the spatial data. We are getting promising results on our test data sets. It is very much hoped that the algorithm would prove to be robust and effective for spatial (climate) data analysis. In future we plan to investigate the estimation of optimal number of clusters automatically.

References

[1] M.N. Ahmed, S.M. Yamany, N. Mohamed, A.A. Farag and T. Moriarty, "A modified fuzzy c-means algorithm for bias field estimation and segmentation of MRI data," *IEEE Trans. Medical Imaging*, vol. 21, pp.193–199, 2002.

[2] A. Ben-Hur, D. Horn, H. Siegelman, and V. Vapnik, "Support vector clustering," *J. Machine Learning Research*, vol. 2(Dec.), pp. 125–137, 2001.

[3] F. Camastra, "Kernel methods for unsupervised learning," PhD thesis, University of Genova, Italy, 2004.

[4] F. Camastra, A. Verri, "A novel kernel method for clustering," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 27, pp. 801-805, May 2005.

[5] J. H. Chen and C. S. Chen, "Fuzzy kernel perceptron," *IEEE Trans. Neural Networks*, vol. 13, pp. 1364–1373, Nov. 2002.

[6] N. Cristianini and J.S.Taylor, *An Introduction to Support Vector Machines*. Cambridge Academic Press, 2000.

[7] I.S. Dhillon, Y. Guan, and B. Kulis, "Kernel kmeans, spectral clustering and normalized cuts," in *Proc. 10th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, pp. 551–556, 2004.

[8] C. Ding and X. He, "K-means clustering via principal component analysis,"in *Proc. 21st Int. Conf. Machine Learning*, July 2004, pp. 225–232.

[9] M. Girolami, "Mercer kernel based clustering in feature space," *IEEE Trans. Neural Networks*, vol. 13(3), pp. 780–784, May 2002.

[10] R.M. Gray, *Vector Quantization and Signal Compression*. Kluwer Academic Press, Dordrecht, 1992.

[11] J. Han, M. Kamber and K. H. Tung, "Spatial clustering methods in data mining: a survey," in *Geographic Data Mining and Knowledge*

Discovery, Harvey J. Miller and Jiawei Han (eds.). Taylor and Francis, 2001.

[12] S.P. Lloyd, "An algorithm for vector quantizer design," *IEEE Trans. Communications*, vol. 28, no. 1, pp. 84–95, 1982.

[13] M.N. Md. Sap, and A. Majid Awan, "Finding spatio-temporal patterns in climate data using clustering," in *Proc. IEEE 2005 Int. Conf. on Cyberworlds*, Singapore, 23-25 November 2005, pp. 155–164.

[14] V. Roth and V. Steinhage, "Nonlinear discriminant analysis using kernel functions," in *Advances in Neural Information Processing Systems* 12, S. A Solla, T. K. Leen, and K.-R. Muller (Eds.). MIT Press, 2000, pp. 568–574.

[15] D.S. Satish and C.C. Sekhar, "Kernel based clustering for multiclass data," in *Proc. Int. Conf. on Neural Information Processing*, Kolkata, India, Nov. 2004.

[16] B. Scholkopf and A. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, 2002.

[17] B. Scholkopf, A. Smola, and K. R. Müller, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural Computation*, vol. 10, no. 5, pp. 1299–1319, 1998.

[18] S. Shekhar, P. Zhang, Y. Huang, and R. Vatsavai, "Trends in spatial data mining," as a chapter in *Data Mining: Next Generation Challenges and Future Directions*, H. Kargupta, A. Joshi, K. Sivakumar, and Y. Yesha (eds.). MIT Press, 2003.

[19] V.N. Vapnik, *The Nature of Statistical Learning Theory*. Springer-Verlag, New York, 1995.

[20] V.N. Vapnik, *Statistical Learning Theory*. John Wiley & Sons, 1998.

[21] L. Xu, J. Neufeld, B. Larson, and D. Schuurmans, "Maximum margin clustering," in *Proc. Advances in Neural Information Processing Systems* (NIPS2004), 2004.

[22] P. Zhang, M. Steinbach, V. Kumar, S. Shekhar, P-N Tan, S. Klooster, and C. Potter, "Discovery of patterns of earth science data using data mining," as a Chapter in *Next Generation of Data Mining Applications*, J. Zurada and M. Kantardzic (eds.). IEEE Press, 2003.

[23] I.S. Dhillon, J. Fan, and Y. Guan, "Efficient clustering of very large document collections," as a chapter in *Data Mining for Scientific and Engineering Applications*, pp.357–381. Kluwer Academic Publishers, 2001.

[24] C. Elkan, "Using the triangle inequality to accelerate k-means," in *Proc. 20th Int. Conf. Machine Learning*, 2003.

[25] A.M. Awan and M.N. Md. Sap, "An intelligent system based on kernel methods for crop yield prediction," in *Proc. 10th Pacific-Asia Conference on Knowledge Discovery and Data Mining* (PAKDD 2006), 9-12 April 2006, Singapore, pp. 841–846.