

UNIVERSITY OF CALGARY

Hybrid Mining Financial Time Series

by

Shang Gao

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE
DEGREE OF MASTER OF SCIENCE

DEPARTMENT OF COMPUTER SCIENCE

CALGARY, ALBERTA

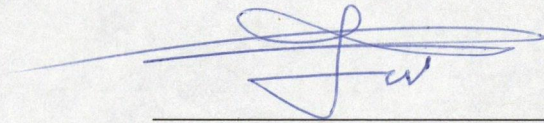
September, 2009

© Shang Gao 2009

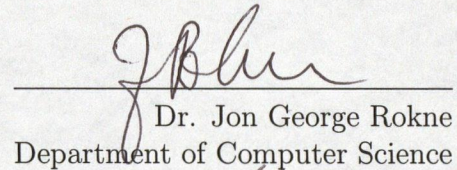
UNIVERSITY OF CALGARY

FACULTY OF GRADUATE STUDIES

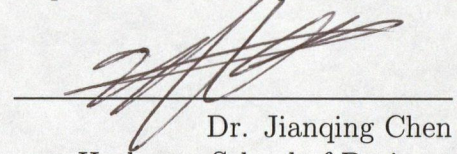
The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies for acceptance, a thesis entitled "Hybrid Mining Financial Time Series" submitted by Shang Gao in partial fulfillment of the requirements for the degree of MASTER OF SCIENCE.



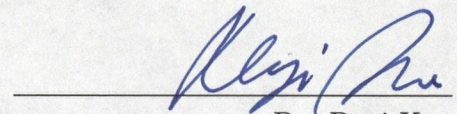
Supervisor, Dr. Reda Alhajj
Department of Computer Science



Dr. Jon George Rokne
Department of Computer Science



Dr. Jianqing Chen
Haskayne School of Business



Dr. Deyi Xue
Department of Mechanical and
Manufacturing Engineering

Date

Abstract

Knowledge discovery in financial datasets has important implications. In deed decision making process on financial datasets is known to be difficult because of the complex knowledge domain and specific statistical characteristics of the data. In this thesis, we investigate the decision making problem on financial time series datasets such as stock market fluctuations by means of statistical modeling while maintaining the interpretable results based on association rules discovered using the *Rough Set* computation and fuzzy discretization. As an alternative, we approach the data mining process by integrating different categories of financial ratios as input to the Rough Set model. Two stepwise forecasting procedures are proposed followed by the experimental results through case studies and simulated datasets. The two main contributions of the thesis are the successful application of efficient and effective data mining techniques to the financial domain and the development of a user friendly model that benefits and guides individual investors in making investment decisions.

Acknowledgements

I thank Dr. Reda Alhajj for the constant support and guidance throughout my Master's research. The project was exclusively supervised by Dr. Alhajj, without whom the research could not be accomplished.

I also thank Dr. Jianqing Chen, Dr. Jon Rokne and Dr. Deyi Xue, my committee members, for their time and effort in reviewing this work and their valuable feedback.

I would like to thank my parents and Dr. Jiwen Guan who had helped me during the past.

Table of Contents

Approval Page.....	ii
Abstract.....	iii
Acknowledgements.....	iv
Table of Contents.....	v
List of Tables	vii
List of Figures and Illustrations	viii
List of Symbols, Abbreviations and Nomenclature.....	ix
 CHAPTER 1 INTRODUCTION	 1
Problem Scope and Motivation.....	1
The Proposed Solution.....	4
Contribution	7
Thesis Organization	7
 CHAPTER 2 BACKGOURND AND RELATED WORK	 9
Related Work	9
Basics of the Rough Set theory.....	13
Statistical Modeling	16
Characterizing Financial Time Series	16
Computational Domain	17
Autoregressive Model (AR) and GARCH Model	18
Fuzzy Time Series	20
Data Mining Techniques.....	22
Association Rule.....	22
K-means clustering and classification concepts	23
 CHAPTER 3 MINING PROCEDURES	 28
Statistical Modeling of Fuzzy Parameters	28
Financial Ratios	31
Forecasting Procedure.....	32
An Alternative Approach.....	39
The alternative decision feature.....	39
Alternative Steps 4 and 5.....	40
 CHAPTER 4 EXPERIMENTAL ANALYSIS	 42
Datasets	42
Case Study I: NASDAQ Composite Index.....	43
Case Study II: Japanese Nikkei Average Index.....	52
Synthetic & non-scaled Data	54
Alternative Procedure	58
Comparative Study	61

CHAPTER 5 SUMMARY, CONCLUSIONS AND FUTURE WORK	64
Summary and Conclusions	64
Future Work.....	65
REFERENCES	68

List of Tables

Table 2.1: A sample decision table	15
Table 2.2: A sample database of transactions	23
Table 3.1: Liquidity, capital structure and solvency ratios	32
Table 3.2: ROI, profitability and earning ratios	32
Table 4.1: NASDAQ Composite Index from Dec. 1971 with normal prices	43
Table 4.2: Summary of features added for NASDAQ dataset	43
Table 4.3: Fuzzified features and clustering results of NASDAQ Composite Index	48
Table 4.4: Experimental results with NASDAQ Index data using training dataset from 1972 to 1992 to forecast various periods	50
Table 4.5: Experimental results with NASDAQ Index data using training dataset from 1972 to 1992 and testing dataset from 2000 to 2008 individually	50
Table 4.6: Experimental results with NASDAQ Index data using different amount of training dataset and testing dataset from 1993 to 2008 individually	51
Table 4.7: Experimental results with Japan Nikkei Average Index data	54
Table 4.8: Synthetic datasets	55
Table 4.9: Input data to the Rough Set model with simulated financial ratios	59
Table 4.10: Performance of Rough Set and Apriori with 2G RAM Pentium M	61

List of Figures and Illustrations

Figure 1.1: A sample time series dataset from Google Finance: CSCO from July 28, 2008 to June 27, 2009	4
Figure 1.2: System work flow for mining financial time series data.....	6
Figure 2.1: A sample membership function.....	21
Figure 3.1: NASDAQ Composite Index from Feb. 1971 to Apr. 2009 normal prices and returns.....	34
Figure 3.2: Decision features for the alternative procedure.....	41
Figure 4.1: Fuzzy membership functions for the asset return feature.....	45
Figure 4.2: ACF computations showing a range of lagging factors	47
Figure 4.3: Cluster plot with output of <i>determine-K(data)</i> algorithm	47
Figure 4.4: Sample association rules found with NASDAQ data.....	49
Figure 4.5: Cluster plot for Nikkei Average Index dataset.....	53
Figure 4.6: Sample of a non-scaled dataset	56
Figure 4.7: 10 Forecasting results with 4 different datasets	57
Figure 4.8: Returns of the original financial time series.....	59
Figure 4.9: Forecasting results for different training-testing data split in percentage with confidence $\geq 60\%$	60
Figure 4.10: Retrieval of financial information in <i>SAP Business One</i>	62
Figure 5.1: The decision making process	66

List of Symbols, Abbreviations and Nomenclature

Symbol

ERP

UoD

sup

conf

VAT

ACF

Definition

Enterprise Resource Planning

Universe of Discourse

Support of an association rule

Confidence of an association rule

Value Added Tax

Autocorrelation function

CHAPTER 1

INTRODUCTION

In this thesis two methods are developed for extracting trends from time varying financial data such as the NASDAQ average sampled on a yearly basis. The methods use techniques from data mining, statistics and generalized set theories.

Problem Scope and Motivation

Data mining aims at finding hidden patterns or rules in datasets. Examples are the data mining of RNA sequences [17] and the data mining to discover trends in financial time series datasets [5, 6, 41]. Data mining is a subject that combines both mathematical models and the information theory [12, 13, 33, 34]. It has a large number of practical applications since knowledge or patterns found during from a well-designed data mining procedure can provide important information for the general analysis of a given dataset.

The early work in data mining was mostly in the area of association rule mining. By this it meant the finding of frequent items or sets of items in a database of commercial transactions such as grocery purchases [3, 50]. The research related to association rule mining was initiated by Agarwal et al. [3], which generated a wider interest in data mining and might therefore be considered as the founding paper for data mining research. Earlier clustering and classification were applied to data mining tasks, but they were techniques developed for other areas of research such as pattern recognition. The *Apriori* algorithm of Agrawal et al. is considered to be the pioneer algorithm for association rules mining. Following this algorithm, the algorithms *Eclat* by Zhaki et al. [52], *FP-growth*

by Han et al. [19] and several other algorithms [18] were developed aiming at improving the efficiency of the association mining process. All of these association rule mining algorithms are widely used for discovering patterns in datasets. The notion of association rule mining has also been extended to the areas of spatial and temporal data mining [35].

Time series datasets and corresponding data mining techniques have become more important in recent data mining research in the context of pattern recognition [15, 30, 39, 45, 47]. The time element in such datasets must be considered carefully since the time ordering significantly affects the data mining operations. The common data mining techniques therefore must be adapted for the data mining of time series. The various domains of the datasets also require special treatment.

The data mining of financial datasets have important applications when the results are embedded into real world applications such as ERP systems like *SAP Business One* and *Microsoft Navision*.

Financial datasets are probably the most important example of time varying data and the algorithms developed for the static data mining tasks such as association rule mining algorithms have to be extended for the time variation. Also, when considering the financial data from stock market, the stock symbols are associated with values such as open, close, daily high/low, bid and ask values, etc. The data mining of financial data is therefore a challenging task, not only because of the complications in the computer modeling of the problem, but also the domain specifics of such datasets. In other words, the domain knowledge needs to be investigated in advance before applying any data mining techniques to discover the hidden knowledge [21, 40, 43]. In the context of mining financial time series data, hidden knowledge includes periodicity discovery [15],

i.e., to find the set of values occurring in certain periodic pattern; value prediction [6, 26, 41, 45] , i.e., to predict unseen values based on the existing time series datasets; outlier detection [30, 39], i.e., to find extremely abnormal data points based on the existing data patterns, and data visualization [20], i.e., to visualize knowledge based on certain characteristics of the datasets facilitating outsiders in making decisions. Several tools are discussed in [1, 25] using several different mining techniques. However, those tools do not include the statistical significance of the financial time series data and they are also not tractable for individual investors.

Data mining of financial datasets involves both theoretical and empirical knowledge. Traditionally such data sets have been modeled with using autoregressive and moving average models, examples can be found in Langdell's article [54]. In this thesis, we intend to investigate financial time series data from a new perspective, that is, incorporating *Rough Set* computations and fuzzy discretizations into the data mining techniques in order to be able to analyze financial time series more realistically. Example of applications of the new approach is the extraction of useful knowledge from historical stock datasets with the aid of applicable domain knowledge and statistical measurements. The aim is to be able to make a user friendly solution framework that benefits individual investors in marking investment decisions. The approach will enable the use of financial time series data mining for extracting knowledge hidden in the datasets such that investment decisions can be made with some probability of success. For example, Figure 1.1 shows the stock time series dataset for Cisco Systems, Inc. from July 28, 2008 to June 27, 2009 with net value decreased -0.67 (-2.98%). From Figure 1.1, we can see that the stock value fluctuated during the period of interest. The investigation using the proposed

tool might be able to explain the trends in the data from the patterns found. From this the knowledge acquired might be usable for the prediction of future values and trends.

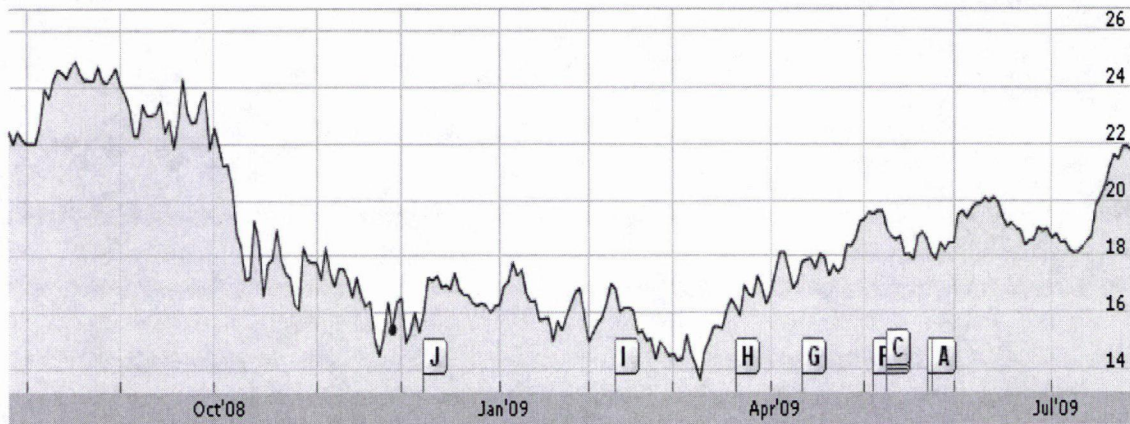


Figure 1.1: A sample time series dataset from Google Finance: CSCO from July 28, 2008 to June 27, 2009

This thesis presents two procedures for discovering hidden knowledge in financial time series datasets using *Rough Set* theory and *Fuzzy Set* theory. The procedures can be used for financial forecasting and they might play important roles in Business Intelligence software and systems.

The Proposed Solution

In this thesis, we address the problem of achieving the efficiency and effectiveness in mining financial time series datasets by means of:

1. Statistical measurements while maintaining the interpretable results. We solve this problem by using two statistical models, namely the *Autoregression* model and the *GARCH* model in pre-processing the time series datasets, and we use the error gauge and volatility measurements in finding patterns or hidden rules. We present

a step-wise procedure for mining the financial time series datasets using this hybrid methodology.

2. Manually appending domain specific features. In other words, we establish the connections between stock fluctuations the investors see everyday in the financial market and the firms' audited financial statements. This helps the investors to understand the trends in the data.

The system workflow is shown in Figure 1.2. In the solution framework, we use four layers in mining the financial time series datasets which are known to be statistically sensitive in nature. The layers are detailed below:

1. Feature acquisition and model fitting layer – this layer focuses on the selection of useful information in forming the *Information System*, i.e., *features* can be used to form the *Information System* defined in the Data Mining stage in the context of the *Rough Set* computation.
2. Fuzzy pre-processing layer – this layer performs the fuzzy discretization process in further preparation of the knowledge discovery phase in terms of the *Fuzzy Set* definitions and the fuzzy membership functions. This pre-processing is advantageous in interpreting the rule-based results.
3. Data mining layer – this layer performs the core knowledge discovery function using mathematical models and data mining techniques. In this thesis, we use the *Rough Set* theory in finding association rules and the unsupervised learning technique *K-means* clustering in formulating the hidden knowledge. The possible fine-tuning of the output rules is also done in this layer.

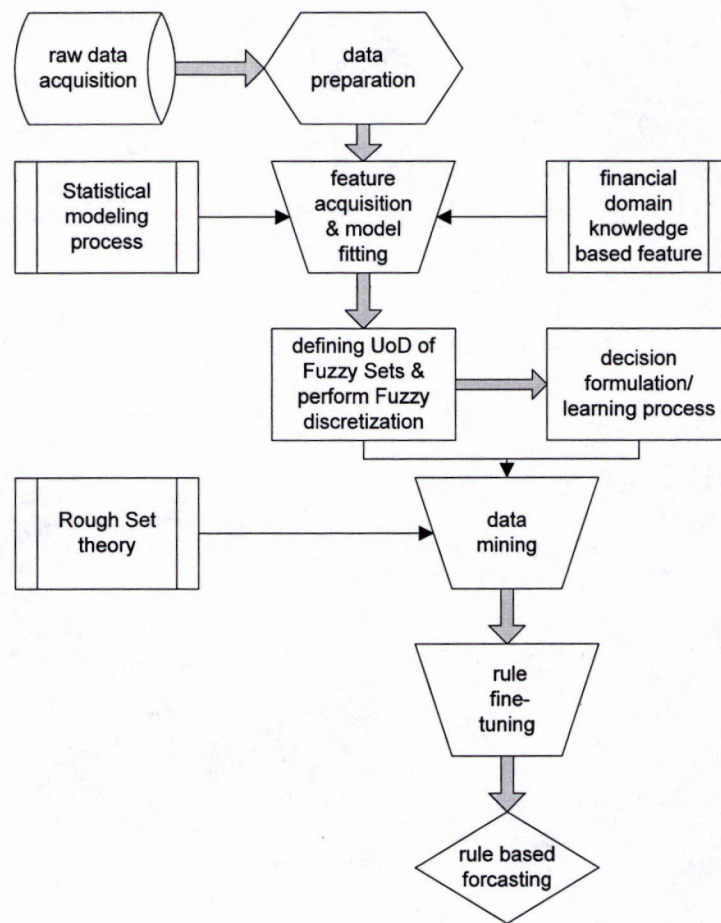


Figure 1.2: System work flow for mining financial time series data

4. Forecasting layer – this layer performs the forecasting task based on the rules and patterns found in the data mining layer. This stage is particularly useful for end-users who are not equipped with domain specific knowledge such as the mathematical models in the mining analysis.

Contribution

The proposed hybrid mining approach can be used to model solutions in finding hidden patterns in financial time series datasets and further perform forecasting tasks. The contributions of my research as described in this thesis are summarized below:

1. Incorporate the well-known domain knowledge (in this investigation, financial indicators or ratios) to gauge the economical situations in association with making investment decisions such as to buy or sell securities or traded shares. This is an important application domain since the framework can be standalone or embedded into real time systems, either at the enterprise level or stock management systems.
2. We incorporate statistical models in deriving domain specific features besides financial ratios that are sometimes difficult to obtain. This cross-disciplinary approach improves the accuracy of the mining result and can be further extended to address other applicable domains.
3. We have achieved efficiency in the hybrid mining procedure using powerful mathematical tools and data mining techniques.

Thesis Organization

To summarize, in this chapter we described the problem tackled in this thesis and the motivations based upon the real world application domain, i.e., the financial time series datasets. The system workflow is presented and the main contributions of this work are listed.

The rest of this document is organized in four chapters. Chapter 2 introduces the related work of the data mining research in the area followed by the basics of the *Rough Set* theory, the *Fuzzy Set* theory and the Data Mining techniques that are used in the mining process described in Figure 1.2; statistical models are presented, which are used to derive system parameters as input to the *Rough Set* computation.

In Chapter 3, we present a hybrid mining procedure that deals with financial time series datasets. We also discuss a different approach of manually appending features with pre-selected financial ratios.

In Chapter 4, we report and discuss the experimental results of the hybrid mining procedures described in Chapter 3. We use real-time historical datasets in evaluating the approach as well as simulated datasets in gauging the effectiveness and limitations of the approach. Case studies are conducted in the context.

Chapter 5 summarizes and concludes the thesis and discusses the future directions of the research.

CHAPTER 2

BACKGOURND AND RELATED WORK

This chapter lays out the background knowledge of the related research, and covers the basics of the *Rough Set* theory, *Fuzzy Set* theory and Data Mining techniques that are used in the hybrid mining procedure described in Figure 1.2; two statistical models, *Autoregression* and *GRACH*, are presented; both are used to derive system parameters as input to the *Rough Set* computation.

Related Work

Pattern recognition and knowledge discovery is a field of machine learning within Artificial Intelligence [18]. Data mining is a relatively new term that combines the algorithm-wise analysis and mathematical computations to find hidden knowledge in the data. In the scope of this thesis, several related concepts and techniques have been widely used in mining financial time series data, depending on the domain specifications of the datasets to be dealt with.

The *Rough Set* theory has been widely employed to find temporal rules and further perform classifications on the datasets [16, 29, 37]. The main advantages of using the *Rough Set* theory in data mining are the efficiency and completeness towards the mining results. Its completeness is acknowledged by the mathematical computation models. For example, in [16], the powerful approach proposed is to use the *Rough Set* theory to find the maximal frequent itemsets instead of using the *Apriori* algorithm, and the approach is more efficient with fewer dataset scans.

In the context of mining time series data, several segmentation techniques are introduced to pre-process the time series data in general, including the *top-down* algorithm, the *bottom-up* algorithm and the *sliding window* algorithm [32]. This category of approaches essentially reveals the dynamic nature of most time series datasets, therefore is worth considering in case of mining financial time series data in a broad perspective. However, the approaches overlook some specifics of the data domain, for example, because the sensitivity of the financial time series data is crucial in most cases [43], those segmentation techniques have to be tailored to adapt for the financial domain.

Similarity retrieval approaches are proposed in [32, 48] in terms of distance measurements and threshold comparison. Similarity measurement has a great impact on pattern and periodicity discovery in the time series data, and is useful in a large number of application domains. Time series data compression and indexing techniques [32] can be seen as the continuation of similarity retrieval methods, as they measure the compressed datasets that preserve the terrain of the original datasets, and then index the time series data in a way that facilitates pattern and periodicity discovery.

Domain specific research has been present, such as the candlestick measurements of the stock data in [26]. This approach facilitates the interpretation of the mining results with fuzzy translations. The *Fuzzy Set* theory was originally invented as the extension of the probabilistic theory to handle the concept of partial truth which could be anywhere between “completely true” and “completely false” [4, 42, 49] and accordingly the membership probability varies between 0 and 1 depending on whether the value is close to “completely true” or “completely false”, respectively. So, in the *Fuzzy Set* theory, a *membership function* is used to categorize data to a certain belief degree. The *Fuzzy Set*

approach has become more and more popular in recent data mining research since it incorporates human logic in reasoning process and the mined results. However, the candlestick theory cannot be used to achieve establishing connections between stock prices and enterprises' financial situations; it rather focuses on the stock variations. The symbolic representation of the time series data is also useful in dealing with stream algorithms and data encoding [28].

Some researchers have focused on mining the model of products' lifecycle as running state data [38]. A data management system based on product running state is introduced with five different layers: basic data layer, data extraction layer, data store layer, information extraction layer (data mining) and data visualization layer. In our investigation, the layered architecture is used to devise a systematic procedure as shown in Figure 1.2. In [38], running state data mining approaches are presented, such as correlation analysis and advantage analysis using the relevance matrix that involves products running factors. Time series data mining is also included in conjunction with *Grey* theory.

The branch of mining sequential patterns can be seen as an instance of mining time series data, as in sequential datasets, time ordering is also an important factor. Given a customer transactional database, where we have transactions made by different customers in different time periods, the central task is to find sequential patterns over the data accumulated sequentially [2, 11, 51, 52]. This can be seen as the extension of the static association rule mining problem solved by *Apriori* algorithm [3]; the so-called *Apriori(All/Some)* algorithms are to address such sequential mining problems. The main drawback of the *Apriori(All/Some)* and *Apriori* algorithm is the efficiency concern: for

sequential mining algorithms like *Apriori(All/Some)*, pre-processing and transformation are required for time series datasets. *Apriori(All/Some)* algorithms are rather costly mainly because of the transformation step. Another disadvantage of this category of approaches is that the overhead of the huge candidate set generation and consequent database scanning for ordered lists of itemsets. *AprioriSome* algorithm alleviates this problem by attempting to balance between making frequent databases scans and finding the maximality of large sequences. *ApriorSome* algorithm overlooks the “usefulness” of data, in other words, in most circumstances users would want a complete list of large sequences regardless of maximality. On the other hand, *DynamicSome* algorithm generates the candidate set on-the-fly, the algorithm has three phases, an intermediate phase in addition to *AprioriSome*, and the performance issue with large databases is still present.

The well-known *FP-growth* algorithm in finding frequent patterns [19] causes the memory and space concern when dealing with large databases, which often is the case for customer transactional databases and time series datasets in general. The time variant in the sequential mining problem can be dealt with using the time series mining techniques described in [23, 48].

The set based approach is more efficient in general in mining the sequential patterns. *SPADE* (Sequential PAttern Discovery using Equivalence classes) algorithm [52] makes the effort to achieve efficiency while preserving the “usefulness” of data. The approach maintains the “ID-Lists” for sequences and the larger sequences can be derived by intersecting lower level ID-lists. The algorithm introduces the *lattice equivalence*

classes in order to address the memory space issues when mining large databases. The large sequences can then be found in each equivalence class.

Basics of the Rough Set theory

The *Rough Set* theory, introduced by Z. Pawlak (1991), is a relatively new and powerful mathematical tool to discover knowledge in data. *Rough Set* theory has been studied by many researchers in computer science and mathematics. After it was introduced, researchers have worked on the theoretical side of the *Rough Set* theory, the main focuses are reasoning under uncertainty with associated rules, reducts searching and relation/space approximations for information systems, etc.

In modern ERP systems, the database often contains a large number of attributes. The abstract view of such relations is the so-called the *decision table*, where attributes can be divided into two groups: *condition variables* and *decision variables* with the relationship that each conditional variable has different degree of support to the decision variable. The minimal set of condition variables that determines the decision variables, also referred as the “core” conditional variables, is called the *key set*. *Rough Set* theory provides such reasoning rules that allow us to find the *key set* in the decision table as well as attributes associations and dependencies, which reveal knowledge in the original decision table. This knowledge can have great influence in financial databases in that external reports can be generated efficiently and implementations of internal analytical modules can be eased.

Our goal is to systematically explore the *Rough Set* theory and apply it to the real-world financial time series datasets. Emphasis will be given to the effectiveness of the approach in the process of problem solving and knowledge discovery.

In the *Rough Set* theory, an *Information System* is used to represent the knowledge. An *Information System* $S = (U, \Omega, V_q, f_q)$ consists of:

- U - A nonempty, finite set of objects called the *universe*.
- Ω - A nonempty, finite set of attributes that can be classified into two groups: *condition attributes (C)* and *decision attributes (D)*. Both C and D are finite.
- V_q - Domain of q for each q in Ω .
- f_q - A mapping function or information function that maps each object in the *universe* to the domain of the object, that is, $f_q : U \rightarrow V_q$.

The abstract view of the *Information System* defined above is called the *Decision Table*. A sample decision table is shown in Table 2.1, where columns present attributes and rows represent objects or observations. In the context of data mining, *condition attributes (C)* and *decision attributes (D)* are often referred as *features*. In such context, attributes are commonly referred to as *features* of the system.

In an Information System, a *P-equivalence* relation or *P-indiscernibility* relation is defined as:

$$IND(P) = \{(x, y) \in U^2 \mid \forall a \in P, a(x) = a(y)\}$$

Table 2.1: A sample decision table

U	a	b	c
u_1	3	5	4
u_2	0	1	0
u_3	2	1	3
u_4	0	3	2
u_5	0	1	3

A partition of U generated by $IND(P)$ is defined as:

$$U / IND(P) = \otimes \{U / IND(a) \mid a \in P\}, \text{ where}$$

$$A \otimes B = \{X \cap Y \mid \forall X \in A, \forall Y \in B, X \cap Y \neq \emptyset\}$$

To illustrate this, consider the Information System in Table 2.1 and let

$P = \{a, b\}$, then

$$IND(a) = \{\{u_1\}, \{u_2, u_4, u_5\}, \{u_3\}\}$$

$$IND(P) = \{\{u_1\}, \{u_2, u_5\}, \{u_3\}, \{u_4\}\}$$

In order to approximate the equivalent classes, *lower approximation* and *upper approximation* are defined as follows: for a decision table $S = (U, \Omega, V_q, f_q)$

where $P \subseteq \Omega, X \subseteq U$:

$$\underline{P}X = \cup \{Y \in U / IND(P) \mid Y \subseteq X\}$$

$$\bar{P}X = \cup \{Y \in U / IND(P) \mid X \cap Y \neq \emptyset\}$$

$$PN_p(X) = \bar{P}X - \underline{P}X$$

For example, consider the Information System in Table 2.1, and let $P = \{a, b\}$ and

$X = \{u_2, u_3\}$ then

$$\underline{P}X = \{u_3\}; \bar{P}X = \{u_2, u_5\} \cup \{u_3\}; PN_p(X) = \{u_2, u_5\}$$

To measure the quality of lower and upper approximations, we define:

$$\underline{\gamma}_P = \frac{|PX|}{|U|}, \quad \overline{\gamma}_P = \frac{|\overline{P}X|}{|U|}$$

The quality of lower approximation of X by a set of features P is the ratio of the number of all certainly classified objects to the total number of objects in the universe; The quality of upper approximation of X by a set of features P is the ratio of the number of all possibly classified objects to the total number of objects in the universe.

Statistical Modeling

Characterizing Financial Time Series

A time series $\{Y_t\}$ is a process where $t=1, \dots, T$ are discrete data points. The process must satisfy:

1. t is discrete in time;
2. The process is continuous in time. In our investigation, t is regarded as equally spaced time intervals since most of the financial time series datasets fulfill this condition. In this statistical definition, the time-variant nature is preserved.

A time series is said to be *weakly stationary* if both (1) and (2) are satisfied:

$$E[X_t] = \mu \text{ for all } t \quad (1)$$

$$\text{Cov}(X_t, X_{t-k}) = E[(X_t - \mu)(X_{t-k} - \mu)] = \gamma_k \text{ for all } t \text{ and any } k \quad (2)$$

In finance literature, it is reasonable to assume that some correlations exist in the evaluated dataset, that is, we assume the datasets are *stationary*. Further, the behaviour in financial time series data renders certain patterns empirically, as it reflects economy performance in the real world that is known to follow certain patterns to some extent.

Computational Domain

In the literature of computational finance, the *asset return* is deemed to have more statistical meaning than normal financial prices. Asset returns are scale-free. This means that return is scaled regardless of normal prices. Therefore, the asset returns are comparable in different financial markets. This can reduce the ambiguity on normal price noises which affect the accuracy of prediction. Furthermore, assets returns reveal more statistical properties. For example, the return of certain assets can be annualized for comparison if multi-period computations are in need.

The simple one-period asset return is defined as:

$$1 + R_t = \frac{P_t}{P_{t-1}}$$

where P_t is the asset price in time period t ; P_{t-1} is the asset price in previous time period $t-1$; R_t is the one-period *simple net return* and $1 + R_t$ is called the *simple gross return*. By re-arranging the equation components, R_t can be written as:

$$R_t = \frac{P_t - P_{t-1}}{P_{t-1}} \quad (3)$$

If we hold the asset for k periods, that is, between time $t-k$ and t , the simple gross return is then:

$$\begin{aligned} 1 + R_t[k] &= \frac{P_t}{P_{t-k}} = \frac{P_t}{P_{t-1}} \times \frac{P_{t-1}}{P_{t-2}} \times \dots \times \frac{P_{t-k+1}}{P_{t-k}} = (1 + R_t)(1 + R_{t-1}) \dots (1 + R_{t-k+1}) \\ &= \prod_{j=0}^{k-1} (1 + R_{t-j}) \end{aligned}$$

In the above formula we can see that the k -period simple gross return is the product of k one-period simple returns.

If the stationary time series data satisfies (1) and (2), the *Autocorrelation* function *ACF* is defined as:

$$\rho_k = \frac{Cov(X_t, X_{t-k})}{\sqrt{Var(X_t)Var(X_{t-k})}} = \frac{\gamma_k}{\gamma_0} \quad (4)$$

The value k is called the *lag*. The correlation between X_t and X_{t-k} is called the *lag- k autocorrelation*.

The ACF function is important because we are assuming the time series is stationary satisfying $\gamma_{-l} = \gamma_l$. By the definition, we know that $0 \leq \rho_k \leq 1$. If $\rho_k = 0$ for all k , we know that the time series is not correlated; If $\rho_k = 1$ for some k , we know that the time series is strongly correlated with lag k . By means of computing autocorrelations, we can informatively characterize the data series.

Autoregressive Model (AR) and GARCH Model

In the analysis of financial time series datasets, it is often the case that the financial data is influenced by several factors. The *AR* model is particularly useful in modeling influential factors. The simplest first order AR model, denoted as *AR(1)* is defined as:

$$Y_t = \mu + \alpha Y_{t-1} + \varepsilon_t \quad (5)$$

where μ is the same as in (1); $|\alpha| < 1$ is the model parameter that weighs influencing factors; ε_t is the random variable independent of time t . In the finance literature, ε_t is referred to as *white noise* where $E(\varepsilon_t) = 0$ and $Var(\varepsilon_t) = \sigma^2$.

The equation (5) is called $AR(1)$ because Y_t is dependent or explained only by Y_{t-1} . In general $AR(n)$ model takes into account previous $t-n$ states in predicting the current state. Because of the stationary nature of AR models, some important properties exist:

$$E(Y_t) = \frac{\mu}{1-\alpha} \quad (6)$$

$$Var(Y_t) = \frac{\sigma^2}{1-\alpha^2} \quad (7)$$

Where σ captures the standard deviation of the white noise ε_t .

While AR model is suitable for finding direct linear relationships in the time series data assuming the stationarity, we are also interested in measuring the incremental change, referred to as the *volatility clustering* of the time series. The $GARCH$ model is designed for this purpose. In general, the $GARCH$ model is defined as:

$$Y_t = \mu + \alpha Y_{t-1} + \varepsilon_t$$

$$\varepsilon_t \sim N(0, \sigma_t^2)$$

Where all parameters are defined in the same way as in AR models except that now

$$\sigma_t^2 = a_0 + a\varepsilon_{t-1}^2 + b\sigma_{t-1}^2 \quad (8)$$

Where $0 \leq a, b \leq 1$ and $a+b \leq 1$.

In $GARCH(1, 1)$ model, we say that the process is stationary if $a+b < 1$. The stationary variance is given by

$$\frac{a_0}{1-a-b} \quad (9)$$

Fuzzy Time Series

The application of the fuzzy time series is presented in [4, 28, 49], and the concept of which has been widely deployed in numerous settings such as in university enrolment forecasting and TAIEX forecasting [26]. The fuzzy computations are useful in pre-processing or discretizing datasets. We convey the basics of the *Fuzzy Set* theory in this section.

Denote UoD as the *universe of discourse*, and let $U = \{x_1, x_2, \dots, x_n\}$, a *fuzzy set* A_i of U is defined by:

$$A_i = \mu_{A_i}(x_1)/x_1 + \mu_{A_i}(x_2)/x_2 + \dots + \mu_{A_i}(x_n)/x_n$$

Where $\mu_{A_i} : U \rightarrow [0, 1]$ is so-called the *membership function* of the fuzzy set A_i . x_k is a value to be fuzzified; $\mu_{A_i}(x_k) \in [0, 1]$ is the degree of membership of x_k to the fuzzy set A_i for $1 \leq k \leq n$. A sample membership function is shown in Figure 2.1.

The fuzzy time series is frequently used in conjunction with fuzzy linguistic variables, for example, A_i can be defined as LARGE, NORMAL, and HIGH, etc., in order to interpret the time series so that computations of the forecasting results are made feasible to understand. However, the high-order of fuzzy time series, i.e., bigger size of each interval in the UoD affects the computation complexity; this issue is address in [10].

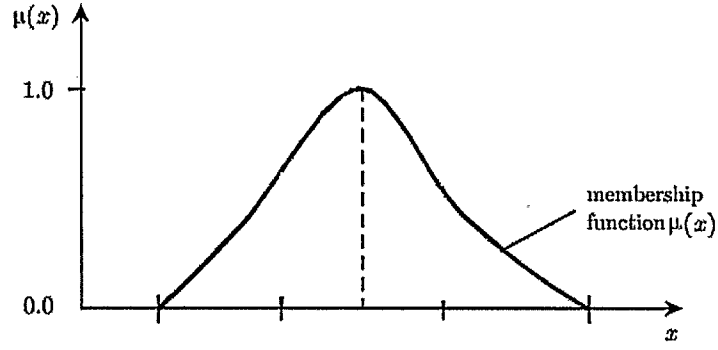


Figure 2.1: A sample membership function

In order to properly define the linguistic variable associated with the fuzzy time series, we need to take the domain knowledge of the time series data into consideration. For example, in this thesis, we are dealing with the financial time series data and if we are to compare the estimated results with stationary values, we can naturally introduce the LOW_STATIONARY, NORMAL_STATIONARY, and HIGH_STATIONARY linguistic values.

Further, *fuzzy modifiers* can be used to enrich the interpretation and flexibility of existing fuzzy linguistic values, and users can define their own fuzzy modifiers to customize the mining results. For example, the fuzzy modifier can be customized as VERY SHORT, NOT and HIGHER_THAN_AVERAGE etc. To apply fuzzy modifiers, the original membership function μ_{A_i} needs to be changed; for example, to apply the negation modifier NOT, simply change the $\mu_{A_i}(x_k)$ to $1 - \mu_{A_i}(x_k)$.

Data Mining Techniques

By definition [53], *data mining* is the process of extracting hidden patterns from data. There are many tasks in data mining including classification, clustering and prediction etc., and there are different algorithms and techniques for each task. In this section, we explore basic data mining tasks and their corresponding techniques used in this thesis.

Association Rule

Association rule mining concerns with finding connections between two set of items that initially seems to be unrelated. Once a rule is formed, the degree of association between the two set of items can be measure with *support* and *confidence* values. We provide formal definitions of the *association rule* below.

Given a set of items $I = \{I_i\}$, $i = 1, \dots, m$ and a list of transactions $D = \{t_j\}$, $j = 1, \dots, n$ called *database*. Each transaction in D contains a subset of items in I , called *itemsets*. An *association rule* is defined in the form of $A \Rightarrow B$, where $A, B \subseteq I$ and $A \cap B = \emptyset$. The *support* (*sup*) of $A \Rightarrow B$ is defined as the percentage of transactions in D that contains $A \cup B$, whereas the *confidence* (*conf*) is simply $\frac{sup(A \cup B)}{sup(A)}$. An itemset that meets the minimum support threshold is called the *frequent itemset*. Finding frequent itemset is an algorithmic task, in this thesis, we use the Rough Set based approach in deriving frequent itemsets and extracting association rules, to be detailed in Chapter 3.

To illustrate association rules, define $I = \{DVD, Camera, Book, Beer\}$ and suppose we have the database of transactions depicted in Table 2.2.

Table 2.2: A sample database of transactions

Transaction ID	Itemsets
1	{ DVD, Camera }
2	{ Camera, Book }
3	{ Beer }
4	{ DVD, Camera, Book }
5	{ Camera }

In this example, both {DVD, Camera} and {Camera, Book} have the support value $2/5 = 0.4$, as {DVD, Camera} appears in transaction 1, 4 and {Camera, Book} appears in transaction 2, 4. If we artificially form an association rule {DVD, Camera} \Rightarrow {Book} the confidence of the rule is then $\frac{\text{sup}(\{DVD, Camera\} \cap \{Book\})}{\text{sup}(\{DVD, Camera\})} = \frac{1/5}{0.4} = 50\%$.

There are many association rule mining algorithms and rule definition variations in current data mining research. The key aspect of association rule mining algorithms is to achieve the efficiency in current database systems, which often contain a large number of transactions. In *Apriori* based algorithms, multiple scans of databases are too costly therefore in this investigation, a set based approach is used.

K-means clustering and classification concepts

Clustering task is an unsupervised learning method with no prior knowledge of data objects, and the goal is to *cluster* similar objects into different groups so that each group of objects share some characteristic to a certain degree. There are three major categories of clustering techniques: partitioning algorithms, hierarchical algorithms and density-based methods. For those different techniques, a fundamental task is to measure the degree of similarities between two data objects, which is often called the *similarity measure* as well as to measure the degree of *dissimilarities* between two data objects.

In this section, we first introduce the famous *K-means* clustering algorithm that is used in the following chapters, and further discuss the issues of evaluating the validity of clustering result with respect to the number of clusters chosen and validity measures.

Algorithm 2.1 K-means clustering algorithm

```

K-means ( $k, D = \{t_j\}, j = 1, \dots, n$ ) { //  $k$  is the desired no. of clusters
                                         //  $D$  is the collection of  $n$  data objects
    Assign initial values for  $k$  means  $c_1, c_2 \dots c_k$ 
    Repeat
        Assign each object in  $D$  to the closest cluster according to the means
        Compute new mean for each cluster
    Until means no longer change
}
```

There are generally two challenges for *K-means* algorithm:

1. How to determine the optimal number of clusters to be used. In our investigation, this number K is crucial because the cluster label is later used as the decision feature in generating association rules.
2. How to distinguish a bad cluster result from a good one. This is a computational merit to be evaluated, and it is closely related to the first challenge with optimal K , as the bad K value in principle deteriorates the evaluation merits. The computational merit is called the *cluster validity* that measures the goodness of a clustering result.

In Chapter 3, we introduce an algorithm in computing an optimal K with 3 different validity measures: *silhouette value*, *Dunn's index* and *Davies-Bouldin's index*. We now describe those validity measures in detail.

- *silhouette value*

The *silhouette* measures average silhouette width for each cluster as well as overall silhouette width for the entire dataset. It further produces a *silhouette plot*, which is a graphical display that allows user to select the optimal number of clusters. The *silhouette value* is computed with the following procedure:

1. Denote $a(i)$ = average dissimilarity of object i to all other objects of the assigned cluster A .

Denote $d(i, C)$ = average dissimilarity of object i to all other objects of cluster $C \neq A$.

2. Compute $d(i, C)$ for all clusters $C \neq A$, and then select the smallest among those, denote $b = \min\{d(i, C)\}$ for all $C \neq A$
3. Let cluster M be the output from the previous step that attains the minimal average dissimilarity, i.e., $d(i, M) = b(i)$, which is called the *neighbour* of object i .
4. The *silhouette value* $s(i)$ is defined as $s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}$

From the procedure, we see that $-1 \leq s(i) \leq 1$. If $s(i)$ is close to 1, it implies that the sample is well clustered; if $s(i)$ is close to -1, it implies that the sample is poorly clustered.

- *Dunn's index*

Dunn's index is used to identify compact and relatively well separated clusters. It can be calculated with the following formula:

$$D = \min_{1 \leq i \leq n} \left\{ \min_{1 \leq j \leq n, i \neq j} \left\{ \frac{d(c_i, c_j)}{\max_{1 \leq k \leq n} \{d'(c_k)\}} \right\} \right\}$$

Where c_i represents the i^{th} cluster among all the n clusters; $d(c_i, c_j)$ measures the distance between clusters c_i and c_j ; $d'(c_k)$ measures intra-cluster distance of c_k .

From the formula, we can see the *Dunn's index* is to maximize the distance among all clusters and compress the intra-cluster distance. Therefore, the greater the index is, the better is the clustering result.

- *Davies-Bouldin's index*

This index is a function of the ratio of the sum of within-cluster scatter to between-cluster separation, it uses both the clusters and their sample means [46]. It can be calculated as follows:

$$DB = \frac{1}{n} \sum_{i=1}^n \max_{i \neq j} \left\{ \frac{S_n(c_i) + S_n(c_j)}{S(c_i, c_j)} \right\}$$

Where n is the total number of clusters; $S_n(c_i)$ is the average distance of all objects from the i^{th} cluster to their cluster centre; $S(c_i, c_j)$ is the distance between clusters' centers. From the definition, we see that smaller *Davies-Bouldin's index* implies the clustering result that is compact and separations of clusters are relatively far enough.

Besides clustering, classification is also an important branch of data mining research that features a set of classes $C = \{C_1, \dots, C_m\}$ and a mapping function that maps each data object to a class in C . After classification process, each class in C contains data objects mapped to it. With this learning methodology, the crucial issue is then how to derive the set of classes C . In this thesis, the classification label is derived in a domain specific manner, i.e., using the domain knowledge to structure the possible classes in

order to best interpret the knowledge. In Chapter 3, we introduce a classification algorithm to be used in the hybrid mining procedure.

In this chapter, we have surveyed important concepts and techniques pertaining to the hybrid mining procedure covered in Chapter 3.

CHAPTER 3

MINING PROCEDURES

Prediction in data mining is a task to estimate future unseen values with certain sets of historical values. Mathematically, both linear models and non-linear models can be used in modeling prediction processes. In this chapter, we look at financial time series forecasting from the hybrid perspective and present two forecasting procedures: one with linear mathematical prediction model and the other with static feature appending. Both procedures use the fuzzy discretization technique to pre-process the datasets and better interpret the mining results. The procedures are potentially expandable with other appropriate statistical models in different settings.

Statistical Modeling of Fuzzy Parameters

In this section, we discuss three statistical modeling factors that constitute the fuzzy discretization process:

- Accuracy of estimation - Since we use $AR(1)$ to find the best-fit of the training dataset and $GARCH(1, 1)$ to estimate the volatility of the financial time series, we need to naturally consider the accuracy of estimations from those models used.
- Model accuracy - We need to gauge certain properties in those statistical models.

In principle, in $AR(1)$ model, the stationary value $\frac{\mu}{1-\alpha}$ measures the degree of reversion from the mean that is controlled by the model parameter α ; in

$GARCH(1, 1)$ model, the stationary variance $\frac{a_0}{1-a-b}$ measures how fast the estimated variance converges to the stationary value. These factors account for the models' characteristic per se.

- Trends in financial time series data - Trends or patterns in financial time series data such as seasonality have a great impact in decision-making process. We consider the following upcoming trend. The trend factor is derived from the statistical models in use. The advantage is that, both of the statistical models we use have multiple-step forecasting estimations, allowing us to formalize fuzzy trend parameters efficiently.

For the “accuracy of estimation” factor, sample linguistic variables include VERY_LOW_ACCURACY, LOW_ACCURACY, NORMAL_ACCURACY and HIGH_ACCURACY. Similarly, sample linguistic variables include VERY_LOW_STAT, LOW_STAT, NORMAL_STAT and HIGH_STAT in order to model the “model accuracy” factor. To model the “trends” factor, we measure the change in percentage between the current return and the previous or the following return. The time window to consider can be derived from the ACF computation in (4). Sample linguistic variables for modeling trends factor include NORMAL, WEAK and STRONG. We use the $GARCH(1, 1)$ estimation in modeling trends because the $GARCH$ model produces more accurate results on volatility clustering with its statistical nature.

In our mining procedures, we use fuzzy discretization technique to pre-process the datasets and better illustrate the mining results. The fuzzy set membership functions to be used in this investigation are:

- RIGHT LINEAR:

$$\text{right-linear}(x:a,b) = \begin{cases} 1 & \text{for } x < a \\ (b-x)/(b-a) & \text{for } a \leq x \leq b \\ 0 & \text{for } b < x \end{cases}$$

- LEFT LINEAR

$$\text{left-linear}(x:a,b) = \begin{cases} 0 & \text{for } x < a \\ (x-a)/(b-a) & \text{for } a \leq x \leq b \\ 1 & \text{for } b < x \end{cases}$$

- TRIANGLE

$$\text{triangle}(x:a,b,c) = \begin{cases} 0 & \text{for } x < a \\ (x-a)/(b-a) & \text{for } a \leq x \leq b \\ (c-x)/(c-b) & \text{for } b < x \leq c \\ 0 & \text{for } x > c \end{cases}$$

We combine these membership functions in translating the financial time series dataset to be detailed in the forecasting procedure. Inputs to the membership functions for the three modeling factors are change in percentage of returns calculated by:

$$\Delta = \frac{r_{t+k} - r_t}{r_t} \times 100$$

Where r_{t+k} is the k -step forward return of the asset and r_t is the return of the asset in the time period t . Unlike in [26] where k is defined by the user, we can compute ACF values for different lags in the time series data to derive a possible k that renders the maximal pattern correlation.

Financial Ratios

In this section we introduce two categories of the most commonly used financial ratios in assessing firms' economical condition, namely *liquidity, capital structure and solvency ratios* and *return on investment, profitability and earning ratios*. All the ratios originate from the firms' publicly available financial statements such as balance sheet, income statement and the cash flow statement.

Liquidity refers to the ease with which assets can be converted to cash. This set of ratios hence measures the relationship of a firm's liquid assets to current liabilities and provides information about the short-term viability of the business. Individual investors sometimes are interested in these ratios to gauge the firm's ability to pay its current obligations and to continue operations. To measure the long-run solvency of the firm, we need capital structure and solvency ratios. *Solvency* is a firm's financial ability to survive in the long run by paying its long-term obligations, whereas liquidity is expressed in terms of short-term obligations. The list of ratios used in this thesis is shown in Table 3.1.

Return on investment, profitability and earning ratios are more directed to individual investors as they measure the return of the financial entities. For example, the *return on investment (ROI)* ratio tells investors whether management is using invested funds wisely. It also provides a profitability measure relating both to the income statement and the balance sheet that can be adjusted to reflect the contributions of creditors or equity providers such as bank. The list of ratios used in this category is shown in Table 3.2.

Table 3.1: Liquidity, capital structure and solvency ratios

Ratio	Definition
Current Ratio	Current assets/Current liabilities
Acid test ratio	(Cash + Cash equivalents + Net receivables + Market securities)/Current liabilities
Receivables turnover ratio	Net credit sales/Average accounts receivable
Inventory turnover ratio	Cost of sales/Average inventory
Operating cycle	365/ Receivables turnover ratio + 365/Inventory turnover ratio

Table 3.2: ROI, profitability and earning ratios

Ratio	Definition
Profit margin on sales	Net income after interest and taxes/Net sales
Total asset turnover ratio	Net sales/Average total assets
Return on investment	Net income after interest and taxes/ Average total assets
Return on common equity	(Net income after interest and taxes – preferred dividends)/Average common equity
Dividend payout ratio	Dividends per common share/Earnings per share

These two categories of financial ratios can be seen as the *taxonomies* in the classification problem and input to the *Rough Set* model. We will treat the ratios listed in Table 3.1 and Table 3.2 as features in one of the procedures in the next section. Our mining goal is essentially to reduce all those features to a minimal and choose the one(s) with the best information selection criteria in the form of association rules in decision making processes.

Forecasting Procedure

Using the statistical models and the fuzzy discretization technique, we can do the forecasting on the financial time series datasets by the following steps:

Step 1. Prepare the data. In this step, we calculate the returns of the assets in adjacent time periods and re-label the time series dataset based on their collecting time. For example, if the data is collected monthly and two years' data is collected, we re-label the data rows sequentially instead of the original "2001Jan, 2001Feb, 2001Dec", as this numerical translation facilitates numerical processing. The returns of the assets are calculated using (3). In this thesis, we only calculate the asset returns in adjacent periods because we intend to capture the nearest stock movement possible. For example, if we take the dataset of NASDAQ Composite Index from Feb. 1971 to Apr. 2009, the prices and returns are both shown in Figure 3.1, where we can see that the returns are well scaled.

Step 2. Fit the dataset into $AR(1)$ and $GARCH(1, 1)$ model. We do the following computations on the training set:

- Fit the return time series into $AR(1)$ model to estimate α in (5).
- Do one-step forecasting based on $E(R_{t+1} | R_t) = \mu + \alpha R_t$, where μ and α are the same as in (5).
- Compute the forecasting error of using $AR(1)$ model by $D'_{t+1} = E(R_{t+1} | R_t) - D_{t+1}$, where D_{t+1} is the actual value in the original time series data and D'_{t+1} is the forecasting error.
- Fit the return time series into $GARCH(1, 1)$ model to estimate α , a_0 , a and b in (8).

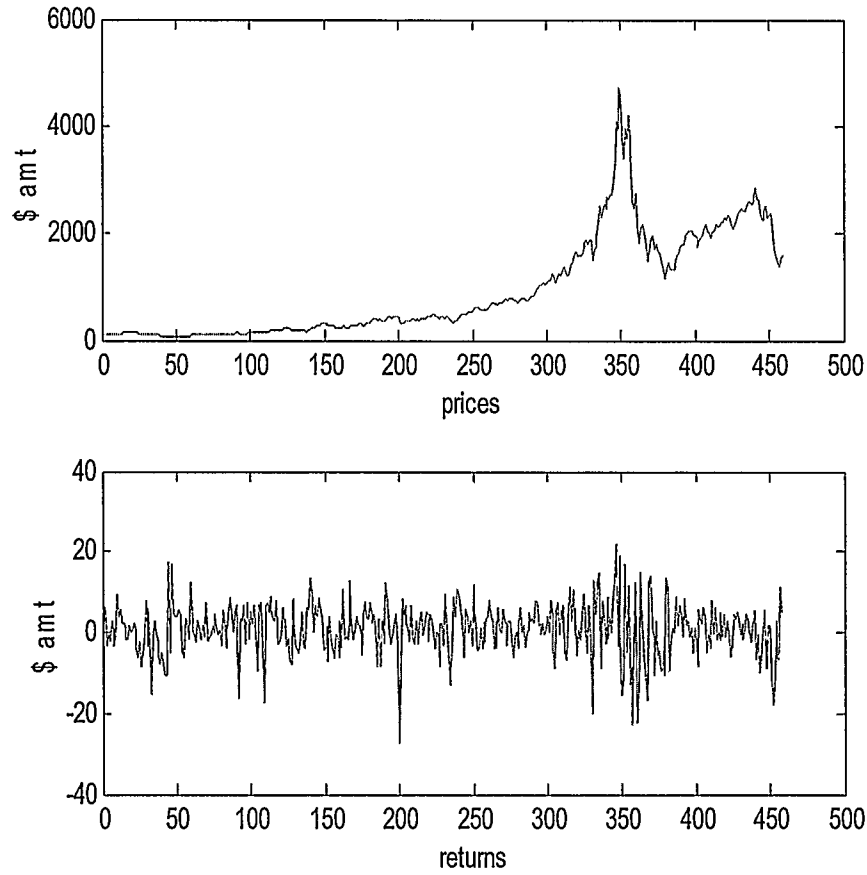


Figure 3.1: NASDAQ Composite Index from Feb. 1971 to Apr. 2009 normal prices and returns.

Step 3. Fuzzify the values computed from step 2. We fuzzify the measurements from the three factors defined previously, that is, we compute the maximum membership for input values. We define the *universe of discourse*, UoD , as $UoD = [\lfloor I_{\min} \rfloor, \lceil I_{\max} \rceil]$, where I_{\max} is the maximum value of the dataset to be fuzzified and I_{\min} is the minimum value of the dataset to be fuzzified. For example, if the data series to be fuzzified is $[-2.1, 1, 4, 10.1]$, the UoD is then $[\lfloor -2.1 \rfloor, \lceil 10.1 \rceil] = [-3, 11]$. We then partition the UoD to form

intervals u_1, \dots, u_m . For example, we can partition the above *UoD* $[-3, 11]$ into the intervals $u_1 = [-3, -1]$, $u_2 = [-1, 1]$, \dots , $u_7 = [9, 11]$. After the *UoD* is partitioned, we can define the fuzzy sets on the *UoD*. This step translates the numerical returns to linguistic variables so that the degree of return variation can be interpreted with fuzzy terms. For example, if we define the following five fuzzy set variables:

$$A_1 = (\text{VERY_SMALL_VARIATION}),$$

$$A_2 = (\text{SMALL_VARIATION}),$$

$$A_3 = (\text{NORMALL_VARIATION}),$$

$$A_4 = (\text{LARGE_VARIATION}), \text{ and}$$

$$A_5 = (\text{VERY_LARGE_VARIATION})$$

The fuzzy sets A_1 to A_5 can be defined as follows on the *UoD*:

$$A_1 = 1/u_1 + 0.3/u_2 + 0/u_3 + 0/u_4 + 0/u_5 + 0/u_6 + 0/u_7$$

$$A_2 = 0/u_1 + 0.7/u_2 + 0.7/u_3 + 0/u_4 + 0/u_5 + 0/u_6 + 0/u_7$$

$$A_3 = 0/u_1 + 0/u_2 + 0.3/u_3 + 1/u_4 + 0.3/u_5 + 0/u_6 + 0/u_7$$

$$A_4 = 0/u_1 + 0/u_2 + 0/u_3 + 0/u_4 + 0.7/u_5 + 0.7/u_6 + 0/u_7$$

$$A_5 = 0/u_1 + 0/u_2 + 0/u_3 + 0/u_4 + 0/u_5 + 0.3/u_6 + 1/u_7$$

This fuzzy translation is highly interpretative in nature. Up to this step in the forecasting procedure, we have combined the statistical models and the accuracy interpretations.

Step 4. In this step, we perform unsupervised learning, i.e., *K-means* clustering. We rely on the statistical models introduced and do not derive classification feature from

the training data beforehand, as it is safer not to pre-judge the unseen data. We use the *K-means* clustering algorithm in our investigation. However, the important parameter of the *K-means* to be determined prior to the clustering process is the number of clusters K . We present the Algorithm 3.1 for determining the number of clusters and further use the volatility clustering estimation from the *GARCH*(1, 1) model to gauge the separateness of the clusters in the Decision Table.

After we find the best K by the Algorithm 3.1, we cluster the training data with the *K-means* algorithm and use clusters as the class labels or decision features for further pattern extraction.

Step 5. We input the Decision Table to the Rough Set model, with definitions of condition (C) and decision attributes (D) as:

C = statistical measures from step 2

$D = 1, 2, \dots, K$ for K clusters from step 4

We rely on the Rough Set theory in finding association rules for prediction. We devise a sub-procedure to discover the rule-based knowledge in financial time series datasets.

5.1. Classifications. In the *Information System* $\langle U, C \cup D \rangle$, let $U = \{u_1, u_2, u_3, \dots, u_{|U|}\}$. We perform the classification U / a for an attribute $a \in C \cup D$ by invoking Algorithm 3.2.

Algorithm 3.1 Determining number of clusters

```

determine-K (data) {
  kmax ← Arg. maximum of ACF(data)
  kmin ← Arg. minimum of ACF(data)
  Initialize list  $V = \{\text{silhouette value, Dunn's index, Davies-Bouldin's index}\}$ 
  Initialize indexMatrix =  $[a_{ij}] \ 1 \leq i \leq kmax; 1 \leq j \leq 3 = 0$ 
  For all k from kmin to kmax
    Run k-means with k clusters
    indexMatrix[k, :] = 3 cluster validity indices in list V
  For t = 1 ... 3
    Sort indexMatrix by column t in the order of increasing cluster validity
    Write indexMatrix[:, t] = 1,...,i

  Output Arg.max( $\sum_{1 \leq p \leq i} \text{indexMatrix}[p, :]$ )
}

```

Algorithm 3.2 Simple classification of U

```

Classification {
  s ← 1 //equivalent class counter
   $V_1, V_2, \dots, V_s = \emptyset$ 
  for each  $u_i, i = 1, 2, \dots, |U|$ 
    if  $a(u_i) = a(V_j)$  for some  $j \in [1, s]$ 
      add  $u_i$  to  $V_j$ 
    else
      s ← s + 1
      make a new class  $V_s$  and add  $u_i$  to  $V_s$ 
}

```

5.2. Find Rough Subsets and Support Subsets. Let W be a subset of U . A *rough subset* of W from classification results of Algorithm 3.2 is a pair of subsets approximated by attribute selection P over C ($\underline{P}W, \overline{P}W$). The lower approximation $\underline{P}W$ is said to be the *support subsets* to W from attribute P . We find the *support subsets* to all classified equivalent classes. For example, in the scope of this investigation, if we have 7 equivalent classes for the decision variable D . We can name the equivalent classes

W_1, W_2, \dots, W_7 and then we can find the *support subsets* of W_1, W_2, \dots, W_7 approximated by attributes x_1, x_2, \dots, x_K .

5.3. Rule generation. In this step, we generate the rules to be used in financial forecasting. For some condition attribute x and decision attribute w , if we have the *support subsets* computed by \underline{PW} , this implies that rule $x \rightarrow w = w(W)$ can be generated.

$\underline{\gamma}_P = \frac{|\underline{PW}|}{|U|}$ is said to be the *belief* to W from attribute(s) P .

Step 6. In this step, we intend to refine the patterns found in step 5. We firstly extract the patterns for prediction, then “merge” some patterns generated from the Rough Set computation. For example, if we have three patterns generated: $\{A, B, C\} \rightarrow \{1\}$, $\{A, B, D\} \rightarrow \{1\}$ and $\{A, B, E\} \rightarrow \{1\}$. In this case, we may view the third attribute as “don’t-care” and the rule can be refined to $\{A, B, *\} \rightarrow \{1\}$. A, B, C, D, E can be any of the fuzzy linguistic variables defined in step 3. The degree of merging is defined by the end user. Upon Step 6, we have generated a set of possibly useful rules for prediction.

Step 7. In this step, we prepare for the forecasting based on the set of rules generated from step 6. We intend to find the confidence values of the generated rules on the training dataset. We use these confidence values as virtual weights to predict unseen values.

Step 8. In this step, we do the final forecasting on the decision variable by the following procedure.

1. Repeat step 1 to 3 to fuzzify the testing dataset.
2. Match the patterns found in step 6 with the testing dataset.

3. If there is only one match, the predicted value is the decision feature indicated in the rule.
4. If there are multiple matches, the predicted value is the decision feature indicated in the rule with confidence value defined by end users.
5. If there is no match, we record a miss and the row is added in the training dataset in future mining tasks.

An Alternative Approach

In the above procedure, we use statistical measurements as features in order to form the Information System as input to the Rough Set model. The hybrid approach essentially incorporates financial literature and the data mining paradigms. As an alternative approach, we can manually append features from the domain specific knowledge, for instance, we can append financial ratios to the dataset and use them as features to form the Information System instead. In this thesis, the financial ratios chosen are listed in Table 3.1 and Table 3.2. Ideally, the financial ratios are from the firms' publicly available financial statement; however, realistically, the dataset is complicated to obtain, as this involves continuous computation effort for each period of concern with different enterprise management software.

The alternative decision feature

As input to the Rough Set model, we fuzzify the calculated financial ratios (from this point onwards, we only deal with the fuzzified dataset) to form the Information System or Decision Table.

The decision feature or classification label needs to be defined formally in this alternative procedure. We define three major decision values representing buy (+1), hold (0) and sell (-1), then fill the intervals with “indifferences”. The decision feature is computed in two steps. First, we calculate the percentage of change in stock returns as follows:

$$\text{change in percentage (\%)} = \frac{r_i - r_{i-1}}{r_{i-1}} \text{ for } i > 0$$

Where r_i is the current stock return and r_{i-1} is the $(i-1)^{\text{th}}$ stock return. Write $c = r(i) - r(i-1)$ for $i > 0$, then we append the decision features by rules depicted in Figure 3.2.

Alternative Steps 4 and 5

The alternative approach modifies steps 4 and 5 of the procedure previously described with calculated decision features. The alternative steps are detailed below:

Step 4’. In this step, we compute the decision features by Figure 3.2. For each row of record, we append the decision attribute to the fuzzified dataset. Now we have all the elements to define the Decision Table in the context of Rough Set computation.

Step 5’. We input the Decision Table to the Rough Set model, with definitions of condition variable (C) and decision attributes (D) as:

$$C = x_i \text{ for } i = 1, 2, \dots, 10 \text{ (ratios)}$$

$$D = 0, +1, -1, \text{ (hold, buy and sell decisions)} \\ \text{"not buy indiff."}, \text{"not buy - hold indiff."}, \\ \text{"buy - hold indiff."}, \text{"buy indiff."}$$

$$\text{decision feature} = \begin{cases} \text{"not buy indiff."} & \text{if } \text{sign}(c) < 0 \text{ and } \% > 50\% \\ -1 & \text{if } \text{sign}(c) < 0 \text{ and } \% \in (25\%, 50\%] \\ \text{"not buy - hold indiff."} & \text{if } \text{sign}(c) < 0 \text{ and } \% \in (0, 25\%] \\ 0 & \text{if } c = 0 \\ \text{"buy - hold indiff."} & \text{if } \text{sign}(c) > 0 \text{ and } \% \in (0, 25\%] \\ +1 & \text{if } \text{sign}(c) > 0 \text{ and } \% \in (25\%, 50\%] \\ \text{"buy indiff."} & \text{if } \text{sign}(c) > 0 \text{ and } \% > 50\% \end{cases}$$

Figure 3.2: Decision features for the alternative procedure

Note that the Decision Table is defined differently from the first procedure, because we are using a different set of features, namely the financial ratios defined in Table 3.1 and Table 3.2 instead of the statistical measurements.

The reason we introduce this alternative approach is that, for modern ERP systems, the difficulty of obtaining the data is alleviated because the data sources are integrated, for example, the balance sheet statement and the income statement can be obtained from the system with less effort from a centralized data repository. Under such circumstances, this alternative approach is useful as it centralizes all the representatives of useful financial information from the firm and process the relatively sound information to get the knowledge that will otherwise be hidden from external users. However, in most cases the information is decentralized, causing the inefficiency of analysis.

In this chapter, we have introduced two hybrid mining procedures that lead to financial prediction; in Chapter 4, we will conduct case studies to exemplify the procedures.

CHAPTER 4

EXPERIMENTAL ANALYSIS

In this chapter, we conduct experimental case studies for the two hybrid procedures presented in Chapter 3. We use real-time historical datasets in evaluation the procedure as well as simulated datasets in gauging the effectiveness of the approach. We also discuss the limitations of the approach by the experimental statistics.

Datasets

In order to conduct case studies for the two procedures, the datasets taken are from two resources, all publicly available:

1. For the industry-wise data such as the NASDAQ and S&P index, the datasets are resourced from <http://www.economy.com/freelunch/>.
2. For the company level data such as the Microsoft's daily or monthly stock returns, the datasets are taken exclusively from the book "*Analysis of Financial Time Series*", Ruey S. Tsay, Wiley 2002. The data is available from the website: <http://faculty.chicagobooth.edu/ruey.tsay/teaching/fts/>

For the alternative procedure where features are manually appended, we need to generate synthetic datasets, as the data is nowhere available publicly and the point of the simulated experimental study is to demonstrate the usability of the model so that software engineers are to embed the procedure into real-time systems. For this purpose of concern, we need to simulate different datasets.

Case Study I: NASDAQ Composite Index

In this section, we study the case of NASDAQ Composite Index from Dec. 1971 to Dec. 2008 to verify the first hybrid mining procedure on the financial time series data. The raw data is accumulated monthly as shown in Table 4.1.

Table 4.1: NASDAQ Composite Index from Dec. 1971 with normal prices

1971M12	114.12
1972M1	118.87
1972M2	125.38
1972M3	128.14
1972M4	131.33
1972M5	132.53
1972M6	130.08
1972M7	127.75
1972M8	129.95
1972M9	129.61
1972M10	130.24001
...	...

Table 4.2: Summary of features added for NASDAQ dataset

n th Column	Features/Condition variables
2	asset returns
3	$AR(I)$ estimates
4	$AR(I)$ estimation errors
5	$GRACH(I, I)$ estimates
6	$\Delta_{AR} = AR(I)$ stationary value – asset returns
7	$\Delta_{GARCH} = GRACH(I, I)$ stationary value – asset returns

In order to fit the data into AR and $GARCH$ models, we first prepare the data and compute the returns based on the normal prices as described in step 1 in the forecasting procedure; after the preparation, in step 2 we fit the data into $AR(I)$ and $GRACH(I, I)$ model and use them as features for clustering. In total, we add five columns of features

that are used as condition variables in the Rough Set computation, summarized in Table 4.2.

After the statistical computations, we are to fuzzify the features in step 3. We introduce ten fuzzy linguistic variables in this case of investigation. The ten fuzzy linguistic variables represent ample information for knowledge interpretation within the scope of our investigation, i.e., the degree of returns' increase or decrease; because we are using financial asset returns instead of normal prices, the datasets of interest are generally evenly distributed among ten intervals with the balance of interpretation effort. We later discuss the impact of the number of intervals and scaling of the datasets.

For example, for the asset return column, we define $UoD = [-28 \ 22]$, and define the partition of the UoD as:

$$u_1 = [-28 \ -22.5], u_2 = [-22.5 \ -17], \dots, u_{10} = [16 \ 22]$$

The ten fuzzy linguistic variables defined are:

$$A_1 = (\text{ABNORMAL_SMALL_DECREASE}),$$

$$A_2 = (\text{EXTREME_SMALL_DECREASE}),$$

$$A_3 = (\text{VERY_SMALL_DECREASE}),$$

$$A_4 = (\text{NORMAL_SMALL_DECREASE}),$$

$$A_5 = (\text{TRIVIAL_SMALL_DECREASE}),$$

$$A_6 = (\text{NORMAL_FLUCTUATION}),$$

$$A_7 = (\text{TRIVIAL_LARGE_INCREASE}),$$

$$A_8 = (\text{NORMAL_LARGE_INCREASE}),$$

$$A_9 = (\text{VERY_LARGE_INCREASE}),$$

$$A_{10} = (\text{EXTREME_LARGE_INCREASE})$$

We use the combination of the membership functions, namely LEFT LINEAR, RIGHT LINEAR, and TRIANGLE. The LEFT LINEAR and RIGHT LINEAR membership functions are used once to fuzzify “both ends”, whereas the TRIANGLE membership function is used to fuzzy the rest of intervals. The membership functions are shown in Figure 4.1.

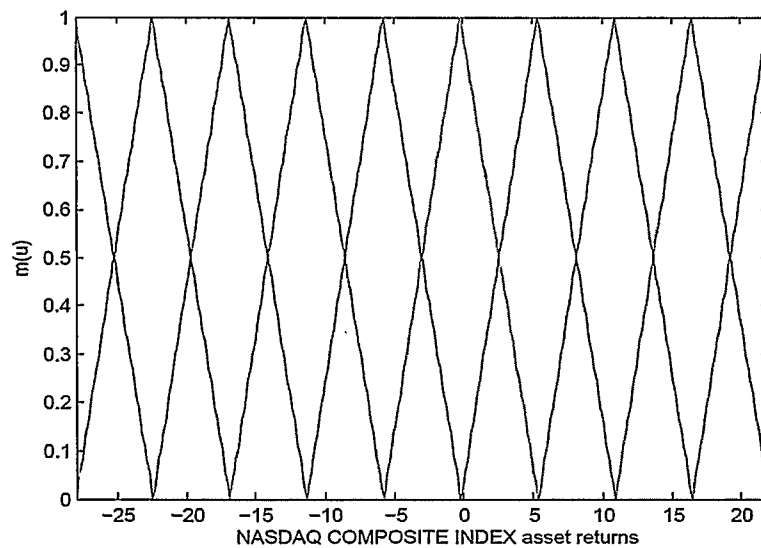


Figure 4.1: Fuzzy membership functions for the asset return feature

We next fuzzify each column with the ten linguistic variables defined. For stationarity comparisons (features in column six and seven), the linguistic variables are being named to reflect the meaning of the features, as follows:

$$A_1 = (\text{ABNORMAL_SMALL_CHANGE}),$$

$$A_2 = (\text{EXTEME_SMALL_CHANGE}),$$

$$A_3 = (\text{VERY_SMALL_CHANGE}),$$

$$A_4 = (\text{NORMAL_SMALL_CHANGE}),$$

$$A_5 = (\text{TRIVIAL_SMALL_CHANGE}),$$

$A_6 = (\text{NORMAL_FLUCATION}),$
 $A_7 = (\text{TRIVIAL_LARGE_CHANGE}),$
 $A_8 = (\text{NORMAL_LARGE_CHANGE}),$
 $A_9 = (\text{VERY_LARGE_CHANGE}),$
 $A_{10} = (\text{EXTREME_LARGE_CHANGE})$

In step 4, we cluster the fuzzified dataset with the algorithm *determine-K(data)*. In order to do the clustering, we first need to specify the minimum and maximum number of clusters with the aid of ACF values to find possible correlation lags in the dataset. The range of possible lags implies the possible number of clusters by the definition of ρ_k in (4). The ACF computation is shown in Figure 4.2.

In Figure 4.2, we see the minimum number of clusters is 2 and the maximum number of cluster is 26; therefore, we choose the optimal number of clusters between 2 and 26 in the algorithm *determine-K(data)*. We cluster the dataset with the output of the algorithm *determine-K(data)*, which is 12 for the NASDAQ Composite Index dataset, shown in Figure 4.3. The fuzzified features and cluster labels are shown in Table 4.3.

In Table 4.3, we use numerical values instead of long linguistic variables for the compact representation, and it is made easier to deal with for the implementations with MATLAB. To interpret values in Table 4.3, substitution can be made as

$A_i \leftarrow i, i = 1, 2, \dots, 10$ for features 1 to 6.

We input the fuzzified dataset and the cluster labels to the Rough Set model, where condition variables (C) decision variables (D) are defined as follows:

C = asset returns, $AR(I)$ estimates, $AR(I)$ errors, $GRACH(I, I)$ estimates,

Δ_{AR} and Δ_{GARCH}

$D = 1, 2, \dots, K$ where K is the clustering label

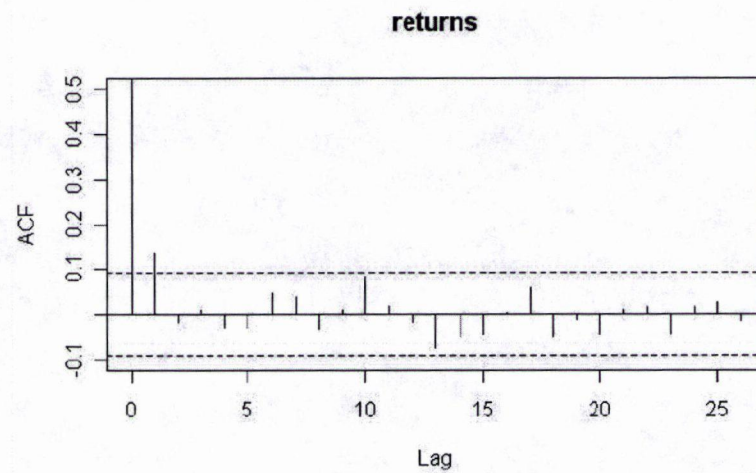


Figure 4.2: ACF computations showing a range of lagging factors

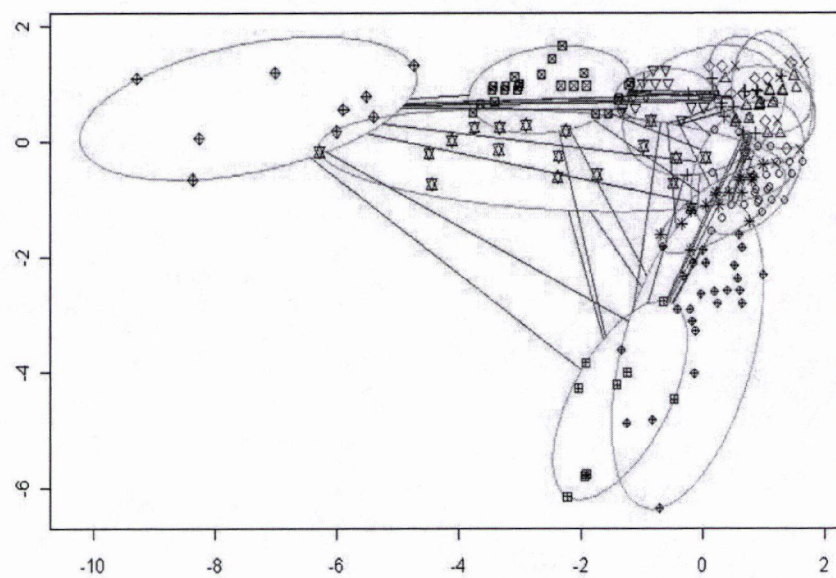


Figure 4.3: Cluster plot with output of *determine-K(data)* algorithm

We use the Rough Set computation methods described in step 5 to find association rules on the fuzzified dataset. With the NASDAQ Composite Index dataset in this case study, we have found 91 rules for the time period considered. Some sample rules are shown in Figure 4.4.

Table 4.3: Fuzzified features and clustering results of NASDAQ Composite Index

Original	AR(1)	AR(1) error	GRACH(1, 1)	Δ_{AR}	Δ_{GARCH}	Cluster
7	8	2	3	2	4	4
7	7	3	3	3	4	4
6	7	2	2	2	5	6
6	7	2	2	2	5	6
6	7	1	2	1	5	12
6	7	2	2	2	5	6
6	6	2	2	2	5	8
6	6	2	2	2	5	8
6	7	2	1	2	5	9
6	6	1	1	1	5	10
6	6	2	1	2	5	8
6	7	1	1	1	5	12
5	7	3	1	3	6	3
5	6	3	1	4	6	3
6	5	2	1	2	6	8
5	6	4	1	4	7	3
5	5	3	2	3	6	3
6	6	2	2	2	5	8
7	6	3	2	3	4	11
5	8	3	2	3	6	3
7	6	3	2	3	4	11
6	7	2	2	2	5	6
3	6	6	2	6	8	12
...

```

rule 1. (A2 = 9) & (A6 = 6) => (D1 = 1);
rule 2. (A4 = 10) & (A5 = 2) => (D1 = 1);
rule 3. (A4 = 8) & (A5 = 2) => (D1 = 1);
rule 4. (A3 = 2) & (A4 = 6) => (D1 = 1);
rule 5. (A2 = 10) => (D1 = 1);
rule 6. (A2 = 9) & (A4 = 6) => (D1 = 1);
rule 7. (A2 = 9) & (A4 = 7) => (D1 = 1);
rule 8. (A4 = 7) & (A5 = 4) & (A6 = 6) => (D1 = 1);
rule 9. (A1 = 8) & (A3 = 6) => (D1 = 2);
rule 10. (A1 = 9) => (D1 = 2);
...      ...      ...      ...

```

Figure 4.4: Sample association rules found with NASDAQ data

If we choose to perform the rule fine-tuning in step 6, the number of rules will be smaller, for example, rule 1, rule 6 and rule 7 can be merged to a new rule as $(A2=9) \Rightarrow (D=1)$. This fine-tuning process is user-dependent in that some features are deemed less important than others in domain specific considerations, as in the above example, one can prioritize the feature A2 compared with features A4 and A6. Therefore, the merging process can be done without loss of relatively important information. In the final step, we can use the rules found to predict certain decision represented by the clustering labels.

Since the financial dataset is usually sensitive and volatile, the confidence of the rules found is lower on average compared with certain other datasets from different domains such as the datasets used for cancer prediction. In the NASDAQ Composite Index dataset, we choose relatively small confidence values (in this and following case studies the confidence value chosen is 0.2) in doing the forecasting. In other words, we are more concerned with the coverage the rules found for the forecasting, i.e., we want to be certain that the rules found match the testing dataset as much as possible in terms of quantity so that the knowledge is relatively sound. In the NASDAQ Composite Index dataset, we set the training dataset to be different ranges, and choose to forecast decision

labels also in different periods. We merge the rules as much as possible to forecast the unseen values; the result is shown in Table 4.4.

Table 4.4: Experimental results with NASDAQ Index data using training dataset from 1972 to 1992 to forecast various periods

Training Data	Testing Data	Rule Coverage	True positives with rules found
1972M1-1992M12	1993M1-1993M12	83%	92%
1972M1-1992M12	1993M1-1995M12	80%	86%
1972M1-1992M12	1993M1-1997M12	79%	86%
1972M1-1992M12	1993M1-1999M12	74%	81%
1972M1-1992M12	1993M1-2001M12	71%	70%
1972M1-1992M12	1993M1-2005M12	65%	63%
1972M1-1992M12	1993M1-2008M12	60%	61%

Table 4.5: Experimental results with NASDAQ Index data using training dataset from 1972 to 1992 and testing dataset from 2000 to 2008 individually

Testing Data	Rule Coverage	Prediction misses with rules found out of 12
2000	84%	5
2001	84%	3
2002	81%	5
2003	82%	7
2004	78%	5
2005	80%	7
2006	80%	6
2007	76%	6
2008	75%	7

Table 4.6: Experimental results with NASDAQ Index data using different amount of training dataset and testing dataset from 1993 to 2008 individually

Test Data	Prediction misses with Training Data A – 10 years prior to Testing Data out of 12	Prediction misses with Training Data B – 20 years prior to Testing Data out of 12
1993	1	0
1994	3	1
1995	2	2
1996	3	2
1997	2	1
1998	3	0
1999	3	2
2000	1	1
2001	2	1
2002	2	2
2003	2	1
2004	1	1
2005	4	1
2006	3	1
2007	1	0
2008	4	4

From the statistics above, we can conclude the following in this case study:

1. As the testing datasets accumulate remotely from the period of the training dataset, the percentage of rule coverage decreases as well as the true positives found under the coverage.
2. The discrepancy illustrates that the effects of the lagging factor, that is, as time elapses, the correlation in the financial time series is weaker.
3. Using the fix training dataset of the same interval to predict relatively remote single years does not give accurate results, for example in Table 4.5, the average prediction miss is 5.7 out 12, or roughly 47%.
4. Using more less-related training dataset slightly increase the prediction accuracy. For example in Table 4.6, by using 10 more years of data to train the model, the

accuracy is getting better in general, but because the added 10 years is further related to testing data period, the improvement is not significant as the previous point implies: the average discrepancy is 0.94

The overall procedure gives satisfactory results, as practically speaking the user requires most recent data to forecast adjacent periods, that is, for example, if user choose to train the model with 20 years' data on hand from 1966 to 1985, he or she would usually use the trained model to predict the unseen values from 1986 onwards. The periodic distance between the training and testing datasets is usually adjacent. Generally speaking, the nature of the financial time series dataset is crucial to the forecasting approach, as the sensitivity and the way of data accumulation influence the fuzzy discretization, for an extreme example, if the data is accumulated yearly, the movement is unlikely to be captured accurately, as some intra-year stock fluctuations along with changing economical conditions will not be reflected. As a result, the lagging factor for the closest fluctuations will be ineffective and that consequently leads to the situation where the clustering labels are incapable of representing the knowledge.

In the following sections, we are to verify how our procedure performs with regard to datasets accumulated differently other than the NASDAQ Composite Index, and to illustrate the limitations of the approach with certain simulated datasets.

Case Study II: Japanese Nikkei Average Index

In this section, we are to briefly apply the forecasting procedure to the shared price index: Japanese Nikkei Average Index dataset that is accumulated daily. The purpose is to investigate how the procedure reacts to more frequently accumulated datasets in general.

The UoD is $[-24, 21]$ for the original returns; we define 10 fuzzy intervals for the returns similar to the NASDAQ Composite Index dataset. The ACF graph suggests that the number of clusters is between 2 and 24. We determine the number of clusters with the algorithm $determine-K(data)$. The optimal number of clusters is 7 in this case. The cluster plot is shown in Figure 4.5.

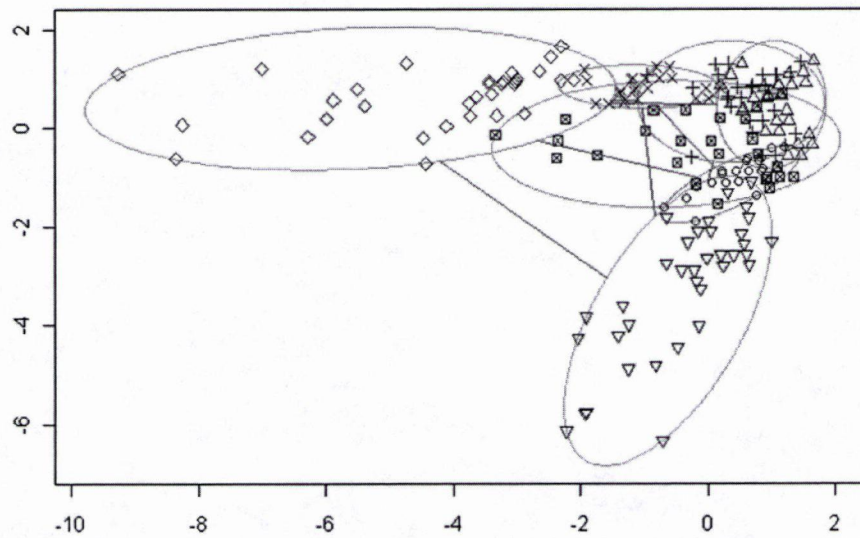


Figure 4.5: Cluster plot for Nikkei Average Index dataset

We can see that the number of clusters is fewer than the case of the NASDAQ Composite Index dataset with daily accumulated data. This implies that the data series is less volatile and the statistical models render more accurate measurements. Since we are using the clustering labels for the decision feature, we can be more certain that the rules to be found are more determinative when the number of possible categories of decisions is smaller, i.e., having less decision labels in the unsupervised learning phase in the overall procedure.

After getting the decision rules from the Rough Set model, we perform a comparative experiment similar to the methodology used in the case of NASDAQ Composite Index dataset to reveal the discrepancy.

Table 4.7: Experimental results with Japan Nikkei Average Index data

Training Data	Testing Data	Rule Coverage	True positives with rules found
first 250 records	next 20 records	98%	95%
first 250 records	next 40 records	96%	90%
first 250 records	next 80 records	95%	88%
first 250 records	next 100 records	91%	86%
first 250 records	next 150 records	90%	72%
first 250 records	next 170 records	90%	71%
first 250 records	next 190 records	87%	68%

In Table 4.7, we can see that on average the procedure generates more accurate results on the Japanese Nikkei Average Index dataset that is accumulated more frequently than the previous NASDAQ Composite Index dataset. With less decision labels, the rule coverage generated from the Rough Set model increases, and consequently the forecasting is more accurate with the greater rule coverage.

In the next section, we are to simulate several datasets to see how different data distributions affect the forecasting procedure.

Synthetic & non-scaled Data

So far we have seen the performance of the forecasting procedure from the historical stock return datasets in different financial markets. In order to thoroughly investigate the limitations of the procedure that combines the *Fuzzy Set* theory and the *Rough Set* theory,

we choose to simulate some synthetic and non-scaled datasets as opposed to the return series datasets in the above two case studies.

We first generate 4 datasets with different value ranges; we are to answer the following two questions:

1. How does the data distribution affect the forecasting results?
2. What are the controllable factors in producing the useful association rules in doing the forecasting?

The datasets are listed in Table 4.8.

Table 4.8: Synthetic datasets

Bach No.	Data range/source	Scaling	No. of records
1	Nikkei Average Index	Non-scaled	300
2	[-20, 20]	scaled	300
3	[-80, 80]	scaled	300
4	[-150, 150]	scaled	300

In the first batch, we use the Japanese Nikkei Average Index dataset with normal prices rather than the return values to see if the procedure gives satisfactory results; the dataset with normal prices is illustrated in Figure 4.6.

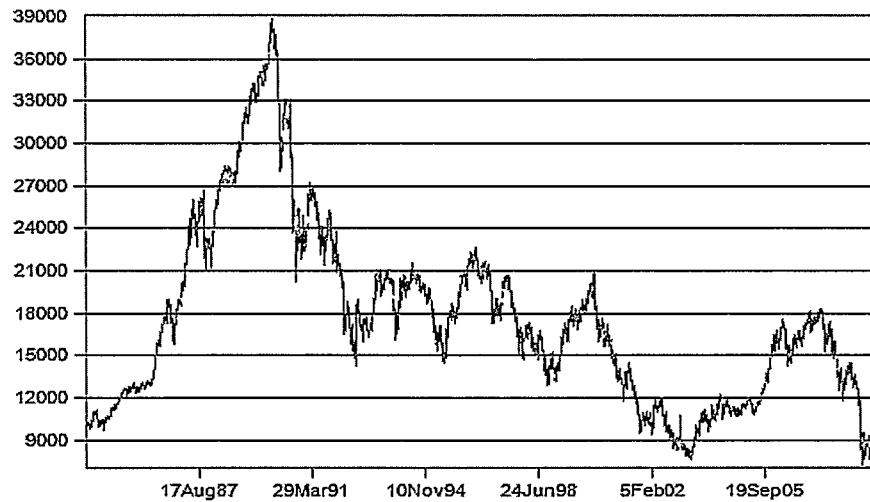


Figure 4.6: Sample of a non-scaled dataset

From Figure 4.6, we can see the dataset is strongly volatile. With 10 fuzzy sets defined, the percentages of rule coverage for the 4 datasets are shown in Figure 4.8. The training and testing datasets are split in sequential pattern, that is, in this synthetic datasets investigation, we take first 70% records of each batch sequentially to be the training dataset and the remaining 30% to be the testing dataset. The results are shown in Figure 4.7.

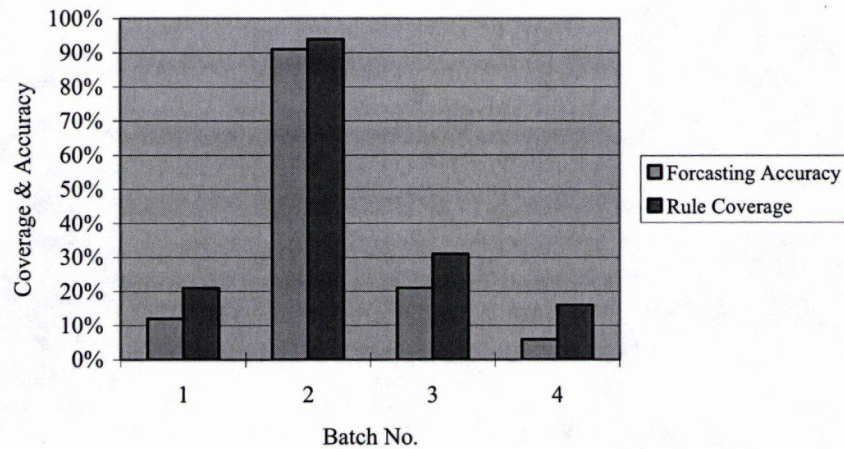


Figure 4.7: 10 Forecasting results with 4 different datasets

From the experiments, we can answer the two questions on the limitations of the procedure:

1. If the financial time series dataset is not scaled, i.e., if we use normal prices rather than the asset returns, the forecasting procedure does not perform well overall, as the 10 fuzzy sets defined fail to discretize the dataset properly. As a result, the fuzzified dataset loses substantial information compared with the original dataset and consequently, the rule-finding process with Rough Set computations cannot generate informative rules to guide the forecasting in the following stage.
2. From batches 2, 3 and 4, we can see the impact of having large and small data value ranges to the forecasting procedure: as the data range gets large, the problematic situation as with non-scaled data arises, that is, the data values are forced to be discretized into 10 fuzzy sets, leading to the uninformative rules to

forecast the unseen values. Therefore, our forecasting procedure is more suitable to less volatile dataset in general.

In this section, we summarized the limitations of the approach with the facts from the synthetic and non-scaled datasets. The sensitive nature of financial time series datasets is difficult to be dealt with in general. With our first proposed forecasting procedure, however, the limitations can be alleviated by possibly increasing the number of the fuzzy sets to ensure that the degree of lost of information in the fuzzifying process is minimized. This change is not difficult, however, since the use of the fuzzy sets mainly aims for interpretation of the forecasting results, and the number of the fuzzy sets to be used can be chosen by the end-users.

Alternative Procedure

In Chapter 3, we have proposed another forecasting procedure with selected features appending, that is, we choose a set of financial ratios believed to best represent the firms' financial situation and append them to form the Information System as input to the Rough Set model as opposed to using the statistical models in deriving the features.

In this section, we study the case of monthly simple returns of Citi-group stock. The data is accumulated monthly. Since the data is already in returns rather than normal prices, we need not compute the returns again. The time series is graphed in Figure 4.8.

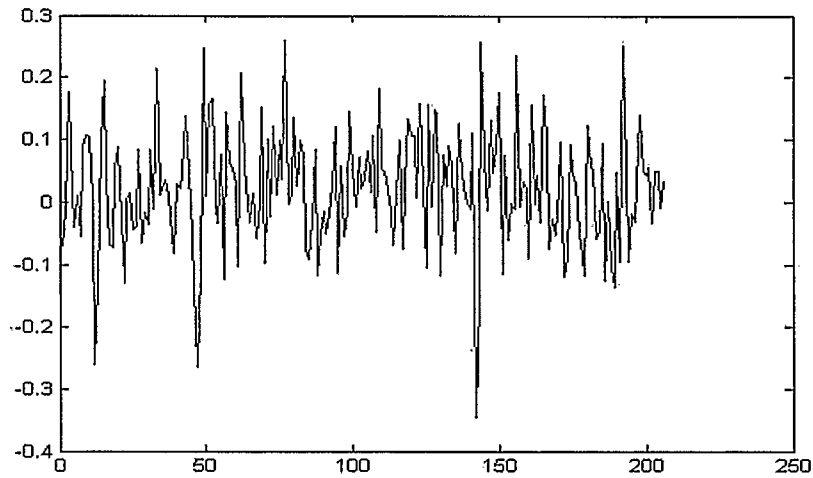


Figure 4.8: Returns of the original financial time series

In this case study, we create a set of simulated financial ratios. By simulating the ratios, we can grasp how well the approach reacts to diversified circumstances. We fuzzify the features and add the decision feature as described in step 4' and 5'. The sample input data to the Rough Set model is shown in Table 4.9.

Table 4.9: Input data to the Rough Set model with simulated financial ratios

Data	Ratio 1	Ratio 2	...	Ratio 9	Ratio 10	Decision
1	LARGE	LARGE	...	MIDDLE	LARGE	0
2	MIDDLE	SMALL	...	NORMAL	VERT LARGE	1
3	MIDDLE	SMALL	...	MIDDLE	LARGE	-1
4	MIDDLE	SMALL	...	VERY LARGE	VERT LARGE	0
5	VERY LARGE	VERY LARGE	...	SMALL	NORMAL	1
6	VERY LARGE	VERY LARGE	...	VERY LARGE	MIDDLE	buy-hold indiff.
7	SMALL	SMALL	...	LARGE	MIDDLE	not buy indiff.
...

Users can merge the patterns based on different preferences, such as limiting the number of features in the left hand side of certain rules. To do the forecasting, we need to first contemplate different training and testing dataset splits. We consider two approaches

in this investigation, randomized proportion split (pick random data records up to certain proportion with replacement) and sequential proportional split (pick data records sequentially up to certain proportion without replacement). The statistics is shown in Figure 4.9.

From the results, we can see that the randomized split approach renders randomized behaviour as expected; and sequential split approach renders progressive behaviour as bigger proportion is taken for training, in other words, hidden knowledge is more realized when more data is taken into consideration during training process.

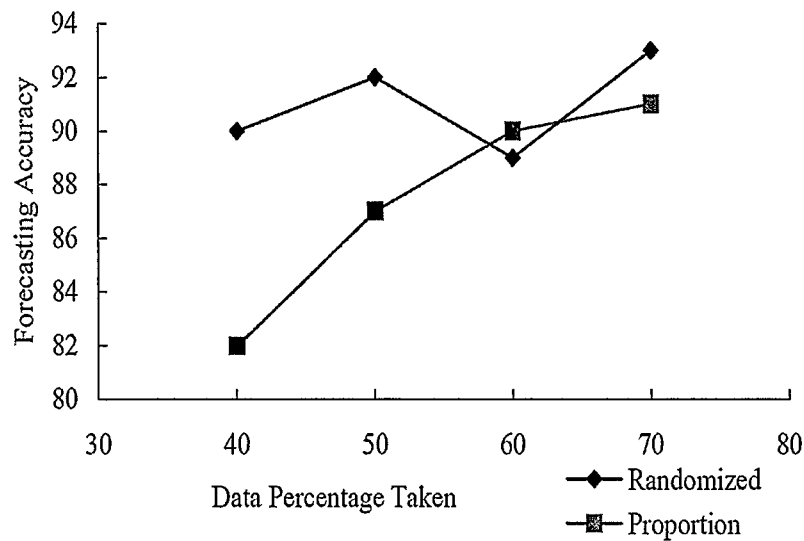


Figure 4.9: Forecasting results for different training-testing data split in percentage with confidence $\geq 60\%$

Comparative Study

In order to demonstrate the efficiency of the mining procedure centred at the Rough Set computations, we compare association rule mining with both *Apriori* algorithm and the Rough Set computation with daily return of Citi-group stock and monthly return of Microsoft stock. The timing statistics are shown in Table 4.10.

Table 4.10: Performance of Rough Set and Apriori with 2G RAM Pentium M

Data	No. of Records	<i>Rough Set</i>	<i>Apriori</i>
Citi Daily	4333	115s	560s
Citi Monthly	203	0.5s	10s
MS Monthly	213	0.5s	10s

We can conclude that the *Apriori* algorithm is less efficient because of multiple scans of datasets. This disadvantage is vital when the datasets grow large: when business data accumulates, we require efficient mining algorithms.

We measure the prediction accuracy by the *mean square error* (MSE) defined by

$$\frac{\sum_{i=1}^d (y_i - y'_i)^2}{d}$$

Where y_i is the test tuple with associated known value for a response variable y ; d is the number of tuples to use in the sample. The effect of outliers is accentuated with the squared difference, therefore this measurement is particularly useful in dealing with financial datasets where outliers are potentially hurtful to investors.

We compare the mean square errors between our approach and the method proposed in [26], the statistics shows that the MSE incurred by our approach is 2411, whereas the method proposed in [26] gives 2689. Other methods mentioned in [26]

generally generate larger MSE. We argue that the magnitude of the MSE has to do with the fuzzifying process, that is, if we are able to assign values more accurately into the fuzzy sets defined, the MSE can be significantly smaller. However, there is no rule of thumb in general when defining fuzzy sets or intervals because of the differentiated nature of knowledge domains.

Another possible contribution of this conceptual model worth mentioning is that, we can embed the analysis into real time systems to produce meaningful results to end users. For instance, in *SAP Business One*, which is designed for small business solutions, our approach can be easily implemented in the *Report* module. The financial ratios and other information of the company can be retrieved from *G/L account criteria*, as shown in Figure 4.10.

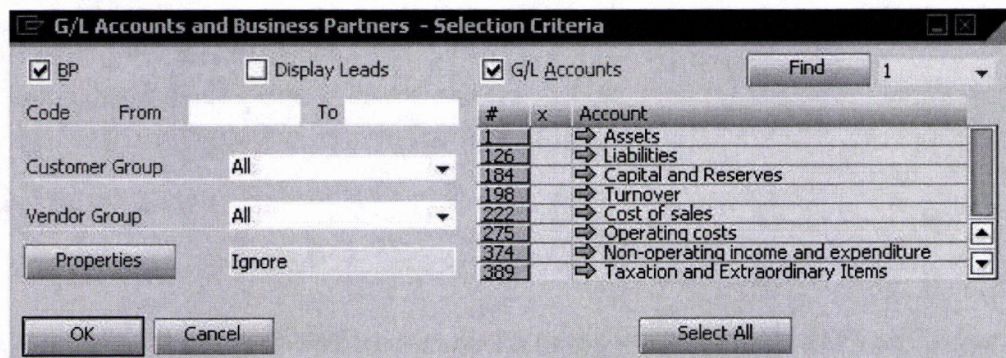


Figure 4.10: Retrieval of financial information in *SAP Business One*

In summary, our approach provides the generality in making associations between the underlying financial instruments and investment decisions. This methodology is more flexible to high-end investors. The expandability of the Rough Set model in discovering

hidden patterns is further advantageous in mining running state data with its effectiveness and efficiency demonstrated in the comparative study.

In this chapter, we have shown the work flow of the two proposed mining procedures and the limitations of the hybrid approach.

CHAPTER 5

SUMMARY, CONCLUSIONS AND FUTURE WORK

In this chapter, we summarize the findings of this thesis and discuss several directions for the further work.

Summary and Conclusions

In this thesis, we addressed the problem of finding hidden rules in the financial time series datasets and further performing financial prediction using rules found. We use two statistical models in estimating returns and gauging the volatility of the time series and fuzzy discretization process in interpreting the numerical values. The hybrid approach centers at a powerful mathematical concept, namely the *Rough Set* model in finding hidden association rules. We devised two procedures in predicting financial asset returns, and established the connection between the stock fluctuations and the firms' financial information. We conducted different case studies on different datasets showing the feasibility and the limitations of our hybrid approach.

The hybrid mining approach produces interpretable and accurate results over financial time series data. From our experimental study, we can conclude the following findings:

1. To make accurate predictions, we need to train the model with nearest datasets in terms of time period. As the training datasets age, the prediction results deteriorate accompanying the decrease of rule coverage.

2. Our experimental results reveal the correlation of the financial time series data, as in the context of the lagging factor.
3. The hybrid mining approach does not generate satisfactory results if the data is not scaled with normal prices and volatile if not used in the financial domain.
4. The overall performance of the procedures is relatively efficient compared with other traditional association rule mining techniques.

To sum up, our hybrid methodology provides an empirical approach in assisting individual investors in making investment decisions, and the main contribution is that we addressed the issue of achieving the efficiency and effectiveness in mining financial time series datasets from a cross-disciplinary perspective.

Future Work

The study developed in this thesis has revealed some interesting findings that could lead to several extensions. We plan to extend the hybrid approach presented in this thesis in a number of major directions:

1. Data Visualization - The ultimate goal is to incorporate the approach into packaged software and integrate the mining techniques for decision making processes. The embedded data work flow is shown in Figure 5.1.
2. Static vs. Dynamic data mining - In this thesis, we have only considered the static data mining techniques in finding association rules. In financial time series, however, the data is accumulated extremely frequently such as the sales and tax data in ERP systems. We need to consider incremental mining techniques to

dynamically evaluate the model as a running state. This can have important influence in real time systems.

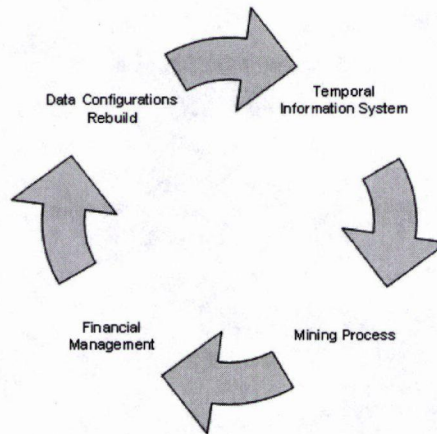


Figure 5.1: The decision making process

3. In this thesis, the hybrid approach is an abstract solution framework of mining financial time series that can be embedded into real-time systems. However, the software engineering side of the solution framework is not presented, such as the problem of retrieving data systematically in different stages and the computation runtime, i.e., if the Rough Set model is running on-demand or periodically based on user queries. This branch of research can certainly lead our hybrid methodology into the next stage.
4. With certain domain extensions, our approach can be potentially extended to address the following issues:
 - For stock value fluctuations of a company, what are the events such as war and economy recession that constitute reasons for those major financial events and to what degree?

- For static stock values in a period window of interest, how ratios from the financial statements such as *current/quick* ratio and *receivable turnover* ratio affect stock prices and to what degree? Intuitively, financial statements are the bond fide resources for investors in forming investment opinions. Consequently, the mined rules should be informative.
- In modern ERP systems, how can association rules reveal data patterns? For example, the VAT calculations are derived from multiple relations, giving difficulties differentiating the most useful financial information.

REFERENCES

- [1] T. Abeel, Y. Van de Peer, Y. Saeys, "Java-ML: A Machine Learning Library." *Journal of Machine Learning Research*, Vol. 10, pp. 931-934, 2009.
- [2] Jean-Marc Adamo, "Data Mining for Association Rules and Sequential Patterns." *Springer Verlag New York*, 2000.
- [3] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules." *In Proc. of the VLDB Conference*, Santiago, Chile, September 1994. Expanded version available as IBM Research Report RJ9839, June 1994.
- [4] R. Aliev, B. Fazlollahi, R. Aliev, B. Guirimov, "Fuzzy time series prediction method based on fuzzy recurrent neural network." *Lecture Notes in Computer Science, Neural Information Processing*, Volume 4233, pp. 860-869, 2006.
- [5] K. K. Ang, C. Quek, "Stock trading using RSPOP: A novel rough set-based neuro-fuzzy approach." *IEEE Transactions on Neural Networks*, Volume 17, Issue 5, pp. 1301-1315, 2006.
- [6] G. S. Atsalakis, K. P. Valavanis, "Forecasting stock market short-term trends using a neuro-fuzzy based methodology." *Expert Systems with Applications*, Volume 36, Issue 7, pp. 10696-10707, 2009.
- [7] G. Booch, R. Maksimchuk, M. Engle, B. Young, J. Conallen, K. Houston, "Object-oriented analysis and design with applications." *Addison-Wesley Professional*, 2007.
- [8] C. Borgelt, "Efficient implementations of apriori and eclat", *Workshop of Frequent Item Set Mining Implementations*, 2003

- [9] L. J. Cao and F. E. H. Tay, "Support vector machine with adaptive parameters in financial time series forecasting." *IEEE Transactions on Neural Networks*, Volume 14, Issue 6, pp. 1506- 1518, 2003.
- [10] Shyi-Ming Chen and Nien-Yi Chung, "Forecasting enrollments using high-order fuzzy time series and genetic algorithms." *International Journal of Intelligent Systems*, Volume 21 Issue 5, pp. 485 – 501, 2006.
- [11] J. Chen, T. Cook, "Mining contiguous sequential patterns from web logs." *In Proc. of the 16th international conference on World Wide Web*, pp. 1177 - 1178, 2007.
- [12] Cohen, L. Ralph, Hess, Kathryn, Voronov, A. Alexander, "String Topology and Cyclic Homology." *Advanced Courses in Mathematics - CRM Barcelona*, 2006.
- [13] G. Cormode, S. Muthukrishnan, "The string edit distance matching problem with moves." *ACM Transactions on Algorithms (TALG)*, Volume 3, Issue 1, 2007.
- [14] K. Czarnecki, S. Helsen, "Feature-based survey of model transformation approaches." *IBM Systems Journal*, Volume 45, No. 3, pp. 621-645, 2006.
- [15] M. G. Elfeky, W. G. Aref and A. K Elmagarmid, "Periodicity Detection in Time Series Databases." *IEEE Transactions on Knowledge and Data Engineering*, Volume 17 No. 7, pp 875-887, 2005.
- [16] J. W Guan, D. A Bell and D.Y Liu, "The rough set approach to association rule mining." *ICDM 2003*, pp 529- 532, 2003.
- [17] M. Hamada, K. Tsuda, T. Kudo, T. Kin, K. Asai, "Mining frequent stem patterns from unaligned RNA sequences." *Bioinformatics*, Volume 22(20), pp. 2480-2487, 2006.
- [18] J. Han, H. Cheng, D. Xin, X. Yan, "Frequent pattern mining: current status and future directions." *Data Mining and Knowledge Discovery*, Volume 15, Number 1, pp. 55-86, 2007.

- [19] Jiawei Han, Jian Pei, and Yiwen Yin, "Mining frequent patterns without candidate generation." *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pp. 1 - 12, 2000
- [20] H. Hochheiser and B. Shneiderman, "Dynamic Query Tools for Time Series Data Sets, Timebox Widgets for Interactive Exploration." *Information Visualization*, Volume 3, pp. 1-18, 2004.
- [21] C. J. Huang, D. X. Yang, Y. T. Chuang, "Application of wrapper approach and composite classifier to the stock trend prediction." *Expert Systems with Applications*, Volume 34, Issue 4, pp. 2870-2878, 2008.
- [22] C. P. Jones, "Investments: analysis and management." *Wiley-India*, 2007.
- [23] Eamonn Keogh, Kaushik Chakrabarti, Michael Pazzani and Sharad Mehrotra, "Dimensionality Reduction for Fast Similarity Search in Large Time Series Databases." *Knowledge and Information Systems*, Volume 3, Number, pp. 263-286, 2001.
- [24] G. Lasso, J. F. Antoniw, J. G. L. Mullins, "A combinatorial pattern discovery approach for the prediction of membrane dipping (re-entrant) loops." Volume 22(14). pg. e290-e297, *Bioinformatics*, 2006.
- [25] C.H.L Lee, W. Chen and A. Liu, "An Implementation of Knowledge Based Pattern Recognition for Financial Prediction." *Proc. 2004 IEEE Conf. Cybernetics and Intelligent Systems (CIS '04)*, pp. 218-223, 2004.
- [26] C. Lee and Alan Liu, "Pattern Discovery of Fuzzy Time Series for Financial Prediction." *IEEE Transactions on Knowledge and Data Engineering*, Volume 18 No. 5, pp 613-625, 2006.

- [27] H. F. Li, S. Y. Lee, "Mining frequent itemsets over data streams using efficient window sliding techniques." *Expert Systems with Applications*, Volume 36, Issue 2, pp. 1466-1477, 2009.
- [28] J. Lin, E. Keogh, S. Lonardi and B. Chiu, "A Symbolic Representation of Time Series, with Implications for Streaming Algorithms." In *Proc. SIGMOD Workshop Research Issues on Data Mining and Knowledge Discovery*, pp. 2-11, 2003.
- [29] Tsau Young Lin, Yiyu Y. Yao, and Lotfi A. Zadeh, "Data mining, rough sets, and granular computing." *Physica-Verlag*, 2002.
- [30] S. Lonardi, J. Lin, E. Keogh, B. Y. Chiu, "Efficient discovery of unusual patterns in time series." *New Generation Computing*, Volume 25, Number 1, pp. 61-93, 2006.
- [31] H. Liu, H. Motoda, "Computational methods of feature selection." *Chapman & Hall*, 2007.
- [32] Last Mark, "Data mining in time series databases." *Series in machine perception and artificial intelligence*, Volume 57, World Scientific, 2004.
- [33] C. D. Manning, P. Raghavan, H. Schtze, "Introduction to information retrieval." *Cambridge University Press*, 2008.
- [34] D. W. Mount, "Strategies for sequence similarity database searches." *Cold Spring Harbor Protocols*, 2007.
- [35] B. Padmanabhan, A. Tuzhilin, "On characterization and discovery of minimal unexpected patterns in rule discovery." *IEEE Transactions on Knowledge and Data Engineering*, Volume 18, Issue 2, pp. 202- 216, 2006.
- [36] S. Papadimitriou, P. Yu, "Optimal multi-scale patterns in time series streams." In *Proc.. of the 2006 ACM SIGMOD international conference on Management of data*. pp. 647-658, 2006.

- [37] Z. Pawlak, A. Skowron, "Rudiments of rough sets." *Information Sciences*, Volume 177, Issue 1, pp. 3-27, 2007.
- [38] Matthew Simon, Graham Bee, Philip Moore, Jun-Sheng Pu and Changwen Xie, "A Study on mining theory and model of product lifecycle-based running state." *Computers in Industry*, Volume 45, Issue 2, June 2001, pp. 111-122
- [39] J. Takeuchi and K. Yamanishi, "A unifying framework for detecting outliers and change points from time series." *IEEE Transactions on Knowledge and Data Engineering*, Volume 18 No. 4, pp 482-492, 2006.
- [40] Tay, E. H. Francis, "Ordinary shares, exotic methods: financial forecasting using data mining techniques." *World Scientific*, 2003.
- [41] A. P. N. Refenes and W.T. Holt, "Forecasting Volatility with Neural Regression: A Contribution to Model Adequacy." *IEEE Trans. Neural Networks*, vol. 12, no. 4, pp. 850-864, July 2001.
- [42] H. J. Teoh, C. H. Cheng, H. H. Chu, J. S. Chen, "Fuzzy time series model based on probabilistic approach and rough set rule induction for empirical research in stock markets." *Data & Knowledge Engineering*, Volume 67, Issue 1, pp. 103-117, 2008.
- [43] Ruey S. Tsay, "Analysis of financial time series." *Wiley*, 2005.
- [44] Kari Vasko and Hannu Toivonen, "Estimating the number of segments in time series data using permutation tests." *Proceedings of the 2002 IEEE International Conference on Data Mining*, pp. 466, 2002.
- [45] E. J. Wagenmakers, P. Grünwald, M. Steyvers, "Accumulative prediction error and the selection of time series models." *Journal of Mathematical Psychology*, Volume 50, Issue 2, pp. 149-166, 2006.

- [46] W. Wang, Y. Zhang, "On fuzzy cluster validity indices." *Fuzzy Sets and Systems*, Volume 158, Issue 19, pp. 2095-2117, 2007.
- [47] Jiong Yang, Wei Wang and P. S. Yu, "Mining asynchronous periodic patterns in time series data." *IEEE Transactions on Knowledge and Data Engineering*, Volume: 15, pp. 613- 628, 2003.
- [48] D. Yankov, E. Keogh, J. Medina, B. Chiu, V. Zordan, "Detecting time series motifs under uniform scaling." *In Proc.. of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 844 - 853, 2007.
- [49] T. H. K. Yu, K. H. Huarng, "A bivariate fuzzy time series model to forecast the TAIEX." *Expert Systems with Applications*, Volume 34, Issue 4, pp. 2945-2952, 2008.
- [50] W. Yu, X. Wang, F. Wang, E. Wang, B. Chen, "The research of improved apriori algorithm for mining association rules." *11th IEEE International Conference on Communication Technology*, pp. 513-516, 2008.
- [51] U. Yun, J. J. Leggett, T. Ong, "Mining Weighted Sequential Patterns Based on Length-Decreasing Support Constraints." *Intelligence and Security Informatics*, Volume 3975, pp.650-651, Springer Berlin / Heidelberg, 2006.
- [52] J. Zaki Mohammed, "SPADE: An Efficient Algorithm for Mining Frequent Sequences." In *Machine Learning Journal*, special issue on Unsupervised Learning (Doug Fisher, ed.), pp. 31-60, Volume 42, Nos. 1/2, 2001.
- [53] http://en.wikipedia.org/wiki/Data_mining
- [54] <http://www.nag.co.uk/IndustryArticles/DMinFinancialApps.pdf>