Forecasting the Direction of GCC Stock Indexes Using Support Vector Machine (SVM) Algorithm

توقع اتجاه مؤشرات أسوق الأسهم في دول مجلس التعاون الخليجي باستخدام خوارزمية آلة متجه الدعم

(SVM)

By

Yaser Abdullah Zeino

110001

Dissertation submitted in partial fulfillment of the requirements for the degree of MSc in Finance and Banking

Faculty of Business

Dissertation Supervisor
Dr. Elango Rengasamy

March 2014

Table of Contents

Introduction 1

Efficient-Market Hypothesis (EMH) 2

Machines Learning and Artificial Intelligence (AI) 2

Rationale of the Study 3

Objectives 4

Hypothesis of the Study 4

Limitations of the study 5

The structure and outlines 5

Literature Review 6

Introduction 6

Financial Forecasting Using SVM 7

Bankruptcy Forecasting Using SVM 13

SVM and Other Forecasting Models 16

Selecting the Parameters of the Forecasting Models 21

Summary, Criticisms, and Comments 24

Methodology and Data Collection 29

Theory of Support Vector Machine Algorithm 29

Support Vector Machine Mathematical Model 30

Random Forest Model 32

Logistic Regression Model 33

Sampling and Data Set 33

Technical Indicators *35*

Experimental Study, Discussion, and Results 38

Application of Support Vector Machine (SVM) *38*

Performance of Support Vector Machine (SVM) *39*

Support Vector Machines, Random Forest, and Logistic Algorithm *43*

Findings and Recommendations 48

Conclusion 52

References 53

Appendices 58

Appendix 1 .... *58*

Appendix 2 .... *59*

Appendix 3 .... *61*

Appendix 4 .... *62*

Appendix 5 .... *65*

Appendix 6 .... *70*

Appendix 7 .... *71*

Appendix 8 .... *72*

Appendix 9 .... *74*

Appendix 10 .... *75*

Appendix 11 .... *76*

Abstract

Forecasting the direction of stock markets is a very challenging task for investors and decision makers. Recently, investors in stock exchange start depending on Artificial Intelligence (AI) systems to build various investment strategies. Support Vector Machine algorithm (SVM) is an advanced technique for classification, regression, and forecasting purposes which introduced by Cortes and Vapnik (1995). This study applies Support Vector Machine to predict the direction of GCC stock indexes movement using the historical prices. Data sample in this study covers the period from 2010 till 2013. This study compares Support Vector Machine (SVM) with other classification methods such as Logistic Regression and Random Forest. This study suggests that Support Vector Machine (SVM) perform well in predicting the direction of indexes movement in GCC stock markets where accuracy average rates are between 72.06% and 83.42% in all markets. In addition to that, Support Vector Machine outperform Random Forest algorithm in predicting the direction of GCC stock indexes. However, Logistic Regression outperforms Support Vector Machine and Random Forest in all GCC markets. The findings of this study suggest that applying SVM and other Artificial Intelligence techniques in the trading systems might attract more investors and bring more commission to the brokerage companies in GCC stock markets.

توقع اتجاه أسواق الأسهم هي مهمة صعبة للغاية بالنسبة للمستثمرين وصناع القرار . في الآونة الأخيرة ، بدأ المستثمرين في البورصة بالاعتماد على نظم الذكاء الاصطناعي (AI) لبناء استراتيجيات الاستثمار المختلفة. خوارزمية آلة متجه الدعم (SVM) هي تقنية متقدمة لأغراض التصنيف ، والانحدار ، والتنبؤ الذي عرضه كورتيس وفابنك (1995) . تطبق هذه الدراسة خوارزمية آلة متجه الدعم (SVM) من أجل توقع اتجاه حركة مؤشرات الأسهم في دول مجلس التعاون الخليجي باستخدام أسعار تاريخية. تغطي هذه الدراسة عينة البيانات في الفترة من 2010 وحتى 2013. تقارن هذه الدراسة خوارزمية آلة متجه الدعم (SVM) مع أساليب أخرى مثل طريقة الإنحدار اللوجيستي في التصنيف و طريقة الغابات العشوائية . تقترح هذه الدراسة أن خوارزمية آلة متجه الدعم (SVM) أداة جيدة في توقع اتجاه حركة مؤشرات الأسهم في دول مجلس التعاون الخليجي حيث متوسط معدلات الدقة ما بين 72.06 ٪ و 83.42 ٪ في جميع الأسواق. بالإضافة إلى ذلك، تتفوق خوارزمية آلة متجه الدعم (SVM) على طريقة الغابات العشوائية في تصنيف وتوقع اتجاه مؤشرات الأسهم الخليجية . ومع ذلك ، تتفوق طريقة الإنحدار اللوجيستي على خوارزمية آلة متجه الدعم (SVM) و طريقة الغابات العشوائية في جميع أسواق دول مجلس التعاون الخليجي . وتشير نتائج هذه الدراسة إلى أن تطبيق خوارزمية آلة متجه الدعم (SVM) وتقنيات الذكاء الاصطناعي الأخرى في أنظمة التداول قد يجذب المزيد من المستثمرين و بالتالي المزيد من العمولات لشركات الوساطة في أسواق الأسهم الخليجية .

Acknowledgment

In the Name of Allah Most Gracious Most Merciful. Allah said in Quran: “but only say, ‘O my Lord, increase me in knowledge”. The prophet Muhammad peace be upon him said: "He who does not thank people, does not thank Allah". Herewith, I would like to express my special thanks to my Professor Dr. Elango Rengasamy who gave me valuable support during the program and dissertation. Dr. Elango has helped me to improve my research skills and my thinking about banking and finance. Secondly, I would like also to thank my parents who were praying for me until I got the Master Degree in Banking and Finance. Special thanks go for my wife Douaa who was beside me in all times with our son Fares. Thanks also to all my friends and colleagues in the university and work.

Introduction

Forecasting the direction of stock market is a popular and important task in the financial sector and academic research. Many factors are causing the fluctuation in the time series for the stock index. These factors include the economic and political events, investor's expectations, war, and the relationship with other time series. So, predicting the direction of stock index is quite difficult for both investors and portfolios managers. Technical analysis and time series analysis are well known methods to achieve this goal. Trading in the financial markets has gained a lot of interest especially in the emerging markets since the last decade. Generally speaking, there are two schools of thoughts of forecasting the financial market. While the fundamental analysis focusing on specific aspects that have critical impact on the company actual business and its strategic goals, technical analysis on the other side examine the price movement of a stock index and employ the historical prices to predict the direction of the stock index in the future. In the financial markets, the accuracy of predicting the stock prices become very important vector for investors and decision makers to develop proper financial strategies and prevent the risk of superior losses.

A few years ago, most of the forecasting models were depending on conventional statistical methods such as the time-series models and multivariate analysis. However, in the recent years, investors start looking for advanced methods that are more accurate for them and can be a challenge for Efficient Market Hypothesis (EMH). In the emerging stock markets such as GCC financial markets, stock or index prices are very difficult to predict using the statistical methods such as leaner regression, multi regression, least squares, and other traditional models because of their chaotic nature. Conventional statistical techniques had many issues because of the relationship between the assets value and its elements changes over the time (Chen and Shih 2006).

There are two types in the financial time series prediction. The first category is the multivariate analysis where the predictor can consider any technical indicator as the input variable in the model regardless of its relationship with the output variable. However, in univariate analysis the input variable has part of the time series under forecasting process.

The overall objective of the forecasting methods is to build strategies for investors that can increase their profits and reduce the potentially risk. This research can help the investors in the GCC to build their financial strategies using advanced forecasting techniques, which depends on the learning machines and artificial intelligence (AI). Artificial intelligence has been shown in many research to have excellent accuracy performance overall the global and international stock markets. However no research has been made to experiment the learning machines algorithm on GCC stock data which can be a challenge for the Efficient Market Hypothesis (EMH).

Efficient-Market Hypothesis (EMH)

Efficient market hypothesis (EMH) suggests that stock prices are reflecting all types of information available for public and private. Efficient Market Hypothesis was proposed by Fama (1970). According to him, it is very difficult for normal investors to profit from the trading in the stock markets and benefit from the abnormal changes of the stock returns. Fama has categorized the efficient market hypothesis (EMH) to three forms. The first form is the weak form efficiency, which suggests that employing investment strategies using the historical prices of the stock cannot make excess returns. This is lead to say that stock market return is not predictable using a technical analysis approaches. The second form of the efficient market hypothesis is the semi-strong form efficiency which suggests that the changes in the stock returns is reflected by the historical information and the stock information which is available to public. The third form is the strong-form efficiency, which means that the stock prices are reflecting all type of information including the private information, which might be accessible by certain people. This form implies that investors cannot predict the stock return and earn more profit by using this kind of information. To sum up with the Efficient Market Hypothesis, investors cannot generate excess return using the historical information.

Machines Learning and Artificial Intelligence (AI)

Machine learning is a programing algorithms used to optimize a performance of a certain method using data sample or historical information. Researcher needs to learn the sample of data where he or she cannot directly solve a given model. There are many applications of the learning machine methods in the business fields. For example, Learning Machines can be used to analyze the past sales for a commercial company or examine the behavior of their customers to improve their revenue. In the financial sector, machine learning was used to analyze the past transactions in a bank and predict customers' credit risks. In addition to that, many researchers have introduced the machines learning algorithms as a solution to forecast the direction of the stock markets based on the historical prices. Recently, investors in stock exchange start depending on Artificial Intelligence (AI) systems to help them in forecasting the stock prices and build a various investment strategies based on the historical movement of the stock prices. Therefore, the most important aspect in any investment strategy is the accuracy of the forecasting model.

Rationale of the Study

Forecasting the direction of stock markets is a very challenging task for investors and decision makers because of the inherent characterizes of the time series. Investors in the stock exchanges, dealers, and portfolios managers who have access on the historical prices need to hedge against potential risks and make profits from the trading in stock markets. Investors and portfolios managers are depending on the traditional statistical tools to forecast the direction of the stock indexes. So, the accurate prediction of the stock direction is very important for both investors and decision makers in the financial markets. This study is important because it gives the investors and market participants an advanced classification method such as Support Vector Machines to examine the performance and the accuracy of their forecasting models. In addition to that, it will provide the brokerage and securities companies in the GCC with an advanced forecasting algorithm where they can implement it in their trading system and make it available for the online traders to attract more investors.

Also, the rational of this study is to propose the Artificial Intelligence algorithms such as Support Vector Machines (SVM) as an advanced classification solution for investors in the GCC stock markets to test the performance of their forecasting models. Moreover, investors and portfolios managers are implementing different trading strategies, so the forecasting model with minimum error may not satisfy their expectations. This is because the trading that is driven by the best forecasting model in term of forecasting error may not be profitable as the trading based on the accurate prediction of the stock market direction. Therefore, it is very important for the investors and market participants to forecast the direction of the market indexes in order to build their trading strategies. This study will help investors and market participants to develop their financial strategies based on the accuracy of predicting the market direction. This study is also important because it is the first study that applying and evaluating the Support Vector Machines and other Artificial Intelligence (AI) classification algorithms to forecast the direction of GCC stock indexes. This study will answer two questions whether the stock direction is predictable using Support Vector Machine and whether technical indicators are efficient to predict the stock direction.

Objectives

In light of the above scenarios, the objectives of this study are follows:

- To apply the Support Vector Machines (SVM) as an advanced algorithm on time series of the GCC stock markets in order to develop a simplified method for financial prediction.
- To develop a classification model for segregating index direction into two categories UP or DOWN for specific period based on its technical indicators.
- To examine the performance of Support Vector Machines in forecasting the direction of GCC stock indexes.
- To critically compare the Support Vector Machines with other learning machines algorithms such as Decision Trees, and Logistic Algorithm.
- To evaluate as to analyze the accuracy of forecasting models for each algorithm for the selected indexes under study.

Hypothesis of the Study

The main two hypotheses for this study are follows:

H1: the Support Vector Machine does not perform comparing to other types of learning machines

H0: the Support Vector Machines outperforms comparing to the other types of learning machines

H2: Sample testing selection does not have significant impact on the accuracy hit ratio using Support Vector Machines

H0: Sample testing selection has significant impact on the accuracy hit ratio using Support Vector Machines

Limitations of the study

Predicting the direction of stock index is an important task for investors and portfolios managers. This study faces challenges in selecting technical indicators which contribute on the daily movements. Data downloading in a correct format and same periods for all GCC markets was very difficult. In addition to that, holiday's time table was not the same for all markets because it is following the courtier's policy and the number of trading days was not the same for all markets. However, this study applied the forecasting model for each market individually for the same period.

The structure and outlines

This study organized as follows. Chapter 2 presents the literature review of Support Vector Machine (SVM) and other learning machines in the financial field. It is followed by the methodology and data collection in chapter 3 which explains the model structure in details and presents the sampling methods and the three forecasting models under study. Then, chapter 4 discusses the experiential study and the testing results. Chapter 5 summarizes the findings and recommendations of the study. Conclusion is presented in chapter 6.

Literature Review

Introduction

Over the last ten years, little attention has been given to the emerging stock markets such as GCC stock markets. In recent years, however, the prediction of market direction using Learning Machines algorithms has gain a lot of interests in the academic studies and the financial markets because of the rapid growth in stock markets investors. However, few works has been done on forecasting the movement of stock markets index's using advanced methods such as Learning Machines algorithm and Artificial Intelligence (AI). There is strong evidence that the accuracy of forecasting models might have big impact on the investor's financial strategies. In addition to that, the predicting of future returns using the traditional techniques might be not sufficient for portfolios managers to create profitable financial strategies for the future. Therefore, the accuracy and the performance of the prediction methods are very challenging tasks for academic researches to stand on the market efficiency hypothesis. To evaluate the performance of forecasting techniques using different periods and different data samples, academic researchers have focused of the advanced financial markets. Some findings suggest strong evidence on the accuracy of the financial techniques, while others show a poor indicator on the performance of the forecasting methods. There are massive researches which forecast the direction of stock indexes as well as the stock returns. However, most of the forecasting methods focusing on the stock return prediction. Few studies have been done on the stock index direction forecasting. This section discusses different studies on the advanced forecasting techniques in the financial markets. More works have been done on the Learning Machines algorithms as new methods in the advanced financial markets. The literature reviews in this study focus primarily on testing the performance of Support Vector Machines (SVM) in the stock markets and on its forecasting accuracy. This section is divided into four subsections. The first one is covering the studies which have been done on the financial time series forecasting using Support Vector Machines. Then, the second subsection is discussing the predictability of Support Vector Machines in forecasting the bankruptcy and credit rating. The third subsection compares the Support Vector Machines (SVM) with other forecasting and learning machines. The forth subsection covers the application of Support Vector Machines on the credit rating analysis. The forth subsection covers the studies done on selecting the forecasting model parameters and features. The fifth subsection includes the comments, criticism, and the contributions of previous studies and this study in the financial sector.

Financial Forecasting Using SVM

Financial forecasting is one of the most complicated and important task in the modern financial markets. As explained by the previous studies, financial time series are noisy, non-stationary. These features suggest that there is no complete information that could be extracted from the past data and explained the dependency between future and current prices. However, recent studies refer that learning machines as advanced forecasting methods are very affective in capturing the useful information and unwanted noise in the time series. For example, the application on learning machines to categorize the announcement in the stock markets has been tested by Williams and Calvo (2002). They compare the performance of many learning machines algorithm on the automation of corporate announcement in the Australian Stock Exchange (ASX). Two tests have been conducted to show the categorization process in the stock exchange. The first one is to the test market sensitivity to the announcement such as take-over announcement and its impact on the index movement. The second test is the report type such as annual report, which classifies the announcement into different reports. Three types of learning machines have been used to compare the performance for each model including, Neural Network, Naïve Bayes classifier, and Support Vector Machines. Data set has been divided into training and testing samples with total of 136630 announcements and reports. The task was to test whether the stock market is sensitive or none-sensitive to the announcement and reports. The main results of their research implies that 88% of the announcements have been categorized correctly using Neural Network algorithm comparing to 80% of Support Vector Machines which indicates that Neural Network (NN) outperforms of Support Vector Machines (SVM) in the categorizing the announcements.

In the contrary, Cao and Tay (2001) suggest that the performance of SVMs in financial forecasting is better than Back Propagation algorithm (BP). They compared the performance of both algorithms based on several criteria such as the Normalized Mean Square Error (NMSE), Mean Absolute Error (MAE), Directional Summary (DS), Correct Up (CP) trend and Correct Down (CD) trend for the daily historical prices of S&P 500 general index. They divided the data sample into two parts. The first part is the training data that covers 9 months period, while the other parts used for testing purpose that covers 10 months historical prices. Two groups of variables have been used as inputs to the forecasting algorithms. The contents of group one is four lagged RDP values using 5 days period and the 15 days Exponential Moving Average (EMA) of the closing prices. The second group includes three technical indicators such as Moving Average Convergence Divergence (MACD), On Balance Volume (OBV), and volatility. The rationale behind these two groups is to examine the accuracy of multivariate analysis. The results of their research indicate that SVMs outperforms comparing to the BP. This is because that there are a small number of free parameters in the SVMs compared to BP. in addition to that, multivariate analysis using MACD, OBV, and volatility does not improve the prediction ability of Neural Networks.

It is more interesting to mention also that many authors used the Support Vector Machines to predict the volatility of the stock markets. Cruz1, Afonso, and Giner (2003) compare the Support Vector Machines (SVMs) with Maximum Likelihood (ML) to forecast GARCH model parameters which are using for predicting the volatility of stock market returns. According to them, the efficiency of Support Vector Machines model in forecasting the volatility of market return does not associate with the probability density function (PDF) over the time series. Therefore, if the distribution of sample data is not following Gaussian distribution, support vector machines can lead to better forecasting than the least squares of ML method. The used a high-frequently time series information for four international stock market indexes such as S&P100, FTSE100, IBEX35, and NIKKEI. The empirical results of their study indicate that the prediction accuracy given by Support Vector Machines (SVMs) is better than the one given by ML estimation approach.

However, some of the studies claimed that forecasting the direction of stock index can generate better result that forecasting the underline value of the share and its return. Kim (2003) applies Support Vector Machines to forecast the direction of Korean stock market index. He compares the performance of Support Vector Machines with other classifications models such Back Propagation (BP) and the nearest-neighbor method Case-Based reasoning (CBR). He used technical indicators as input variables on the daily stock price changes to predict the direction of the general index. He applied 20% on the total sample as test sample and 80% of the data as training sample. Polynomial kernel and Gaussian radial functions have been used to examine the parameters which cause under-fitting problems for Support Vector Machines. These parameters are including the upper bound C and the Kernel parameter. The result of his research suggests that the prediction performance of Support Vector Machines is sensitive to the values of both parameters upper-bound and kernel function. According to this finding the predictability of SVMs might increase if the optimum parameters of SVM are selected. He concluded that support Vector Machines proposes a capable alternative for the financial time series forecasting.

In addition to that, Support Vector Machines has been used widely to forecast the commodities index. Harland (2002) applied Support Vector Machine on aluminum Futures contracts in London Metal Exchange (LME). According to him, support vector machine is replacing other classification methods in forecasting and pattern recognizing tasks. Support Vector Machines has driven from the advanced statistical learning theory and easy to understand and apply on the financial sector. He has developed several forecasting models to come up with an optimized model that provides evidence on the aluminum market efficiency. He applied the Support Vector Machines on LME provisional daily closing prices. He divided the data sample, which covers 13 years period into training, and validation samples. While the training sample covers closing prices of 2136 days, the validation sample covers 400 days. He kept the maximum numbers of inputs per model to six and the standard log returns used as dependent variable. The results of his research suggest that Aluminum trading prices are inefficient and it is predictable using machine learning algorithm. Therefore, creating optimized forecasting model from sub-models should result in a usable trading system for LME.

The advantages of using Support Vector Machines have been explained by its ability to predict the model parameters of the testing sample. Yuan (2011) argued that traditional forecasting models are unable to predict the index direction due to out-of-sample issue and parameters instability. He claimed that Support Vector Machine (SVM) is overcome these issues and can build valuable investment strategies. He applied Support Vector Machine to forecast the returns of S&P 500. He compared Support Vector Machine with conventional predictive methods to create a new strategy of forecasting in stock markets and to evaluate the performance of this model. To forecast the equity risk premium, he selected different microeconomic variables such as Dividend Price Ratio (DBR), Earning Price Ratio (EPR), Stock Variance, Book To Market Ratio (BMR), and Default Yield Spread (DFY). He applied the algorithm on monthly information for around 72 years excluding the crises period on 2008 till 2010. The experiment result of SVM recorded 86.1% perdition accuracy out of the total sample. According to him, Support Vector Machine outperforms other traditional predictive models in terms of accuracy and out of sample issue.

Nevertheless, Dutta, Bandopadhyay, and Sengupta (2012) confirmed that the Logistic Regression algorithm (LR) has more powerful to forecast the time series. They used the Logistic Regression algorithm (LR) and different financial ratios as input to the model to examine the impact of financial indicators on Indian stock performance. They applied 8 financial ratios including Change in Net Sales, Sales Net Assets Ratio, Price-Earnings per share ratio, Price-Book value, and Book Value on 30 selected shares of Indian stock market for data covers 4 years period. The objective of their study was to classify the selected companies into GOOD or POOR based on their rate of return (ROR). According to them, Logistic Regression compared to other learning machines algorithm can enhance the ability of investors to forecast stock price. However, their study did not take into consideration the micro economic variables and its influence on the share return. The result of their study indicates that Logistic Model can classify up to 74.6% from the selected companies correctly into GOOD or POOR so it is possible for investors to predict the good shares if they select the correct financial ratios.

To confirm the performance of the Support Vector Machines (SVM) in predicting the stock markets, some authors compared its performance with other learning machines. For example, Trafalis and Ince (2000) compared the Support Vector Machines algorithm with Backpropagation (BP) and Radial Basis Function (RBF). They applied the SVM on daily stock prices data to forecast the investment returns for IBM, Yahoo, and America Online. Data sample divided into two set including the training and testing sample. The forecasting period was very short to get the accurate result of the forecasting models. Matlab used to implement the algorithm and apply the model. However, authors have faced few issues during the implementation. The main problem was to determine the major parameters of the model such as the kernel parameters. According to them, a support vector machine is a robust algorithm for financial forecasting.

On the same side, Gavrishchaka, Ganguli (2003) have identified the limitations of conventional volatility models for predicting foreign exchange and stock index volatility. According to them, Support Vector Machines is an effective method for extracting and implementing historical prices from multiscale and high-dimensional market data. They sate that support vector machines is expected to be the greatest forecasting method for emerging markets and high-volatility time series. They have differentiated the volatility models into two categories. The first one is the deterministic models, which consider the function of conditional returns variance such as the autoregressive conditional heteroskedastic ARCH, and GARCH models. However, the stochastic models describe the volatility stochastic procedure. The main disadvantage of the conditional volatility models is their inability to capture the heterogeneity of transactions executing at different time series. The main result of their research suggest that Support Vector Machine avoid the restrictions of other conventional volatility forecasting models in handling high-dimensional and time series length.

In the contrast, Yang, Chan, and King (2002) state that applying Support Vector Regression model overlook the choices of margin setting. According to them, Support Vector Machines algorithm is an extension of the standard regression forecasting method. They have applied support Vector Machine to forecast the volatility of Hong Kong's Hang Seng stock index (HIS). They have provided new methods for minimizing the risk including Fixed and Asymmetrical Margins (FAAM) and Non Fixed and Symmetrical Margins (NASM). Following to previous research, they determine the model parameters such as "c" the cost of error, kernel function, and the margins. Their experiment result suggests that using Support Vector Machine investors can predict the standard deviation and indeed the volatility of market returns in the HIS stock market.

Support Vector Machines (SVM) can generate extensive profit for short-term investors. This statement has been confirmed by many authors. For instance, Wang and Choi (2013) state that predicting the direction of stock index may benefit short-term investors and can be used as an early financial distress warning system for long-term investors. They applied learning machine methods such as Principal Component Analysis (PCA) and support Vector Machine to forecast the direction of Korean and Hong Kong stock movement. They claimed that PCA and SVM can be used along with the economic factors for time series training and forecasting task. They conducted ten years historical prices to predict next day stock direction. According to them, the main disadvantage of support Vector Machine (SVM) in predicting the direction of stock index is that the input variables lie in a high-dimensional feature space where the issue in variables storing the data computing time. So, the objective was to reduce the dimensions and the variables in order to achieve an efficient and discriminative representation before conducting the classification task. The results of their research show that both PCA and SVM as integrated model provides high hit ratios in predicting the movement direction of KOSPI and HIS stock index. Also, the result suggests that the co-movement between the Korean and Hong Kong stock markets and American stock markets is affecting the predicting results.

On the other side, Fan and Palaniswami (2001) state that fundamental accounting has to be used beside the price information in order to achieve good investment strategy. They used the fundamental accounting and price information to beat the Asutralian Stock market using Support Vector Machines. The objective of their study was to identify stocks that have exceptional profits compared to other stocks. According to them, implementing the Structure Risk Minimization will make SVM as promising model in selecting the classifying the stocks. Accounting information trained using the model and stocks classified based on stocks return calculated from stock prices. This information includes 8 indicators used as input vectors calculated from the financial reports and the available historical prices. The experiment result shows that using Support Vector Machines as classification method can produce 208% return over the testing period compared 71% of the benchmark.

There is strong evidence that Support Vector Machine (SVM) can be an effective tool to classify the outperform stocks in the oil sector. King, Vandrot, Weng (2010) show that there are no strong assumptions and over-fitting on the sample in the Support Vector Machines because it is not empirical error minimization model. They applied the model on the oil sector in S&P500 index. They have selected well-known companies from the index such as Exxon Mobil and Hess Corp. the objective was to use the Support Vector Machines in classifying the performance of the stock in the oil sector against the performance of other stocks in order to make a valuable investment decision. They included the historical prices, trading volume, historical oil prices, P/E ratio, total assets ratio, and enterprise value as input variables to the model. According to them, Support Vector Machines classify the stocks within the oils sector with a good rate of accuracy around 52%. The following subsection discusses the previous research of the application of Support Vector Machine (SVM) in credit rating and bankruptcy forecasting.

Bankruptcy Forecasting Using SVM

The credit risk information and credit rating techniques benefit most of the player in the financial markets including issuers, traders, investors, regulators and other contributors in the financial market. Therefore, it is required for all these participants to forecast and predict the bankruptcy of the firms in the financial markets. Support Vector Machine has been used widely to build the credit rating systems. The work of Shih and Chen (2006) suggests that the rating model built using the classification method Support Vector Machine (SVM) is more accurate than the benchmark Back propagation (BP) model. The two researchers compare Support Vector Machine method with propagation neural network (BP). The main objective of their study is to test SVM performance in the classification of credit rating using different variables such as subordination, the scale, and the period of issue. Authors use multi-year data set in their study. The rationale behind this selection is that rating agencies use multi-year data as the benchmark for their decision. They divided their data set into two parts, the first part used for training and validation the sample and prevent the over-fitting issue in the testing. While the second part used for testing and prediction. The total sample used for training and validation were 73 and 26 sample used for running the test. The overall result refers to accuracy rate of 84.62% for the Support Vector Machine which indicate the BP outperforms of SVM comparing to other classification models.

It is generally believed that Support Vector Machines (SVMs) has been used by many researchers in the credit rating analysis. Huanga,, Chena, Hsua, Chenb, and Soushan Wuc (2004) have applied the Support Vector Machines on the corporate credit rating to help the bond investors and debt issuers in measuring the riskiness of the companies and bonds. According to them, the overall purpose of the credit rating prediction is to develop methods that can extract information of credit risk evaluation from the past observations and using it in predicting the credit risk in the future. They used back propagation neural network (BNN) as benchmark to examine the accuracy of the prediction models. They used two data sets from both United States and Taiwan to apply the Support Vector Machines on the credit rating analysis. The data covers 74 ratings in Taiwan and 265 ratings in U.S. Different variables have been select to obtain the testing. These variables include the financial ratios and two balance measures used in bond rating such as the total assets and total liabilities. The results of the study indicate that Support Vector Machines archived 80% accuracy compared 75% of back propagation neural network.

However, some authors claimed that transparency might affect the result of credit rating whatever the advanced technique used for rating. Auria and Moro (2008) used Support Vector Machine as an alternative method for company rating. They have compared this classification method with other traditional techniques such as Logistic Regression (LR) and Discriminant Analysis (DA). The Support Vector Machines can be used for classification purpose in the case of non-regularity in the series on unknown distribution. They claimed that the lack of transparency of the results might affect the Support Vector Machine performance. For example, Support Vector Machines cannot rate all companies as a simple parametric function of the financial indicators. So, the contribution of each financial indicator in the rating score might be different. The empirical results indicates that support vector machines outperforms other classification methods in classifying companies into solvent or non-solvent companies

Nevertheless, Shin, Lee1, and Kim (2005) show this limitation can be avoided by including many variables in the classification method. They investigate the performance of Support Vector Machines in predicting the corporates bankruptcy. According to them, Back-Propagation neural network (BP) outperforms other classification methods in recognizing patterns for the time series. However, it has limitation of loading too many training sets to find the weights of each attribute in the network. They argue that support vector machines solve this issue by capturing geometric characteristics of feature space without developing weights of the network from the training data. Support Vector Machines (SVM) considers the structural risk minimizing principle. This lead to say that as numbers of support vectors increase, number of training errors decrease and increase the risk of sample over-fitting. They applied the model on non-audited medium-size manufacturing firms provided by Korea Credit Guarantee Fund. They have selected 250 financial ratios and applied t-test to test the independency of the variables in the sample. Their experiment result indicates that the performance of Support Vector Machines increase as the number of training sets decreases. Therefore, the SVM has the highest level of accuracy compared to other classification methods like BP as the training sets decrease. The accuracy rate of the support vector machines has been compared to other models hit rations. The next subsection discusses the previous research which compared the support vector machine (SVM) with other forecasting tools.

SVM and Other Forecasting Models

There are vast academic studies which compared the performance of Support Vector Machines (SVM) with other conventional and classification methods. The work of Chen, Shih, and Wu (2006) compared the algorithm of Support Vector Machines (SVMs) with Back Propagation (BP) neural networks in forecasting the Asian stock markets indexes. According to them, most of the prediction methods were based on conventional statistical techniques such as time series forecasting techniques. However, more attention in the recent years was given to the Artificial Intelligence and Artificial Neural Networks algorithms. They have tested six Asian stock indexes including Nikkei 225 (NK), Hang Seng (HS) and Straits Times (ST). The two authors apply the same method of Cao and Tay (2001) in comparing the performance of Support Vector Machines (SVMs) with Back Propagation (BP). However, the results of their research criticize the results of Cao and Tay (2001) research which indicates that SVMs and BP are outperform compared to the other Artificial Intelligence algorithms such as the benchmark (AR) when it is applied in the Asian stock indexes.

In the contrary, Kumar and Thenmozhi (2006) argue that minimizing the prediction error does not lead to a capital gain. They applied Support Vector Machines and Random Forest to forecast the direction of S&P CNX NIFTY index. They followed the same method of Kim (2003) in using the technical indicators and the daily prices as input to the model. They divided the sample of 5 years data into two periods. Around 4 years were used for the training purpose and model estimation and one year for evaluation and testing. They have categorized the direction of the stock index into 0 for down in the price and 1 for up in the prices. They compared the Support Vector Machines with other classification models such as Random forest, Linear Discriminant Analysis, Logit Model, and Neural Network. Result if their study indicates that SVM outperformed Random Forest, Neural Network, and other classification models. The reason behind that is the

SVM applies the structure risk reducing principle which reduces the upper bound of the generalization error rather than minimizing the training error.

On the same side, Shah (2007) compared the Support Vector Machines with other methods when he applied the Support Vector Machines to predict the direction of stock indexes before any announcement. He conducted Machine Learning Algorithms such as Decision Stump, Linear Regression, and Support Vector Machines to recognize the patterns in the stock prices. He applied three technical indicators such as Moving Average (MA), Exponential Moving average (EMA), Rate of Change (ROC), and Relative Strength Index (RSI) as input variables to the testing model. According to him, news articles might affect detecting and determining the patterns in the stock price. He applied more weight on the articles which comes from the credible sources and news headlines on two big companies Google and Yahoo. The result of his research confirms that Support Vector Machine outperforms other learning machines.

Nevertheless, Yu, Wang, and Lai (2005) examined the complexity of Support Vector Machines for variables selection. He used the Support Vector Machine to identify the direction of stock markets using historical prices of S&P 500 stock index. One of the steps to build the forecasting model is to reduce the complexity of Support Vector Machine. This has been done using Genetic Algorithm (GA) when they select the inputs variables. According to them, Support Vector Machine (SVM) is outperforms other classification and forecasting methods such as ARIMA and BP. In addition to that, the complexity of support vector machines tested by Bildirici and Ersin (2013) when they applied it to examine the performance of GARCH models in forecasting the volatility of stock prices can be tested using Support Vector Machines algorithm. They have applied the model on daily returns of Istanbul stock index ISE100. According to their research, volatility clustering, irregularity, and nonlinearity characteristics can be more efficiency using Support Vector Machines compared to GARCH. A Diebold-Mariano equal forecast accuracy test has selected the Support Vector Machines (SVM) over the GARCH model. The result of their research refers that Support Vector Machine record the lowest error in out-of-sample testing. However, the null of forecast accuracy hypothesis cannot be rejected for all-time series of the shares in ISE100 stock market.

There are few studies which compared the Support Vector Machine (SVM) with other learning machine and classification methods. For example, Cao and Tay (2003) test the feasibility of SVM application on time series forecasting. They compared the performance of Support Vector Machines (SVM) with Back Propagation (BP) neural network and the Regularized Radial Basis function (RBF). They have focused in this study on free parameters of the Support Vector Machines to evaluate its performance. Their data collection contains five real futures contracts from Chicago Mercantile Market. According to them, adaptive parameters can accomplish higher generalization performance when it is applied in financial forecasting. From point of their view, there are three unique criteria of using Support Vector Machine model compared to other learning machines. The first one, it is using a set of linear functions in a high dimensional space when it is estimating the regression. Second, it takes the risk minimization into consideration for the regression estimation by using loss function. Third, it is using a combination of empirical error and regularized term for minimizing the risk function. The empirical results of their study suggest that Support Vector Machines outperform other learning machines and provides an alternative tool to Back Propagation (BP) in forecasting time series. Moreover, Abraham, Philip, and Saratchandran (2003) investigate the application of learning machines on the stock market chaotic behavior. They used several algorithms such as Levenberg Marquardt, Support Vector Machine (SVM), Takagi-Sugeno neuro fuzzy, and Boosting Neural Network on the Nasdaq and S&P CNX NIFTY stock data. They divided the sample of 7 years monthly data into two equal samples for training and testing propose. The objective of their research was to develop an efficient forecasting model to predict the following date stock price using the opening price, closing price, and high price. According to them, opening and closing price have the highest impact on the next day share price. The experiment result of the study indicates that SVM outperforms other algorithm in term of RMSE values, training time, and highest correlation coefficient.

In the contrast, Kara, Boyacioglu, and Baykan (2011) compared the performance of Artificial Neural Network (ANN) and Support Vector Machines (SVM) in forecasting the direction of daily index for Istanbul Stock Exchange (ISE100). According to them, the accuracy of predicting models may result profits for investors. They used ten technical indicators as input variables for the two classification methods. They selected daily closing prices for ISE100 index which covered 10 years period. Their experiment results indicate that training performance of ANN model varied between 84.53% and 99.64% while the performance of Support Vector Machines varied between 71.17% and 82.85%. According to these results, they conclude that both algorithm ANN and SVM recorded significant performance in predicting the direction of ISE100. However, ANN model outperform Support Vector Machine on the average of predicting accuracy.

Some of the previous studies, however, showed that there are limitations in applying the learning machines algorithm on the stock market information. For instance, Yildiz, Yalama, and Coskun (2008) state that the reasons behind using the Artificial Neural Networks (ANN) as an alternative to the conventional forecasting techniques are that the financial and economic variables are non-linear and the ANN can form a flexible linear method or non-linear relationship between the variables. They used ANN models to forecast the direction of Istanbul Stock Exchange National 100 index (ISE100). They divided the sample data which includes the highest prices, lowest prices, closing prices, exchange rate, and response rate into training and testing sets. They conducted Think Pro Networks for Windows software for their analysis. The result of their study suggests that ANN models forecast the direction of stock index with 74.51% accuracy rate.

However, one of the advanced and recent classification methods avoids the limitations above. Olatunji, Al-Ahmadi, Elshafei, and Fallatah (2013) used the Artificial Neural Network (ANN) for predicting the Saudi stock market. According to them, achieving good accuracy rate of prediction models will increase the confidence in trading in the Saudi Stock Market. From their point of view, the standard Neural Network has the ability to lean the existing relationships between the data in the sample. They used only closing prices as input variables to the model. They conducted simulation experiment to forecast the best possible price of the share. They tested the model on three different shares which have the major contribution in the Saudi Stock Markets. These companies include SABIC, Saudi Telecommunication Company (STC), and Al Rajhi Bank. Their experimental result indicates that Artificial Neural Network predicts the next day closing price with a very small RMSE with 1.81 and high correlation up to 99.9% in the testing sample. According to their research, ANN model can predict the market share prices with high rate of accuracy

Moreover, HÁJEK (2012) compared Prototype Generation Classifier with Support Vector Machine in predicting the direction of NASDAQ Composite index. According to him, more attention has to be paid to the index direction movement forecasting rather than index return forecasting. He claimed that behavior of stock prices movement is predictable using Prototype Generation Classifies. In the contrary of other studies, he divided the sample into 10 parts with the same size and trained 9 parts where the last part used for testing. According to him, Prototype Generation Classier was significantly more accurate compared to support Vector Machines where the hit ratio buy or sell could be detected with a high accuracy 92.05% using Modified Change's Algorithm MCA method.

Few academic researches focused on the decision trees models as alternative forecasting tools to the learning machines algorithm. Senyurt and Subasi (2012) used tree algorithm to forecast the movement direction of Istanbul Stock Exchange Index. Tree algorithms include CART, C4.5, and Random Forest. According to them, tree algorithm gives better estimation result for portfolio and fund managers than the conventional statistical tools. They used the technical indicators as input variables to the model. According to them, model parameters should be adjusted by different experiments or modifying the selected variables to improve the performace of the forecasting models. Their result indicates the technical indicators explain 78% of the index movement. However, there are vast studies on selecting the forecasting models parameters. The next subsection discusses the literature reviews on the predicting models parameters and their impact on the accuracy rate.

Selecting the Parameters of the Forecasting Models

There are exit vast studies which concentrate on the parameter and variables selection issues in the forecasting models of the stock markets. These parameters have been tested by the academic researcher such as Nalbantov, Bauer, and Sprinkhuizen-Kuyper (2005) when they examined the predictability of Support Vector Machines using macro-economic predictors. Their approach suggests constructing value and size premium in the U.S on S&P style indexes using set of macro-economic and technical predictors. These indicators include lagged value, Volatility, six months momentum, influence of industrial production, yield-curve cap, core inflation CPI, oil price, and corporate credit spread on the value premium. According to them, increasing number of variables at some level might fail the out-off-sample forecasting ability of the alternation models. Results of their study refer that Support Vector Machines highly predict the value and premiums over the trading period. This adds evidence on the predictability of SVM for growth in returns and values for the large stocks.

On the same side, Huanga, Nakamoria, and Wangb (2004) included the macroeconomic variables in their study. They investigate the capability of Support Vector Machines (SVM) to forecast the movement direction of stock index. They compare the performance of SVM with other classification methods such as Linear Discriminant Analysis, Quadratic Discriminant Analysis, Elman Back propagation, and Neural Network. They used weekly prices of NIKKEI225 index. The main objective of their research was to test the cross-sectional relationship between the stock prices and macroeconomic variables including interest rates, consumer price index, industrial production, government consumption, and gross domestic product (GDP). In addition to that, they have added the influence of USA economy on Japan economy as major indicator in the NIKKEI225 index. Around 7% of the sample was used as testing sample and the rest of sample as training sample. The main result of the study refers that SVM has the highest forecasting accuracy between the other forecasting methods were is recorded 75% as accuracy rate.
On the other side, Chena, Leungb, and Daoukc (2003) claimed that the macroeconomic variables are affecting the trading in the stock markets and indeed the forecasting results. They attempted to predict the direction of Taiwan Stock Exchange return. They applied

the probabilistic neural network (PNN) algorithm in order to predict the direction of index return. They applied the forecasting method after evaluating the training sample of historical prices. PNN algorithm compared with Generalized Methods of Moments (GMM) and Kalman filter. The motivation behind their study is to create profitable investment strategies on the index level. However, they did not take into consideration the type of investment strategies whether long term or short term investment. In addition to that, this study has to consider the influence of other economic variables and the interest rates which is reflecting the growing of stock markets. The data set for training proposes covers 5 years and the same can be said for testing and forecasting. According to them, PNN method has determined a stronger forecasting performance compared to GMM, Kalman filter, and Random Forest Model. Hence, the PNN is creating investment strategies that provide high return for the investor.

However, there should be a tool to select the valuable variables for predicting the stock index. Jerzy Krawczuk (2011) used CPL model as a linear classifier to predict the index movement direction. He used feature selection method RLS to select 52 attributes of the model. According to him, the main challenges of the predicting model is selecting the optimal parameters, features, and discovering the trading rules. Daily market data including open and close prices for 56 years period. The selected attributes include open price, gap change, daily change, 2 days change, 5 days change, 2 days back daily change, 9 days moving average, 12 days moving average, and 26 days moving average. Using the CPL model he concludes that best result of 53.91% accuracy rate could give advantage for investors to profit from forecasting the direction movement. In addition to that, Niaki and Hoseinzade (2013) increase the number of variables in the forecasting model. They applied Artificial Neural Network (ANN) to forecast the daily direction of Standard & Poor's 500 (S&P500) stock index. They have selected 27 factors that impacting the direction of the stock index. According to them, if the ANN model uses these factors as features and nodes, the daily direction of stock index will be predictable with more accuracy rather than traditional logit model. The data for their research includes 3650 daily closing prices. 80% of the sample was used as training sample, 10% for verification, and 10% for performance testing. The 27 factors are including financial and economic factors. Their results refer that exchange rates between US dollar and the main currieries such as British pound and Canadian dollar have significant impact on S&P500 direction and the ANN outperforms the logit model.

Nevertheless, Huang, Wang, Yu2, Bao1, and Wang (2006) proposed an efficient computational model to select the input variables of a forecasting method using Neural Networks. They compared the performance of neural networks based on the different selection of the input variables that affecting the direction of S&P500 and NIKKEI225 indexes. According to them, time series forecasting is explained by data intensity, series noise, non-stationary, unstructured nature, and high degree of uncertainty. They claimed that their method does not have any prior assumption regarding the time series in their study. The result of their research suggests that Neural Networks outperforms other time series forecasting techniques. Also, Kaur and Mangat (2012) propose Differential Evolution Support Vector Machine model (DE-SVM) to forecast the stock price. Differential Evolution used to select the best parameters for support Vector Machines to improve the results. The free parameters include the cost penalty, insensitive-loss function, and kernel parameter. They state that for maximizing the prediction performance of the model, under-fitting and over-fitting has to be depressed at the same time in the data mining. According to them, the ability of SVM in predicting the stock price is effecting by the choice of its parameters. The result of the research indicates that normalization of dataset improved the performance of the forecasting model and indeed the accuracy of the model output. Therefore, Support Vector Machines (SVM) performs better if the data sampled normalized.

Moreover, Emir, Dinçer, and Timor (2012) included the fundamental and technical analysis variables into the Support Vector Machines to classify the stock's performance. According to them, defining the most important parameters for the proper investment period is the main issue for the stocks classification. They compared the Artificial Neural Network (ANN) with Support Vector Machine on Istanbul Stock Exchange ISE30 index. They applied the two models on 12 technical indicators for 8 years period. They conducted 2 years only for training and the following year for testing. According to their experimental results, SVM showing the highest rate of classification between other models. They claimed that the performance results for both models might vary based on the technical and fundamental analysis. Moreover, LIN, GUO and HU (2013) predict the trend of stock market using Support Vector Machine as approach. According to them, there are two parts of the forecasting model feature selection such as the correlation and forecasting model. They used the correlation-based support vector machine (SVM) filter to rank the good and well performed financial indexes. Fore predicting the direction of the index, they used quasi-linear SVM on the historical prices of Taiwan Stock Market. They selected 53 technical indicators in their research as input features to the model to predict the stock price tendency. The experimental results refer that forecasting the accuracy of linear kernel outperform other non-linear kernel. This is lead to the result that SVM produces better performance that conventional methods according to their hit ratio.

Recently, few authors compared the financial indicators with the technical indicators used as attributes to the forecasting model. Han Chen (2007) claimed that financial indicators are more reliable, non-volatile and valid compared to the technical indicators. They proposed the Support Vector Machines (SVM) based on the financial indicates generated from the financial statement analysis to predict the stock prices. Their research data sample selected from the financial statement which released by the stock companies of Shanghai and Shenzhen. The financial indicators include Earning per Share (EPS), Book Value Per Share (BVPS), and Net Profit Growth Rate (NPGR) and classified the stock based on the Outstanding Achievement Growth Rate (OAGR) into +1 and -1. The forecasting experimental generated eight kinds of the models. Their study's result suggest that model developer can improve the accuracy of the forecasting using Support Vector Machines by removing or adding one of the financial indicators respectively.

Summary, Criticisms, and Comments

There has been a growing in the number of academic research studying the predictability of advanced learning machines algorithm in forecasting the direction of stock index. Most of these studies have made tremendous efforts to forecast the future direction of the stock markets or its returns and provide the traders with in the financial markets with financial trading strategies to turn the forecasting results into gain. Williams and Calvo (2002) study the automation of corporate announcement using Support Vector Machines.

Cao and Tay (2001) examine the performance of SVMs in financial forecasting and compare it with Back Propagation algorithm (BP). Cruz1, Afonso, and Giner (2003) compare the Support Vector Machines (SVMs) with Maximum Likelihood (ML) to forecast GARCH model parameters. Kim (2003) applies Support Vector Machines to forecast the direction of Korean stock market index. Harland (2002) applied Support Vector Machine on aluminum Futures contracts. Yuan (2011) argued that traditional forecasting models are unable to predict the index direction. Dutta, Bandopadhyay, and Sengupta (2012) confirmed that the Logistic Regression algorithm (LR) has more powerful to forecast the time series. Trafalis and Ince (2000) compared the Support Vector Machines algorithm with Backpropagation (BP) and Radial Basis Function (RBF). Gavrishchaka, Ganguli (2003) have identified the limitations of conventional volatility models for predicting foreign exchange and stock index volatility. Yang, Chan, and King (2002) state that applying Support Vector Regression model overlook the choices of margin setting. Wang and Choi (2013) state that predicting the direction of stock index may benefit short-term investors. Fan and Palaniswami (2001) claimed that fundamental accounting has to be used beside the price information in order to achieve good investment strategies. King, Vandrot, Weng (2010) show that there are no strong assumptions and over-fitting on the sample in the Support Vector Machines.

In addition to that, there are few studies which applied the Support Vector Machin to forecast the bankruptcy of the firms. The work of (Shih and Chen 2006) suggests that Support Vector Machine (SVM) is more accurate than the benchmark Back propagation (BP) model. Huanga,, Chena, Hsua, Chenb, and Soushan Wuc (2003) applied the Support Vector Machines on the corporate credit rating to help the bond investors and debt issuers in measuring the riskiness of the companies and bonds. Auria and Moro (2008) compared the Support Vector Machines with other traditional techniques such as Logistic Regression (LR) and Discriminant Analysis (DA). Shin, Lee1, and Kim (2005) show that Back-Propagation neural network (BP) outperforms other classification methods in recognizing patterns for the time series.

Recently the Support Vector Machines has been also compared to other forecasting models. Chen, Shih, and Wu (2006) compared the algorithm of Support Vector Machines (SVMs) with Back Propagation (BP) neural networks. Kumar and Thenmozhi (2006) compared Support Vector Machines and Random Forest. They argued that minimizing the prediction error does not lead to a capital gain. Shah (2007) compared the Support Vector Machines with Decision Stump and Linear Regression. Yu, Wang, and Lai (2005) compared the complexity of Support Vector Machines for variables selection with ARIMA and BP. Cao and Tay (2003) test the feasibility of SVM application on time series forecasting with Propagation (BP) neural network and the Regularized Radial Basis function (RBF). Abraham, Philip, and Saratchandran (2003) investigate the application of learning machines with Levenberg Marquardt, Support Vector Machine (SVM), Takagi-Sugeno neuro fuzzy, and Boosting Neural Network. Kara, Boyacioglu, and Baykan (2011) compared the performance of Artificial Neural Network (ANN) and Support Vector Machines (SVM). Yildiz, Yalama, and Coskun (2008) state that the reasons behind using the Artificial Neural Networks (ANN) as an alternative to the conventional forecasting techniques are that the financial and economic variables are non-linear and the ANN can form a flexible linear method or non-linear relationship between the variables. Olatunji, Al-Ahmadi, Elshafei, and Fallatah (2013) used the Artificial Neural Network (ANN) for predicting the Saudi stock market, HÁJEK (2012) compared Prototype Generation Classifier with Support Vector Machine. Senyurt and Subasi (2012) used tree algorithm to forecast the movement direction of Istanbul Stock Exchange Index.

Some research tend to focus on the variables selection issues in the forecasting models. Nalbantov, Bauer, and Sprinkhuizen-Kuyper (2005) examined the predictability of Support Vector Machines using macro-economic predictors. Huanga, Nakamoria, and Wangb (2004) included the macroeconomic variables in their study. Chena, Leungb, and Daoukc (2003) claimed that the macroeconomic variables are affecting the trading in the stock markets and indeed the forecasting results. Jerzy Krawczuk (2011) used CPL model and RLS to select 52 attributes of the model. Niaki and Hoseinzade (2013) increase the number of variables in the forecasting model. Huang, Wang, Yu2, Bao1, and

Wang (2006) proposed an efficient computational model to select the input variables of a forecasting method using Neural Networks. Kaur and Mangat (2012) propose Differential Evolution Support Vector Machine model (DE-SVM) to forecast the stock price. Differential Evolution used to select the best parameters for support Vector Machines to improve the results. Emir, Dinçer, and Timor (2012) included the fundamental and technical analysis variables into the Support Vector Machines to classify the stock's performance. LIN, GUO and HU (2013) predict the trend of stock market using Support Vector Machine as approach. According to them, there are two parts of the forecasting model feature selection such as the correlation and forecasting model. Han Chen (2007) claimed that financial indicators are more reliable, non-volatile and valid compared to the technical indicators.

However, critics have been raised about these advanced techniques and their ability to meet the investors and portfolio managers requirements. Firstly, the difficulties in designing the model structure and function in specific mathematical conditions. Secondly, the sample selection issues which come from the complexity and intensity of the automation efforts involved. Thirdly, out-of-sample testing might be used on small data sets that are improbable reprehensive the full market behavior. In addition to that, the evaluation of the forecasting models has in some cases restricted only to the hit ratio of the accuracy ignoring other performance measurements related to the trading strategies. Moreover, the previous studies focused primary on the microeconomic variables as inputs to the forecasting model. These variables might not have direct impact on the daily prices of the stock indexes. So, it is better to include more variables which impact on the stock direction such as the technical variables. Also, the previous literature focused mainly on forecasting the daily, monthly, or yearly stock returns. However, more attention has to be given to the stock direction to build short-term or long-term investment strategies

To sum up with, the performance measurements in the previous studies might not represent the potential losses that come from the trading strategies provided from the research to the investors and market participants. The contribution of this research is doubled. First, it is the first study which applies the Support Vector Machine (SVM) algorithm on the GCC stock indexes and compares it with the performance of other forecasting models. Second, unlikely to the previous studies, this research includes most of the technical indicators used in the financial markets as input variables to forecast the direction of the daily GCC stock indexes. Generally speaking, none of the trading systems in the GCC brokerage companies apply this algorithm in their trading systems. Therefore, if this algorithm used in the online trading system as adviser to the investor, it will attract and bring more investors and portfolios to the brokerage companies for trading and indeed bring more profits to the brokerage companies.

Methodology and Data Collection

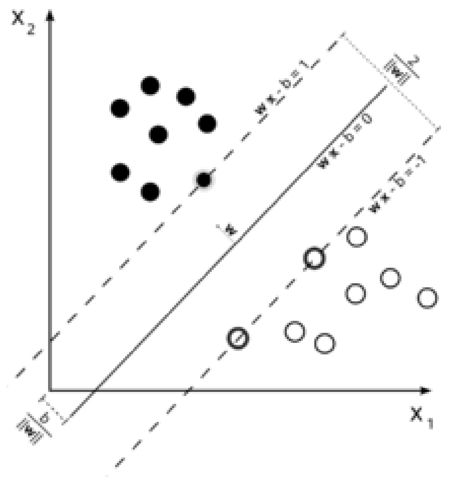
The objective of this study is to introduce the Support Vector Machines algorithm as an advanced tool to the trading systems in the GCC financial markets. The previous studies mentioned above have indicated that Support Vector Machine (SVM) can be effective in the financial sector and can be advanced techniques for decision makers. This section presents the basic concept of Support Vector Machine as described by previous studies.

Theory of Support Vector Machine Algorithm

Support Vector Machine (SVM) is an advanced technique for classification, regression, and forecasting purposes. SVMs are a collection of learning algorithms that analyze data and recognize patterns in the time series. The current standard SVM was introduced by Cortes and Vapnik (1995). The main advantages of SVM is that it is simple for user's understanding and implementing, better accuracy rate compared to other methods such as Neural Networks, Decision Tree, and Logistic regression. Moreover, it avoids most of the issues in traditional forecasting models which discussed in the previous sections. Support Vector Machine is able to overcome most of the limitations encountered when applying other learning machines as well.

Support Vector Machine (SVM) uses linear model to apply nonlinear classification through mapping the input vectors x into high-dimensional feature space. The linear model created in the new space can symbolize a non-linear decision boundary in the main space. In the new constructed pace, an optimal classifier hyper-plane is created. So, Support Vector Machine defined as algorithms that find the optimal linear model which maximize the distance between decision classes which called the margin hype-plane. The training samples which are closest to the optimal hyper-plane called support vectors and other training samples are not important for defining the binary class. Figure (1) explains the concept of Support Vector Machine

Figure (1)



The hyper-plane that separating the two decision classes in two attributes case can be represented by the following equation:

$y = w_0 + w_1x_1 + w_2x_2$ (1)

Where y is the output of the model, x_1, x_2 … x_i are the inputs values, and there are three parameters w_0, w_1, w_2 … w_i to be learned by the algorithm. These parameters determine the hyperplane.

Therefore, the maximum margin between the hyperplanes can be represented by the following:

$\boldsymbol{Y} = b + \sum a_i \, y_i \, \boldsymbol{x}(\mathrm{i}).\ \mathbf{x}$ (2)

Where $\boldsymbol{Y}$ is the class value of training sample $\boldsymbol{x}$(i), **x** refer to the test sample , and $\boldsymbol{x}$(i) refer to the support vectors in the sample. b and a_i are the parameters which construct the hyperplane.

Support Vector Machine Mathematical Model

This section presents the mathematical model of the Support Vector Machine (SVM). Let’s refer to the distance between the hyperplane and the support vectors in the testing sample by *D*. The smallest distance between the support vectors and the origin hyperplane which separate the points into either side called margin. The objective is to find the optimal hyperplane which separate the points perfectly. This objective will be achieved if the margin is maximized. Let’s refer to the margin by *q*.

The objective of this study is to forecast the direction of daily indexes in the GCC stock markets. To achieve this goal, the problem of separating the data sample into two classes {UP, DOWN} need be solved.

Consider the binary set $G = \{(x_i, y_i), i = 1, 2\ldots n\}$

The hyperplane is given by

$w^T \varphi(x) + b = 0$

where $(x_i \epsilon R^n$ is the input variables and y_i is the class target $y_i \epsilon \{-1,1\}$ where -1 represent the DOWN direction , and 1 the UP direction.

Taking into consideration the following conditions:

$w^T \varphi(x) + b \geq 1$ if $y_i=1$ (3)

$w^T \varphi(x) + b \leq -1$ if $y_i= -1$ (4)

The above two equations are equal to $y_i [w^T \varphi(x) + b] \geq 1 \quad i = 1, 2\ldots n$ (5)

This is in case of two dimensions space. However, there should be feature to map the input space to a high dimensional space where data sample become linearly separable. So, mapping should be as $\varphi : R^n \rightarrow R^m$

The distance of the sample x_i from the hyperplane can be given by

$$d(x_i, w, b) = \frac{|w^T \varphi(x_i) + b|}{\|w\|^2} \quad (6)$$

The margin is calculated as mentioned in figure (1) by:

$q = 2/\|w\|$ (7)

In order to find the hyperplane which optimally separate the sample, we have to solve the optimization issue as follows:

$\min \varphi (w) = 1/2\|w\|^2$ (8)

The solution for above issue is given by Lagrange function as follows:

$$L_{P1} = \frac{1}{2} \|w\|^2 - \sum_{i=1}^{N} \alpha_i [y_i(w^T \varphi(x_i) + b) - 1] \quad (9)$$

Hence, the equation to calculate the weight for each attribute given by

$w = \sum \alpha_i y_i \varphi (x_i)$ (10)

From the above, b factor is calculated for one point as $b = y_i - w^T \varphi(x)$ and for the entire sample as follows:

$$b = \frac{1}{N_s} \sum_{0<\alpha_j<C} [(y_j - w^T \varphi(x_j)] \tag{11}$$

Where *Ns* is the number of support vectors in the sample. Therefore, for any new data set x, the classification function is given by:

$$f(x) = \text{Sign}\left(\sum_{i=1}^{N} \alpha_i y_i \varphi(x_i)^T \varphi(x) + \frac{1}{N_s} \sum_{0<\alpha_j<C} \left(y_j - \sum_{i=1}^{N} \alpha_i y_i \varphi(x_i)^T \varphi(x_j) \right) \right) \tag{12}$$

This study follows the research of Cao and Tay (2001) by using the polynomial kernel function in order to increase the performance of SMV in the training process. The kernel function considers the dot-product data sample in a high dimensional feature space. The kernel function is given by: k $(x_i , x_{j\ )=}$ φ $(x_i)^T$ φ (x_j), the rational for this function is to avoid overfitting and underfitting issues while selecting the model parameters. So, the non-linear classification model can be written by:

$$f(x) = \text{Sign}\left(\sum_{i=1}^{N} \alpha_i y_i K(x_i, x) + \frac{1}{N_s} \sum_{0<\alpha_j<C} \left(y_j - \sum_{i=1}^{N} \alpha_i y_i K(x_i, x_j) \right) \right) \tag{13}$$

This research compares also the performance of Support Vector Machine with other classification models such as Logistic Regression (Logit Model) and Random Forest (Decision Tree Model). The following section discuss briefly these two models

Random Forest Model

It is a learning machine algorithm used for classification and forecasting. The idea of random forest is to create multitude of decision trees for the training sample and provide the output which make the classes results popular by individual trees. This model was developed by Breiman (2001). According to him, each decision tree is created using the following algorithm:

1. let's refer to the number of training sets as *N* and number of attributes as *M*
2. let's consider the number of input variables in this study the technical indicators as *m* which are used to determine the decision at a specific node of the decision tree *m* < *M*

3. the next step is to select the training set for this tree by selecting *n* times training cases alternatively. The remaining sets will be used for predicting the accuracy of the model
4. randomly chose *m* for each node out of the *M* variables to select the best split
5. each individual tree will grow vertically to reach to the final classifier

The prediction model for the whole forest can be given using the following formula:

$$F(X) = \frac{1}{M}\sum_{m=1}^{M} T_m(X) = \frac{1}{M}\sum_{m=1}^{M}\sum_{i=1}^{n} W_{im}(X)Y_i = \sum_{i=1}^{n}\left(\frac{1}{M}\sum_{m=1}^{M} W_{im}(X)\right)Y_i \tag{14}$$

Where *X* is the set point in the sample, *Y* is the classifier (UP, DOWN), and *W* is the weights

Logistic Regression Model

It is the logit regression used as classification and forecasting model. It is measuring the relationship between the independent and dependent variables in a data set by predicting the weights for each independent variable. Logistic regression can be binominal or multinomial. While the binominal regression deals with two possible outcomes from the dependent variables, multinomial has three or more possible types of the dependent variables outcomes. This study used multinomial regression as comparison model with Support Vector Machine (SVM) to forecast the direction of the stock indexes.

Logit model can be written using the formula below:

$$\operatorname{logit}(\mathbb{E}[Y_i \mid \mathbf{X}_i]) = \operatorname{logit}(p_i) = \ln\left(\frac{p_i}{1-p_i}\right) = \boldsymbol{\beta}\cdot\mathbf{X}_i$$

(15)

Where *β* the regression coefficient and X_i is the variables added to the model. The next section presents the sampling and data collection for the selected seven GCC indexes

Sampling and Data Set

After the financial crises of 2007 and 2008 around the world which impacted GCC financial markets as well, recovery started on both real estate and financial sectors from 2010. The interest of both local and foreign investors to trade in the GCC stock markets shown a high increased during the period of 2010 and 2013. The research data used in this study contains the daily closing prices of the seven GCC stock indexes which include Dubai Financial Market (DFM), Abu Dhabi stock Exchange (ADX), Bahrain Stock

Market (BHM), Doha Stock Market (DSM), Kuwait Stock Exchange (KSE), Muscat Stock Market (MSM), and Saudi Stock Market (TADAWUL). Data sample for GCC markets covers four years period span from Jan-2010 to Jan-2014. The data downloaded from two databases, Money Experts Club (MEC) and the website of Saudi stock market. As mentioned above, data set does not contain the unusual period of 2007 to 2010 which is the main feature of the data sampling. Daily price report selected for each market contains date, open price, high price, low price, and closing price. The research periods for the seven GCC stock markets are the same. However, number of instances might vary from market to another due to the holiday's policy of each country. The whole data sample covers the period from the 31st of December 2009 to the 31st of December 2013. The total number of instances for each market is described in Table 1.

Table1

Financial Years	Market	Direction UP	Direction Down	No Direction	%UP	% Down	%False	Total
2010-2013	ADX	538	465	9	53%	46%	0.9%	1012
2010-2013	DFM	514	486	3	51%	48%	0.3%	1003
2010-2013	BHM	505	469	14	51%	47%	1.4%	988
2010-2013	DSM	674	849	8	44%	55%	0.5%	1531
2010-2013	KSE	459	531	6	46%	53%	0.6%	996
2010-2013	MSM	557	717	11	44%	57%	0.9%	1258
2010-2013	TADAWUL	592	403	1	59%	40%	0.1%	996

Two types of sampling methods were conducted to evaluate the prediction models. The first one is 10-fold cross-validation where the model takes the full data set and creates 10 equal sized samples. Each set is divided into two samples. Let's refer to the total sample as K. So, K-1 group is used for training the sample to produce the classifier model and set1 for testing and evaluating. The same can be repeated for the entire sample to produce n sets. The average of testing can be taken to produce the final model.

The second type of sampling is to split the entire sample into two set. The first set which present 75% of the total sample used to create the model. The remaining 25% of the sample used to determine the model parameters values, testing, and evaluating the three models. Using this kind of sampling methods, the parameters values becomes more capable to present the whole data sample. The data set contains also ten technical indicators as input variables to the forecasting model. The following section explains the ten technical indicators used to create the initial attributes of the model.

Technical Indicators

In the light of the previous literature, it is claimed that technical indicators can be used as signals for future market direction (Kim, 2003). Therefore, many types of technical indicators can be used as input variables to build an effective prediction models to forecast the direction of stock markets. This study used 10 technical indicators as input variables to the forecasting model. These indicators include the following:

1. Stochastic % K: is a momentum indicator illustrating the position of the closing price relative to high and low range over a specific period. It is given by the following formula: Stochastic %K = (Current Close - Lowest Low)/(Highest High - Lowest Low) * 100
2. Stochastic % D: is three days moving average of %K.
3. Stochastic Slow %D: it is simple moving average of %D
4. Rate of Change: it is the difference between the current price and price of n days back: ROC = [(Close - Close n periods ago) / (Close n periods ago)] * 100
5. Williams % R: is technical indicators that measures the overbought and oversold level over a given period. The default parameters for Williams %R is 14 periods which can be days, weeks, moths, or years. Williams indicator range from 0 to -100. All records from 0 to -20 are considered as overbought records. In the contrast, records from -80 to -100 are considered as oversold. It is given by: %R = (Highest High - Close)/(Highest High - Lowest Low) * -100
6. Disparity5: is the distance of current closing price from 5-days moving average. It is given by : (Close)/(5-dyas MAV)*100
7. Disparity10: is the distance of current closing price from 10-days moving average. It is given by: (Close)/(10-dyas MAV)*100
8. Price Oscillator (OSCP): is the difference between the 5-days and 10-days moving average. It is given by: (MA5-MA10)/MA5*100

9. Commodity Channel Index (CCI): is the technical indicator which measures the difference between the security's price and its statistical mean. It is given by:

   CCI = (Typical Price - 20-period SMA of TP) / (.015 x Mean Deviation)

   Where

   Typical Price (TP) = (High + Low + Close)/3

   And

   Constant = .015

10. Relative Strength Index (RSI): is defined as a price following an oscillator which fluctuates between 0 and 100. It is calculated using the following formula:

    RSI = 100 – (100/(1 + RS)

    Where:

    RS = Average Gain / Average Loss

    The default parameter of RSI is 14 periods

Table 2 shows the summary statistics of each indicator for one market which is DFM. Reset of other markets attributes are presented under Appendix 4

Table 2

Indicator	Minimum	Maximum	Mean	StdDev
StochasticK	0	100	53.825	32.288
StochasticD	2.743	100	53.777	30.504
SlowD	3.144	99.081	53.759	30.168
ROC	83.688	116.847	100.76	4.72
WilliamsR	-100	0	-46.175	32.288
Disparity5	94.177	105.746	100.126	1.42
Disparity10	91.698	107.441	100.287	2.289
OSCP	-0.044	0.034	-0.002	0.013
20-day CCI	-323.666	361.205	4.879	112.987
14-day RSI	1.976	96.592	52.982	19.531

To sum up with, daily prices for the general indexes in all GCC stock markets were used to calculate the 10 technical indicators as input variables to the model. The direction has been determined by comparing the current price with yesterday's price. Direction is "UP" if the current price is greater than yesterday's price. In contrast, daily direction is "DOWN" if the current price is less than yesterday's price. However, if the price did not change from day to day, the formula is returning “FALSE” value. Two types of experiments were conducted to produce the model and evaluate the performance. The next section presets the experimental study and analyses the results based on the research objectives and hypothesis.

Experimental Study, Discussion, and Results

This section presents the experimental study on GCC stock markets data using Weka software as analytical tool. Weka is set of machine learning algorithms for classification regression, visualization, and huge data processing. The biggest advantage of the Weka software is the capability of searching for the optimal model. Furthermore, it can use different algorithm and produce the output of the prediction models. As mentioned earlier, this study apply Support Vector Machine algorithm to forecast the daily direction of GCC stock indexes. The next section is explaining the process of applying Support Vector Machine (SVM) using Weka software.

Application of Support Vector Machine (SVM)

Two types of testing have been conducted on the GCC stock markets data. The file type which has the final calculation of ten technical indicators along with the classifier (UP, DOWN) is CSV file. The file created for each market separately. This file is uploaded to the Weka software to run the model. Total number of attributes including the classifier is 11. The first test mode is 10-fold cross-validation using Support Vector Machine (SVM). The kernel used in this testing is linear kernel where K(x,y) = <x,y>. The testing objective is to classify the direction of next day index price into two classes DOWN, UP.

In order to produce the prediction and classification model, this study follows the traditional cross validation method by dividing the total data sample into two independent sets: training set and validating set. The training set is used for producing the optimal model, whereas validating set for evaluating the model.

Table 3 shows the attributes weights for Support Vector Machine forecasting model using 10-fold cross-validation testing method as experiment 1 which producing the prediction model. The sign "+" refer to the positive relation of each indicator and the sign "-"refer to the negative relation of each indicator.

Table 3

MARKET	ADX	BHM	DFM	DSM	KSE	MSM	TADAWUL
StochasticK	(+)5.6444	(+)-5.1875	(+)4.75	(+)5.8765	(+)-4.6135	(+)-5.4018	(+)5.2576
StochasticD	(+)-4.8096	(+)2.3273	(+)-5.4106	(+)-6.1534	(+)5.2953	(+)5.8021	(+)-2.6748
SlowD	(+)-5.923	(+)3.5658	(+)-3.8161	(+)-5.2764	(+)4.0122	(+)4.76	(+)-4.5851
ROC	(+)0.9175	(+)0.4173	(+)0.6433	(+)0.9225	(+)-0.2734	(+)0.0486	(+)0.3666
WilliamsR	(+)5.6444	(+)-5.1875	(+)4.75	(+)5.8765	(+)-4.6135	(+)-5.4018	(+)5.2576
Disparity5	(+)0.4441	(+)-6.1262	(+)2.4948	(+)0.1802	(+)-3.2042	(+)-3.2633	(+)5.6921
Disparity10	(+)0.3915	(+)-2.5035	(+)1.0968	(+)0.4583	(+)-0.9058	(+)-1.3975	(+)2.6435
OSCP	(+)0.4348	(+)-1.9568	(+)1.5338	(+)0.022	(+)-2.0076	(+)-2.622	(+)2.2831
20-day CCI	(+)2.4413	(+)0.9107	(+)-0.673	(+)2.0584	(+)-0.7672	(+)-2.0495	(+)1.549
14-day RSI	(+)1.1645	(+)-1.6003	(+)1.9977	(+)1.7482	(+)-1.6893	(+)-1.029	(+)1.4533
b	(-)3.3378	(+)8.0091	(-)4.0442	(-)2.9221	(+)4.6624	(+)6.3258	(-)11.3991

In the next experiment, data sample split into two parts. The first part which is 75% for the data sample was used for sample learning and constructing the forecasting model. Rest of the sample 25% was used for testing proposes. Time taken to build the model using Weka software is 0.05 seconds. Attributes weight in both experiments are the same as Table 3. However, the performance of the Support Vector Machine is varying between the two experiments. The next section shows the evaluation of Support Vector Machine for the GCC stock markets.

Performance of Support Vector Machine (SVM)

Following to previous literature, this study used confusion matrix, accuracy hit ratio, overall accuracy hit ratio of the model, and the average hit ratio for the two types of experiments to evaluate the performance of Support Vector Machine (SVM). To evaluate the prediction ability of Support Vector Machine (SVM), comparison of accuracy hit ratios between the GCC stock markets is conducted. To avoid the data over-fitting issue, two experiments used for this comparison. The first one is 10-fold cross-validation and the second one is 75% sample split. Table 4 shows the overall accuracy hit ratios of the classification model in GCC stock markets under study.

Table 4

Market		ADX	BHM	DFM	DSM	KSE	MSM	TADAWUL
Evaluation on test split	Correctly Classified Instances	81.03%	72.06%	82.07%	81.20%	79.92%	79.75%	75.50%
	Incorrectly Classified Instances	18.97%	27.94%	17.93%	18.80%	20.08%	20.25%	24.50%
10-fold cross-validation	Correctly Classified Instances	82.11%	76.72%	81.16%	81.25%	80.62%	*83.42%	77.51%
	Incorrectly Classified Instances	17.89%	23.28%	18.84%	18.75%	19.38%	16.58%	22.49%
	Average Accuracy	***82%**	**74%**	***82%**	**81%**	**80%**	***82%**	**77%**

As can be seen from Table 4, Support Vector Machine is outperforming in all markets in both experiments where the average accuracy classification rate is between 72.06% and to 83.42% in all markets. However, the highest average accuracy rate 82% using SVM were recorded in ADX, DFM, and MSM respectively. Moreover, SVM recorded highest accuracy rate using 10-fold cross-validation in MSM (Muscat Stock Market) where the accuracy rate us 83.42%. While the lowest accuracy rate was recorded using test split evaluation method when the model classified the index direction 72.06% of the total testing sample correctly. Clearly, the accuracy hit ratio of SVM using test-split method is lower than accuracy rate using 10-fold cross-validation method. Therefore, H2 is rejected and H0 is accepted. So, the sample testing selection has significant impact on the accuracy hit ratio using Support Vector Machines. This is lead to say that SVM is outperforming when 10-fold cross-validation is used for evaluating the model.

The performance of Support Vector Machines was tested also using the confusion matrix. To evaluate the performance of Support Vector Machine between the GCC stock markets, this study analyze the confusion matrix for each experiment. Table 5 shows the confusion matrix for split 75.0% train experiment, while Table 5 & 6 show the confusion matrix for 10-fold cross-validation of SVM for 7 GCC stock indexes.

Table 5

Test mode: split 75.0% train

Table 6

Test mode: 10-fold cross-validation

Market	Down	Up	FALSE	Total	Market	Down	Up	FALSE	Total
	93	25	0	118		369	96	0	465
ADX	20	112	0	132	ADX	76	462	0	538
	3	0	0	3		4	5	0	9
Total	**116**	**137**	**0**	**253**	**Total**	**449**	**563**	**0**	**1012**
	83	31	0	114		345	124	0	469
BHM	33	95	0	128	BHM	92	413	0	505
	2	3	0	5		6	8	0	14
Total	**118**	**129**	**0**	**247**	**Total**	**443**	**545**	**0**	**988**
	102	22	0	124		396	90	0	486
DFM	22	104	0	126	DFM	96	418	0	514
	0	1	0	1		1	2	0	3
Total	**124**	**127**	**0**	**251**	**Total**	**493**	**510**	**0**	**1003**
	123	49	0	172		488	186	0	674
DSM	21	188	0	209	DSM	93	756	0	849
	1	1	0	2		2	6	0	8
Total	**145**	**238**	**0**	**383**	**Total**	**583**	**948**	**0**	**1531**
	110	22	0	132		452	79	0	531
KSE	28	89	0	117	KSE	108	351	0	459
	0	0	0	0		1	5	0	6
Total	**138**	**111**	**0**	**249**	**Total**	**561**	**435**	**0**	**996**
	155	28	0	183		633	84	0	717
MSM	34	101	0	135	MSM	118	439	0	557
	2	1	0	3		8	3	0	11
Total	**191**	**130**	**0**	**321**	**Total**	**759**	**526**	**0**	**1285**
	50	52	0	102		231	172	0	403
TADAWUL	9	138	0	147	TADAWUL	51	541	0	592
	0	0	0	0		1	0	0	1
Total	**59**	**190**	**0**	**249**	**Total**	**283**	**713**	**0**	**996**

The confusion matrix is an error matrix that is evaluating the performance of the algorithm. The column is representing the instances in the forecasted class, while, the row is representing the actual output of the class. The two tables above are summarizing the results of testing SVM algorithm. As can be seen from table 5, the prediction model classified 93 instances as DOWN direction correctly out of 118 which is 78% of the total DOWN direction instances in the testing sample for ADX using the 75% Split mode. On the other side, using 10-flod cross validation it is classifying 369 instance as “DOWN” out of 465 correctly which is 79% of the total number of the DOWN instances. Only 25 out of 118 and 96 out of 465 incorrectly classified using both experiments respectively. The same analysis can be said for “UP” direction in ADX. To evaluate the classification model capability better, numbers in Table 5 and Table 6 can be converted to percentage for comparison and analyze the difference between the two experiment.

Test mode: split 75.0% train **Test mode: 10-fold cross-validation**

Market	Down	Up	FALSE	Total	Market	Down	Up	FALSE	Total
	79%	21%	0%	118		79%	21%	0%	465
ADX	15%	*85%	0%	132	ADX	14%	*86%	0%	538
	100%	0%	0%	3		44%	56%	0%	9
Total	**116**	**137**	**0**	**253**	**Total**	**449**	**563**	**0**	**1012**
	73%	27%	0%	114		74%	26%	0%	469
BHM	26%	*74%	0%	128	BHM	18%	*82%	0%	505
	40%	60%	0%	5		43%	57%	0%	14
Total	**118**	**129**	**0**	**247**	**Total**	**443**	**545**	**0**	**988**
	82%	18%	0%	124		81%	19%	0%	486
DFM	17%	*83%	0%	126	DFM	19%	*81%	0%	514
	0%	100%	0%	1		33%	67%	0%	3
Total	**124**	**127**	**0**	**251**	**Total**	**493**	**510**	**0**	**1003**
	72%	28%	0%	172		72%	28%	0%	674
DSM	10%	*90%	0%	209	DSM	11%	*89%	0%	849
	50%	50%	0%	2		25%	75%	0%	8
Total	**145**	**238**	**0**	**383**	**Total**	**583**	**948**	**0**	**1531**
	*83%	17%	0%	132		*85%	15%	0%	531
KSE	24%	76%	0%	117	KSE	24%	76%	0%	459
	NA	NA	NA	0		17%	83%	0%	6
Total	**138**	**111**	**0**	**249**	**Total**	**561**	**435**	**0**	**996**
	*85%	15%	0%	183		*88%	12%	0%	717
MSM	25%	75%	0%	135	MSM	21%	79%	0%	557
	67%	33%	0%	3		73%	27%	0%	11
Total	**191**	**130**	**0**	**321**	**Total**	**759**	**526**	**0**	**1285**
	49%	51%	0%	102		57%	43%	0%	403
TADAWUL	6%	*94%	0%	147	TADAWUL	9%	*91%	0%	592
	NA	NA	NA	0		100%	0%	0%	1
Total	**59**	**190**	**0**	**249**	**Total**	**283**	**713**	**0**	**996**

The Support Vector Machine algorithm perform well in classifying and predicting the "UP" direction for ADX, BHM, DFM, DSM, and TADAWUL using split 75% train mode where the misclassified sample recorded 15%, 26%, 17%, 10%,and 6% of the total "UP" direction instances respectively. On the same side, SVM perform well in predicting the "UP" direction using 10-fold cross-validation in ADX, BHM, DFM, DSM, and TADAWUL where misclassified sample recorded 14%, 18%, 19%, 11%, and 9% . In the contrast, Support Vector Machine (SVM) algorithm outperform in classifying and predicting the "DOWN" directing in KSE and MSM where 17% and 15% of the total DOWN instances are misclassified. Table 5 & 6 shows also that SVM algorithm did not perform well in classifying the DOWN directing for TADAWUL index where only 49% of the total DOWN instances was classified correctly and 51% of the same are misclassified using the 75% split mode train. The same can be said for 10-fold cross-validation experiment where only 57% of the total DOWN instances is classified correctly using the algorithm and 43% are misclassified. This may be because there are some input variables which need to be included into the analysis for Saudi stock market (TADAWUL). This model finds difficulties in selecting these variables because of the poor outlook of Saudi domestic economy during the last 3 years financial period. The performance of Support Vector Machine is compared with other classification models such as Random Forest and Logistic Algorithm. The next section makes a comparison between the three classification models for the selected markets under study.

Support Vector Machines, Random Forest, and Logistic Algorithm

This study follows the previous studies of Cao and Tay (2001) in evaluating and comparing the performance of support Vector Machine with other classification algorithms. The prediction models is evaluated for each market using the overall Accuracy hit ratio, Mean Absolute Error, and Root Mean Squared Error. These statistics are measuring the correctness of the prediction by comparing it with the actual predicted values in the testing sample. The smaller the error values, the better the performance and the predicted values are closer to the actual values. Table 7 compares and analyzes the performance of Support Vector Machine with other classifications algorithm using split 75.0% train, remainder test experiment.

Table 7: Test mode: split 75.0% train, remainder test

	Market	Accuracy hit Ratio	Mean Absolute Error	Root Mean Squared Error
SVM	ADX	81.03%	0.267	0.3448
	BHM	72.06%	0.2888	0.375
	DFM	*82.07%	*0.2629	*0.3388
	DSM	81.20%	0.2652	0.3421
	KSE	79.92%	0.2668	0.3445
	MSM	79.75%	0.2693	0.3481
	TADAWUL	75.50%	0.2767	0.3585
	MAX	**82.07%**	**28.88%**	**37.50%**
	MIN	**72.06%**	**26.29%**	**33.88%**
Logit	ADX	82.61%	0.1597	0.2882
	BHM	75.71%	0.2008	0.3253
	DFM	*85.66%	*0.1389	*0.2724
	DSM	82.77%	0.1527	0.2847
	KSE	85.54%	0.1358	0.2497
	MSM	85.36%	0.1352	0.2623
	TADAWUL	80.32%	0.1704	0.2991
	MAX	**85.66%**	**20.08%**	**32.53%**
	MIN	**75.71%**	**13.52%**	**24.97%**
Random Forest	ADX	77.08%	0.1755	0.3174
	BHM	74.09%	0.2148	0.3435
	DFM	80.48%	0.175	0.3022
	DSM	*81.98%	*0.1629	*0.296
	KSE	78.71%	0.1775	0.3106
	MSM	81.62%	0.1607	0.3046
	TADAWUL	79.92%	0.1692	0.311
	MAX	**81.98%**	**21.48%**	**34.35%**
	MIN	**74.09%**	**16.07%**	**29.60%**

Table 7 shows that the maximum accuracy rate using SVM in all markets is 82.07% in DFM. It is followed by 81.20% in DSM and 81.03% in ADX. SVM algorithm outperforms better in UAE stock markets and Doha Stock Market (DSM). Both MAE and RMSE are confirming the same where the Mean Absolut Error is 0.2629, 0.2652, and 0.267 and Root Mean Square Error 0.3388, 0.3421, and 0.3448 in DFM, DSM, and ADX respectively. In contrast, the accuracy hit ratio decreases when SVM is applied on

BHM stock data were the accuracy rate is 72.06% and the Mean absolute error is 0.2888%. Table 7 shows also that the Logistic Regression Algorithm (Logit) outperform in all markets where the accuracy rates crossed 80% in most of the markets. However, the better performance for this model applied on DFM where the accuracy rate is 85.66%. It is followed by 85.54% in KSE, 85.36% in MSM, 82.77% in DSM, and 82.61% in ADX. However, the accuracy rate shows a slight decrease in BHM where it is recorded 75.71%. Random Forest decision tree recorded the lowest accuracy hit ratios between the three models. Random Forest performs well only in DSM, MSM, and DFM where the prediction accuracy rates are 81.98%, 81.62%, and 80.48% respectively. The lowest accuracy rate 74.09% and the maximum the Mean absolute error 0.21 values were in BHM.

To conclude with the comparison, SVM and Logistic Regression are performing well in ADX, while the accuracy rate for Random forest is less. The performance of all models is little bit less when it is applied on BHM market. The logistic Regression has better accuracy rate in BHM compared to SVM and Random Forest. All models are outperforming when it is applied on DFM market data. The accuracy rate for SVM is 82.07%, while it is 85.66% using Logistic Regression and 80.48% using Random Forest. This is lead to say that Logistic Regression outperform other forecasting models in Dubai Financial Market (DFM). In addition to that, all models are outperforming when it is applied on Doha Stock Market (DSM). The accuracy rate for SVM, Logit, and Random Forest Decision tree are 81.20%, 82.77%, 81.98% respectively. However, the best performance was for Logistic Regression Algorithm. Logit model and Random Forest classification models have better performance than SVM in DSM stock market.

The same analyses have been done using the experiment of 10-fold cross-validation. Table 8 shows the comparison results of the three models. It is showing the accuracy hit ratios for classifying the direction of each market index into “UP” and “DWON” and it is showing also the Mean Absolute Error and Root Mean Square Error for the three models.

Table 8

	Market	Accuracy hit Ratio	Mean Absolute Error	Root Mean Squared Error
SVM	ADX	82.11%	0.2639	0.3403
	BHM	76.72%	0.2771	0.3591
	DFM	81.16%	0.2648	0.3415
	DSM	81.25%	0.265	0.3419
	KSE	80.62%	0.2666	0.3442
	MSM	83.42%	0.261	0.3359
	TADAWUL	82.13%	0.1664	0.2872
	MAX	**83.42%**	**27.71%**	**35.91%**
	MIN	**76.72%**	**16.64%**	**28.72%**
Logit	ADX	84.19%	0.1516	0.2769
	BHM	80.67%	0.183	0.3032
	DFM	85.24%	0.1454	0.2709
	DSM	83.15%	0.1581	0.282
	KSE	85.24%	0.1361	0.2619
	MSM	87.47%	0.1325	0.2552
	TADAWUL	82.13%	0.1664	0.2872
	MAX	**87.47%**	**18.30%**	**30.32%**
	MIN	**80.67%**	**13.25%**	**25.52%**
Random Forest	ADX	81.32%	0.1663	0.3041
	BHM	74.60%	0.2075	0.3415
	DFM	78.66%	0.1749	0.3125
	DSM	80.34%	0.1675	0.3035
	KSE	78.41%	0.1803	0.3189
	MSM	82.96%	0.1523	0.286
	TADAWUL	81.22%	0.1607	0.2975
	MAX	**82.96%**	**20.75%**	**34.15%**
	MIN	**74.60%**	**15.23%**	**28.60%**

After examining the results in table 8, it is found that the performance of Support Vector Machines has been increased in all markets except DFM where the accuracy hit ratio decreased form 82.07% to 81.16% and slight increase of both error measures. The performance of Logit model shows a slight decrease in DFM and KSE. Moreover, the performance of Random Forest decreased in three market indexes DFM, DSM, and KSE. A part from that, all models performance has been increased when 10-fold cross-validation testing methods is applied. In BHM market, the performance of both Support Vector Machine (SVM) and Logit model has been increased significantly by around 5%. The performance of Random Forest showed a significant increase in ADX market when the accuracy hit ratio increased by 4.25% after changing the sampling method to 10-fold cross-validation.

Overall, Support Vector Machine outperform Random Forest algorithm in classifying and predicting the direction of GCC stock indexes using 10-fold cross-validation testing method. Support Vector Machine also outperforms Random Forest algorithm when split 75.0% train on ADX, DFM, and KSE time series. In contrast, Logistic Regression outperforms Support Vector Machine and Random Forest in all markets for both experimental studies. This is lead to say that, H1 in this study is accepted and H0 is rejected when Support Vector Machine is compared with Logistic Regression algorithm in all markets. So, Support Vector Machine does not perform comparing to Logistic regression. However, H1 is rejected for ADX, DFM, and KSE and accepted for BHM, DSM, MSM, and Tadawul when SVM is compared to Radom Forest using split 75.0% train testing method. On the other side, H1 is rejected in all markets when SVM is compared to Radom Forest using 10-fold cross-validation testing method. So, the Support Vector Machines outperforms comparing to Random Forest in all Markets using 10-fold cross-validation testing method. The next section summarizes the results and finding and gives recommendations based on the same.

Findings and Recommendations

Based on the experimental study above, this section summarizes the major finding of this research. These findings will be categorized according to the objectives of this study and according to the two types of experiments conducted on the GCC stock markets. These findings will include also the overall results of the comparison studies on three prediction and classification algorithms. The output of the study assumes that most of the technical indicators which have been included have significant impact on the market direction. However, the financial indicators might have also significant impact on the results when it is included to this study. Therefore, these findings are based on the technical indicators which are considered as input to the forecasting model.

First of all, based on the accuracy hit ratios and confusion matrix of Support Vector Machine algorithm in all markets, it can be concluded that Support Vector Machine outperforms in predicting the direction of all GCC stock indexes movements using both types of experiments test-split 75% and 10-fold cross-validation method. The accuracy average rates were recorded between 72.06% and 83.42% in all markets. Also, the incorrectly classified ratios were recorded between 16.58% and 27.94% in all markets which considered as significant indicators compared to other traditional forecasting models.

Support Vector Machine has produced a comparative and optimized forecasting model which can be found in table 3. Weka software is one of the best software which is evaluating the algorithm since it has taken 0.05 seconds in building the model. In addition to that, Support Vector Machine shows the highest performance in predicting MSM index direction when it was classified the two directions correctly with 83.42% accuracy rate. On the contrast, lowest performance was recorded in predicting the direction movements of BHM stock index with 72.06% accuracy rate.

Another interesting finding in evaluating the Support Vector Machine as classification and prediction algorithm is that the performance of the algorithm has increased when 10-fold cross-validation is applied as testing and sampling method. The accuracy rates are better than the one in test-split 75% method in all markets except DFM where there was a slight decrease of the hit ratio by around 1% when the method has been changed to 10-fold cross-validation.

Based on the confusion matrix of Support Vector Machine algorithm, SVM perform well in classifying and predicting the “UP” direction in ADX, BHM, DFM, DSM, and TADAWUL using both split 75% train mode and 10-fold cross-validation train method. However, SVM perform well in classifying and predicting “DOWN” direction in KSE and MSM. In the contrast, SVM algorithm did not perform well in classifying the DOWN direction for TADAWUL index where only 49% of the total DOWN instances was classified correctly and 51% of the same are misclassified using the 75% split mode train. The same can be said for 10-fold cross-validation experiment where only 57% of the total DOWN instances is classified correctly using the algorithm and 43% are misclassified.

According to the analytical and comparison study of the three classification algorithms, Logistic Regression Algorithm (Logit) outperform in all markets where the accuracy rates crossed 80% in most of the markets. However, the better performance for this model applied on DFM where the accuracy rate is 85.66%. It is followed by 85.54% in KSE, 85.36% in MSM, 82.77% in DSM, and 82.61% in ADX. Random Forest decision tree recorded the lowest accuracy hit ratios between the three models. Random Forest performs well only in DSM, MSM, and DFM where the prediction accuracy rates are 81.98%, 81.62%, and 80.48% respectively.

SVM and Logistic Regression are performing well in ADX, while the accuracy rate for Random forest is less. The performance of all models is little bit less when it is applied on BHM market. The logistic Regression has better accuracy rate in BHM compared to SVM and Random Forest. All models are outperforming when it is applied on DFM market data. However, Logistic Regression outperforms other forecasting models in Dubai Financial Market (DFM). The best performance was for Logistic Regression Algorithm. Logit model and Random Forest classification models have better performance than SVM in DSM stock market.

After the comparison study, it has been found also that the performance of Support Vector Machines has been increased in all markets except DFM where the accuracy hit ratio decreased form 82.07% to 81.16% and slight increase of both error measures when the testing applied using 10-fold cross-validation. All models performance has been increased when 10-fold cross-validation testing methods is applied. In BHM market, the performance of both Support Vector Machine (SVM) and Logit model has been increased significantly by around 5%.

In conclusion, Support Vector Machine outperform Random Forest algorithm in classifying and predicting the direction of GCC stock indexes using 10-fold cross-validation testing method. On the same side, Support Vector Machine outperforms Random Forest algorithm when split 75.0% train on ADX, DFM, and KSE time series. However, Logistic Regression outperforms Support Vector Machine and Random Forest in all markets for both experimental studies.

Based on the findings above, this study suggests that Support Vector Machine (SVM) perform well in predicting the direction of indexes movement in GCC stock markets. Therefore, Support Vector Machine can assist brokers, portfolios managers, and traders to develop financial decision systems. In addition to that, developing trading systems for the brokerage companies in the GCC which have SVM and other Artificial Intelligence techniques might attract the investors and online traders and increase their confidence to trade with these companies and indeed bring more commission and revenue.

After examining the prediction models, it has been found that SVM and Logistic Regression have the highest accuracy hit ratio which crossed 80% and the lowest prediction error around 15%. So, Minimizing the error and maximizing the prediction accuracy rate using these models will help the investors in building different trading strategies based on the predicted direction and make a profit from these strategies. This study is the first study in the GCC which is applying and implementing the SVM as learning machine to predict the direction of stock indexes. Support Vector Machine SVM provides a promising alternative to traditional forecasting models. Further research might include the financial indicators as input variables to the SVM algorithm. Further research can be focused on other learning machines such as Neural Network and Back Propagation algorithm (BP) as advanced classification techniques in the GCC financial markets.

Conclusion

Predicting the direction of stock index is a challenging and important task for the participants in the financial markets. Many variables are affecting the fluctuation of GCC market indexes. Technical indicators have been used as input variables to the forecasting models in the recent years. This study applied the new advanced forecasting model on the GCC stock indexes Support Vector Machine (SVM) along with other learning algorithms Logistic Regression and Random Forest Decision Tree to predict the direction of general index. Two types of sampling methods have been conducted to run experiment. The forecasting model was produced for ADX, DSM, DFM, MSM, KSE, and TADAWUL time series. Analytical study on the model output has been discussed. This study conducted a comparison study between three SVM and another two classification models to evaluate the performance of SVM. This study showed that SVM outperform in predicting the direction of GCC stock indexes with 80% average accuracy rate.

In this study, the performance of SVM, Logistic Regression, and Random Forest has been evaluated. The comparison study suggests that Support Vector Machine outperforms Random Forest algorithm in predicting the direction of GCC stock indexes using 10-fold cross-validation as sampling and testing methods. Also, Support Vector Machine outperforms Random Forest algorithm when split 75.0% train used on ADX, DFM, and KSE time series. In contrast, Logistic Regression outperforms Support Vector Machine and Random Forest in all markets for both sampling methods. In addition to that, this study has analyzed the accuracy of forecasting models for all GCC stock indexes. The analytical study showed that SVM had better performance in predicting the direction of MSM stock index. Moreover, Logistic Regression has better performance in predicting DFM stock index movements. However, random Forest showed a high performance in predicting DSM stock index movements.

References

Abraham, A., Philip, N., & Saratchandran, P. (2003). Modeling Chaotic Behavior of Stock Indices Using Intelligent Paradigms [online]. [Accessed 29 May 2013]. Available at: http://www.svms.org/finance/

Auria1, L. . Moro, R. (2008). Support Vector Machines (SVM) as a Technique for Solvency Analysis [online]. [Accessed 29 May 2013]. Available at: http://hdl.handle.net/10419/27334

Bildirici, M. . Ersin, O. (2012). Support Vector Machine GARCH and Neural Network GARCH Models in Modeling Conditional Volatility: An Application to Turkish Financial Markets [online]. [Accessed 28 May 2013]. Available at: http://ssrn.com/abstract=2227747

Bruder, B., Dao, T., & Roncalli, T. (2011). Support Vector Machine in Finance [online]. [Accessed 09 December 2013]. Available at: http://daotunglam.free.fr/952FD6C4-0350-46B5-A812-5804BB61CE93/FinalDownload/DownloadId-3CCE326AC423B3A2C7EC239029B75ABB/952FD6C4-0350-46B5-A812-5804BB61CE93/Works/SVMinFinance.pdf

Calvo, R. . Williams, K. (2002). Automatic Categorization of Announcements on the Australian Stock Exchange [online]. [Accessed 04 July 2013]. Available at: http://www.svms.org/finance/

Cao, L. & Tay, F. (2003). Support Vector Machine With Adaptive Parameters in Financial Time Series Forecasting, IEEE Transactions On Neural Networks, Vol. 14(6),PP. 1506-1518.

Cao, L. (2002). Support vector machines experts for time series forecasting [online]. [Accessed 20 June 2013]. Available at: http://www.svms.org/finance/

Cao, L. . Tay, F. (2001). Financial Forecasting Using Support Vector Machines [online]. [Accessed 20 June 2013]. Available at: http://www.svms.org/finance/

Chen, W. & Shih, J. (2006). Comparison of support-vector machines and back propagation neural networks in forecasting the six major Asian stock markets, Int. J. Electronic Finance, Vol. 1(1),PP. 49-65.

Chen, W. . Shih, J. (2011). A study of Taiwan's issuer credit rating systems using support vector machines [online]. [Accessed 04 July 2013]. Available at: http://www.svms.org/finance/

Chena, G., Leungb, M., & Daoukc, H. (2003). Application of Neural Networks to an Emerging Financial Market: Forecasting and Trading the Taiwan Stock Index [online]. [Accessed 11 November 2013]. Available at: http://ssrn.com/abstract=237038

Cortes, C. . Vapnik, V. (1995). Support-Vector Networks , Machine Learning , Vol. 20(3),PP. 273-297.

Cruz, F., Rodrıguez, J., & Giner, J. (2003). Estimating GARCH models using support vector machines [online]. [Accessed 04 July 2013]. Available at: http://www.svms.org/finance/

Dutta, A., Sengupta,A.,& Bandopadhyay, G. (2012). Prediction of Stock Performance in the Indian Stock Market Using Logistic Regression, International Journal of Business and Information, Vol. 7(1),PP. 106-132.

Emir, S., Dinçer, H., & Timor, M. (2012). A Stock Selection Model Based on Fundamental and Technical Analysis Variables by Using Artificial Neural Networks and Support Vector Machines [online]. [Accessed 09 December 2013]. Available at: http://www.econbiz.de/Record/stock-selection-model-based-fundamental-and-technical-analysis-variables-using-artificial-neural-networks-and-support-vector-machines-emir-%C5%9Fenol/10010078136

Fama, E. (1970). Efficient Capital Markets: A Review Of Theory And Empirical Work , The Journal of Finance , Vol. 25(2),PP. 383-417.

Fan, A. . Palaniswami, M. (2001). Stock Selection using Support Vector Machines [online]. [Accessed 09 December 2013]. Available at: http://cbio.ensmp.fr/~jvert/svn/bibli/local/Fan2001Stock.pdf

Haiqin, Y. (2003). Margin Variations in Support Vector Regression
for the Stock Market Prediction [online]. [Accessed 04 July 2013]. Available at: http://www.svms.org/finance/

Hájek, P. (2012). Forecasting Stock Market Trend using Prototype Generation Classifiers, Wseas Transactions On Systems, Vol. 11(12),PP. 671-677.

Han, S. & Chen, R. (2007). Using SVM with Financial Statement Analysis for Prediction of Stocks, Communications of the IIMA, Vol. 7(4),PP. 63-71.

Harland, Z. (2002). Using Support Vector Machines to Trade Aluminium on
the LME [online]. [Accessed 04 July 2013]. Available at: http://www.svms.org/finance/

Huang, W., Wang, S., Yu,L., Bao,Y., & Wang, L. (2006). A New Computational Method of Input Selection for Stock Market Forecasting with Neural Networks [online]. [Accessed 26 November 2013]. Available at http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.88.6190&rep=rep1&type=pdf

Huanga, W., Nakamoria, Y., & Wangb, S. (2004). Forecasting Stock Market Movement Direction With Support Vector Machine [online]. [Accessed 20 June 2013]. Available at: http://www.svms.org/finance/

Huanga, Z., Chena, H., Hsua,C., Chenb,W., & Wuc, S. (2004). Credit rating analysis with support vector machines and neural networks: a market comparative study [online]. [Accessed 29 May 2013]. Available at: http://www.svms.org/finance/

Huertaa, R., Corbachob, F., & Elkanc, S. (2013). Nonlinear Support Vector Machines Can Systematically Identify Stocks with High and Low Future Returns [online]. [Accessed 28 May 2013]. Available at: http://papers.ssrn.com/sol3/papers.cfm?abstract_id=1930709

Kara, Y., Boyacioglu, M., & Baykan, O. (2011). Predicting Direction Of Stock Price Index Movement Using Artificial Neural Networks And Support Vector Machines: The sample of the Istanbul Stock Exchange , Expert Systems with Applications, Vol. 38(5),PP. 5311-5319.

Kaur, S. & Mangat, V. (2012 Improved Accuracy of PSO and DE using Normalization: an Application to Stock Price Prediction, International Journal of Advanced Computer Science and Applications, Vol. 3(9),PP. 197-205.

Kim, K. (2003). Financial time series forecasting using support vector machines [online]. [Accessed 29 May 2013]. Available at: http://www.svms.org/finance/

King, C., Vandrot, C., & Weng, J. (2010). A SVM Approach To Stock Trading [online]. [Accessed 09 December 2013]. Available at: http://cs229.stanford.edu/proj2009/KingVandrotWeng.pdf

Kumar, M. . Thenmozhi, M. (2006). Forecasting Stock Index Movement: A Comparision Of Support Vector Machines And Random Forest [online]. [Accessed 28 May 2013]. Available at: http://www.svms.org/finance/

Lin, Y., Guo, H., & Hu, J. (2013). An SVM-based Approach for Stock Market Trend Prediction [online]. [Accessed 09 December 2013]. Available at: http://www.hflab.ips.waseda.ac.jp/~jinglu/Publics/ijcnn2013Yuling.pdf

Lunga, D. . Marwala, T. (2006). Online Forecasting of Stock Market Movement Direction Using the Improved Incremental Algorithm [online]. [Accessed 26 November 2013]. Available at: http://link.springer.com/chapter/10.1007%2F11893295_49#page-1

Majumder, M. . Hussian, M. (2013). Forecasting Of Indian Stock Market Index Using Artificial Neural Network [online]. [Accessed 26 November 2013]. Available at:

http://www.nseindia.com/content/research/FinalPaper206.pdf

Nalbantov, G., Bauer, R., & Kuyper, S. (2005). Equity Style Timing using Support Vector Regressions [online]. [Accessed 04 July 2013]. Available at: http://www.svms.org/finance/

Niaki, S. & Hoseinzade, S. (2013). Forecasting S&P 500 index using artificial neural networks and design of experiments, Journal of Industrial Engineering International, Vol. 9(1),PP. 2-9.

Olatunji, S., AlAhmadi, M., Elshafei,M., & Fallatah, Y. (2013). Forecasting the Saudi Arabia stock prices based on artificial neural networks model, International Journal of Intelligent Information Systems, Vol. 2(5),PP. 77-86.

Rodriguez, P. . Rodriguez, A. (2004). Predicting stock market indices movements [online]. [Accessed 26 November 2013]. Available at: http://www.researchgate.net/publication/228289340_Predicting_Stock_Market_Indices_Movements

Senyurt, G. . Subasi, A. (2012). Stock Market Movement Direction Prediction Using Tree Algorithms [online]. [Accessed 26 November 2013]. Available at: http://eprints.ibu.edu.ba/1187/1/41.%20Stock%20market%20movement%20direction%20prediction%20using%20tree%20algorithms.pdf

Shah, V. (2007). Machine Learning Techniques for Stock Prediction [online]. [Accessed 09 June 2013]. Available at: http://www.vatsals.com/Essays/MachineLearningTechniquesforStockPrediction.pdf

Shin, K., Lee, T., & Kim, H. (2005). An Application Of Support Vector Machines In Bankruptcy Prediction Model [online]. [Accessed 14 November 2013]. Available at: http://www.svms.org/finance/

Tay, F. . Cao, L. (2001 Modified Support Vector Machines in Financial Time Series Forecasting [online]. [Accessed 20 June 2013]. Available at: http://www.sciencedirect.com/science/article/pii/S0925231201006762

Trafalis, T. . Ince, H. (2000). Support Vector Machine For Regression And Applications To Financial Forecasting [online]. [Accessed 04 June 2013]. Available at: http://www.svms.org/finance/

Wang, Y. . Choi, I. (2013). Market Index and Stock Price Direction Prediction using Machine Learning Techniques: An empirical study on the KOSPI and HSI [online]. [Accessed 26 November 2013]. Available at: http://arxivweb3.library.cornell.edu/abs/1309.7119?context=cs.LG

Yang, H., King, I., Chan,L., & Huang, K. (2002). Financial Time Series Prediction Using Non-fixed and Asymmetrical Margin Setting with Momentum in Support Vector Regression [online]. [Accessed 07 July 2013]. Available at: http://www.svms.org/finance/

Yang, H., King, I., Chan,L., & Huang, K. (2002). Support Vector Machine Regression for Volatile Stock Market Prediction [online]. [Accessed 04 November 2013]. Available at: http://link.springer.com/chapter/10.1007/3-540-45675-9_58#page-1

Yildiz, B., Yalama, A., & Coskun, M. (2008). Forecasting the Istanbul Stock Exchange National 100 Index Using an Artificial Neural Network [online]. [Accessed 16 November 2013]. Available at: http://waset.org/publications/11383

Yu, L., Wang, S., & Lai, K. (2005). Mining Stock Market Tendency Using GA-Based Support Vector Machines [online]. [Accessed 04 July 2013]. Available at: http://www.svms.org/finance/

Yuan, C. (2011). Predicting S&P 500 Returns Using Support Vector Machines: Theory and Empirics [online]. [Accessed 05 June 2013]. Available at: http://apps.olin.wustl.edu/cres/research/calendar/files/YuanPaper.pdf

Appendices

Appendix 1

Appendix 1 shows sample of data downloaded for DFM stock market.

	A	B	C	D	E
1	**<Date>**	**<Open>**	**<High>**	**<Low>**	**<Close>**
1602	20091130	2093.15991	2093.15991	1940.35999	1940.35999
1603	20091201	1940.35999	1940.35999	1809.98999	1831.47998
1604	20091206	1794.01001	1877.72998	1789.72998	1853.13
1605	20091207	1837.02002	1837.02002	1736.62	1744.82996
1606	20091208	1716.08997	1720.20996	1630.46997	1638.05005
1607	20091209	1585.33997	1592.85999	1528.92004	1533.35999
1608	20091210	1539.80005	1641.75	1461.75	1640.76001
1609	20091213	1655.89001	1734.58997	1645.92004	1695.34998
1610	20091214	1866.60999	1871.19995	1864.89001	1871.19995
1611	20091215	2000.71997	2000.71997	1839.73999	1843.27002
1612	20091216	1852.52002	1892.05005	1829.28003	1889.98999
1613	20091217	1881.48999	1894.68994	1861.41003	1879.26001
1614	20091220	1877.08997	1891.96997	1824.35999	1831.41003
1615	20091221	1827.44995	1834.37	1793.95996	1827.52002
1616	20091222	1841.32996	1846.65002	1790.64001	1803.31006
1617	20091223	1796.79004	1808.53003	1724.34998	1735.69995
1618	20091224	1737.33997	1776.63	1726.56006	1759.20996
1619	20091227	1766.98999	1782.16003	1746.29004	1769.27002
1620	20091228	1776.35999	1833.08997	1765.55005	1828.63
1621	20091229	1832.18005	1838.66003	1784.38	1789.26001
1622	20091230	1787.02002	1810.92004	1783.33997	1810.23999
1623	20091231	1814.34998	1824.37	1797.85999	1803.57996

Appendix 2

Appendix 2 illustrates the calculation of technical indicators for DFM on 31/12/2009 as example using Excel Sheet. It shows also the results for one record as of 31/12/2009

- StochasticK = F1623 = ((E1623-MIN(D1610:D1623))/(MAX(C1610:C1623)-MIN(D1610:D1623)))*100
- StochasticD = G1623 =AVERAGE(F1621:F1623)
- SlowD:= H1623 =AVERAGE(G1622:G1623)
- ROC= I1623= (E1623/(E1612))*100
- WilliamsR= J1623 =((MAX(C1610:C1623)-E1623)/(MAX(C1610:C1623)-MIN(D1610:D1623)))*100
- Disparity5= K1623 =(E1623/(AVERAGE(E1619:E1623)))*100
- Disparity10= L1623 =(E1623/(AVERAGE(E1614:E1623))*100)
- OSCP= M1623=(AVERAGE(E1614:E1618)-AVERAGE(E1614:E1623))/(AVERAGE(E1614:E1618))
- 20-day CCI: =(N1623-O1623)/(0.015*P1623)

  1) P1623= =(ABS(O1623-N1604)+ABS(O1623-N1605)+ABS(O1623-N1606)+ABS(O1623-N1607)+ABS(O1623-N1608)+ABS(O1623-N1609)+ABS(O1623-N1610)+ABS(O1623-N1611)+ABS(O1623-N1612)+ABS(O1623-N1613)+ABS(O1623-N1614)+ABS(O1623-N1615)+ABS(O1623-N1616)+ABS(O1623-N1617)+ABS(O1623-N1618)+ABS(O1623-N1619)+ABS(O1623-N1620)+ABS(O1623-N1621)+ABS(O1623-N1622)+ABS(O1623-N1623))/20
  2) O1623 =AVERAGE(N1604:N1623)
  3) N1623= =AVERAGE(C1623:E1623)
- 14-day RSI=IF(V1623=0,100,100-(100/(1+W1623)))

  1) X1623 =IF(V1623=0,100,100-(100/(1+W1623)))
  2) W1623 =U1623/V1623
  3) V1623 =AVERAGE(T1610:T1623)
  4) U162 =AVERAGE(S1610:S1623)
  5) T1623 =IF(R1623<0,-R1623,0)

6) S1623 =IF(R1623>0,R1623,0)

7) R1623 =E1623-E1622

- Classifier =IF(E1623>E1622,"UP",IF(E1623<E1622,"DOWN")

Results for 31/12/2009

StochasticK	28.66808
StochasticD	45.24914
SlowD	51.81622
ROC	95.42802
WilliamsR	-71.3319
Disparity5	100.188
Disparity10	100.4325
OSCP	-0.00245
20-day CCI	27.47343
14-day RSI	59.58245
Classifier	DOWN

Appendix 3

Appendix 4 describes the total number of instances for each market for 4 years period

Financial Years	Market	Direction UP	Direction Down	No Direction	% UP	% Down	%False	Total
2010-2013	ADX	538	465	9	53%	46%	0.9%	1012
2010-2013	DFM	514	486	3	51%	48%	0.3%	1003
2010-2013	BHM	505	469	14	51%	47%	1.4%	988
2010-2013	DSM	674	849	8	44%	55%	0.5%	1531
2010-2013	KSE	459	531	6	46%	53%	0.6%	996
2010-2013	MSM	557	717	11	44%	57%	0.9%	1258
2010-2013	TADAWUL	592	403	1	59%	40%	0.1%	996

Appendix 4

Appendix 4 shows the summary statistics of each indicator for each market

DFM

Indicator	Minimum	Maximum	Mean	StdDev
StochasticK	0	100	53.825	32.288
StochasticD	2.743	100	53.777	30.504
SlowD	3.144	99.081	53.759	30.168
ROC	83.688	116.847	100.76	4.72
WilliamsR	-100	0	-46.175	32.288
Disparity5	94.177	105.746	100.126	1.42
Disparity10	91.698	107.441	100.287	2.289
OSCP	-0.044	0.034	-0.002	0.013
20-day CCI	-323.666	361.205	4.879	112.987
14-day RSI	1.976	96.592	52.982	19.531

BHM

Indicator	Minimum	Maximum	Mean	StdDev
StochasticK	-175.716	328.61	46.592	43.048
StochasticD	-78.949	209.791	46.545	38.086
SlowD	-64.039	202.059	46.529	37.411
ROC	91.781	105.058	99.828	1.798
WilliamsR	-275.716	228.61	-53.408	43.048
Disparity5	97.288	101.956	99.969	0.559
Disparity10	95.004	103.318	99.925	0.891
OSCP	-0.015	0.025	0	0.005
20-day CCI	-359.728	406.67	-4.84	116.31
14-day RSI	3.954	99.081	47.624	18.956

ADX

Indicator	Minimum	Maximum	Mean	StdDev
StochasticK	-4.601	104.811	54.764	35.39
StochasticD	-1.405	104.255	54.725	33.798
SlowD	-1.126	102.485	54.707	33.475
ROC	1.3	8025.844	115.779	348.106
WilliamsR	-104.601	4.811	-45.236	35.39
Disparity5	1.637	166.894	100.096	5.519
Disparity10	1.444	127.734	100.207	5.123
OSCP	-0.346	0.196	-0.002	0.024
20-day CCI	-666.667	343.025	8.152	116.487
14-day RSI	1.412	99.53	53.211	22.323

DSM

Indicator	Minimum	Maximum	Mean	StdDev
StochasticK	-4.601	104.811	60.274	32.947
StochasticD	-1.405	104.255	60.236	31.27
SlowD	-1.126	102.485	60.218	30.943
ROC	1.3	8025.844	110.267	283.157
WilliamsR	-104.601	4.811	-39.726	32.947
Disparity5	1.637	166.894	99.979	5.49
Disparity10	1.444	127.734	99.952	6.279
OSCP	-0.346	0.385	0	0.029
20-day CCI	-666.667	323.987	26.712	115.297
14-day RSI	0.328	99.53	56.269	20.807

KSE

Indicator	Minimum	Maximum	Mean	StdDev
StochasticK	1.282	100	57.104	33.64
StochasticD	3.304	99.925	57.154	32.361
SlowD	4.35	99.844	57.184	32.104
ROC	89.093	109.183	100.12	2.677
WilliamsR	-98.718	0	-42.896	33.64
Disparity5	96.684	102.942	100.016	0.761
Disparity10	94.098	104.516	100.038	1.291
OSCP	-0.027	0.026	0	0.007
20-day CCI	-419.505	278.311	8.166	118.158
14-day RSI	0.279	99.266	52.548	23.429

MSM

Indicator	Minimum	Maximum	Mean	StdDev
StochasticK	0	100	57.769	35.18
StochasticD	0	100	57.83	33.687
SlowD	0.035	99.982	57.859	33.368
ROC	82.794	109.183	100.22	3.072
WilliamsR	-100	0	-42.231	35.18
Disparity5	84.694	102.942	100.03	0.972
Disparity10	83.74	104.516	100.074	1.555
OSCP	-0.027	0.086	0	0.008
20-day CCI	-641.34	306.055	23.286	114.462
14-day RSI	0.279	99.266	55.476	22.912

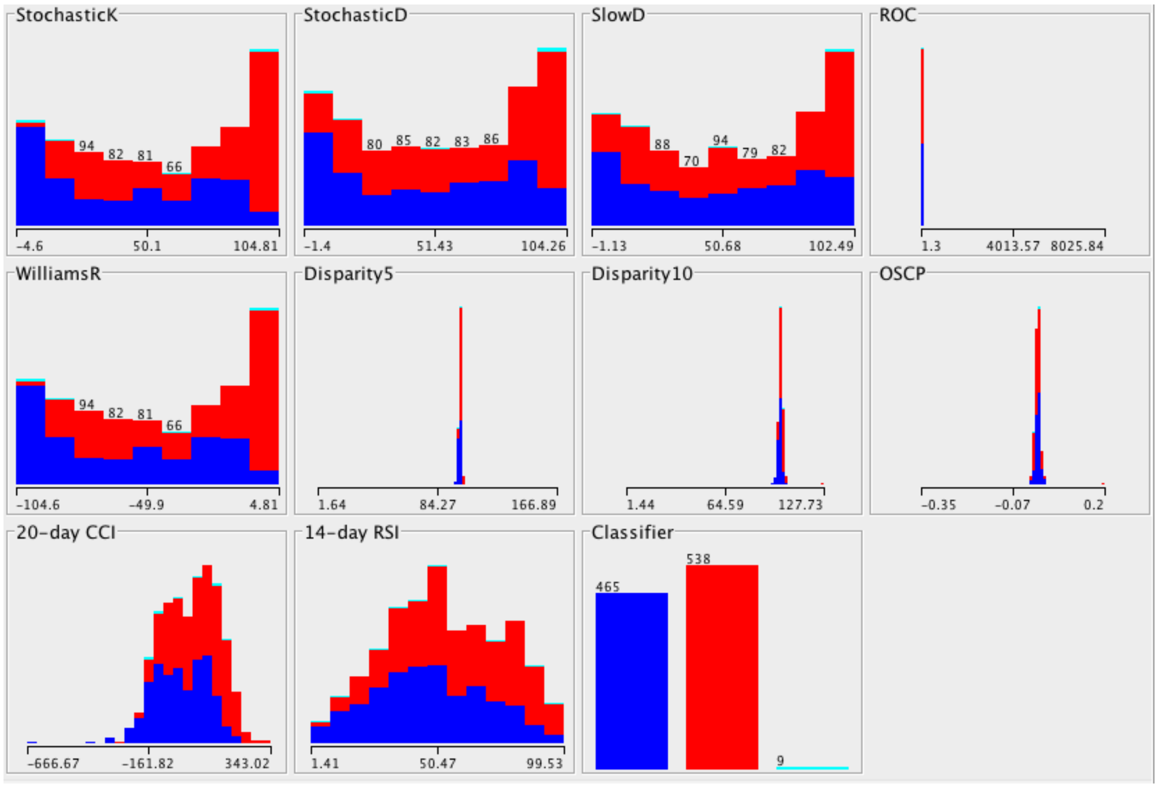
TADAWUL

Indicator	Minimum	Maximum	Mean	StdDev
StochasticK	-398.798	100	35.596	56.32
StochasticD	-161.74	100	35.565	50.827
SlowD	-154.287	100	35.567	49.94
ROC	79.427	118.738	100.411	3.459
WilliamsR	-498.798	0	-64.404	56.32
Disparity5	91.188	106.332	100.067	1.197
Disparity10	86.819	109.217	100.15	1.555
OSCP	-0.06	0.058	-0.001	0.01
20-day CCI	-435.542	260.902	26.221	108.435
14-day RSI	3.983	99.239	57.028	19.928

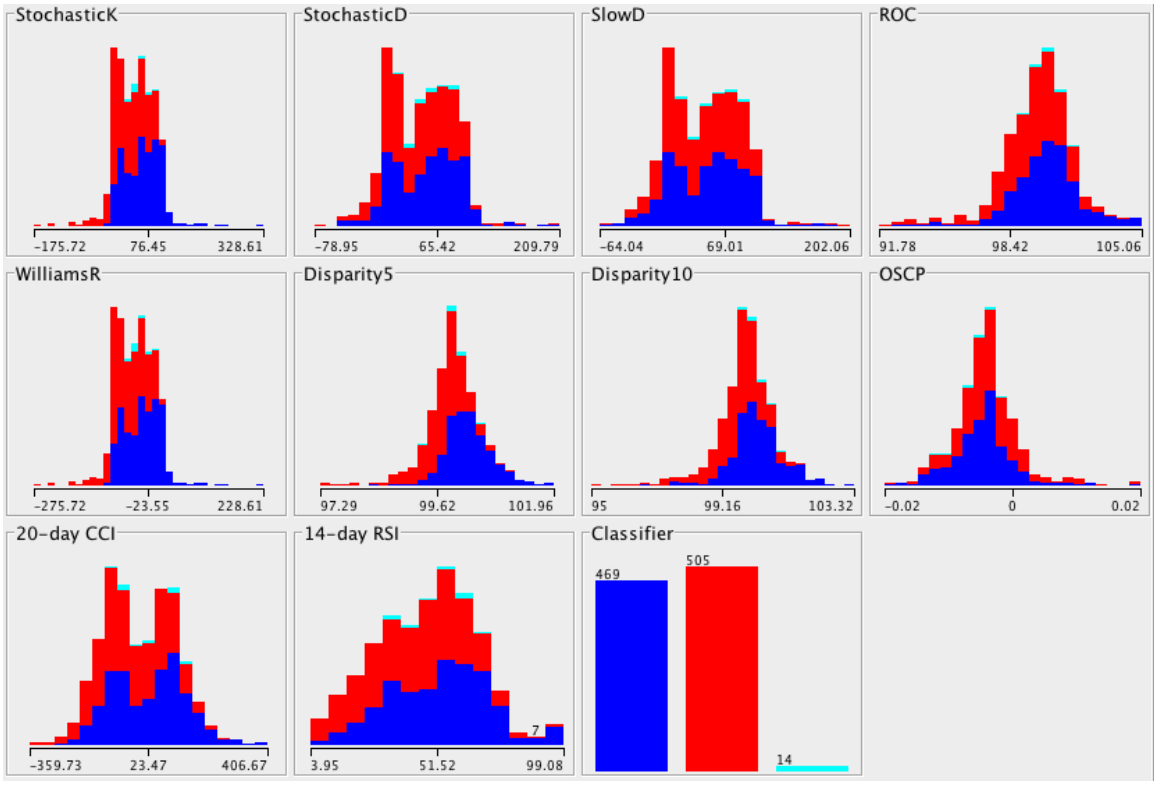
Appendix 5

Appendix 5 shows the distribution charts for each attribute according to the nominal classifier (UP, DOWN). Red color refer to UP direction instances while blue color refer to DOWN direction instances

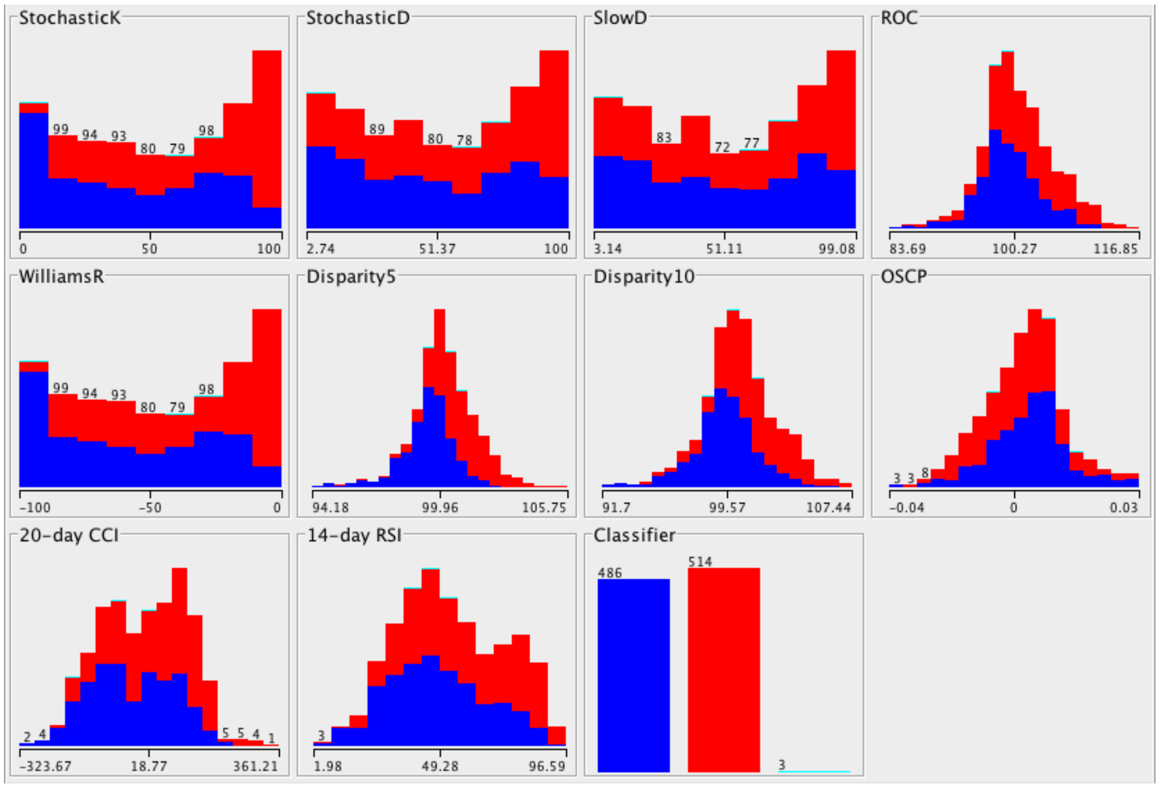
ADX



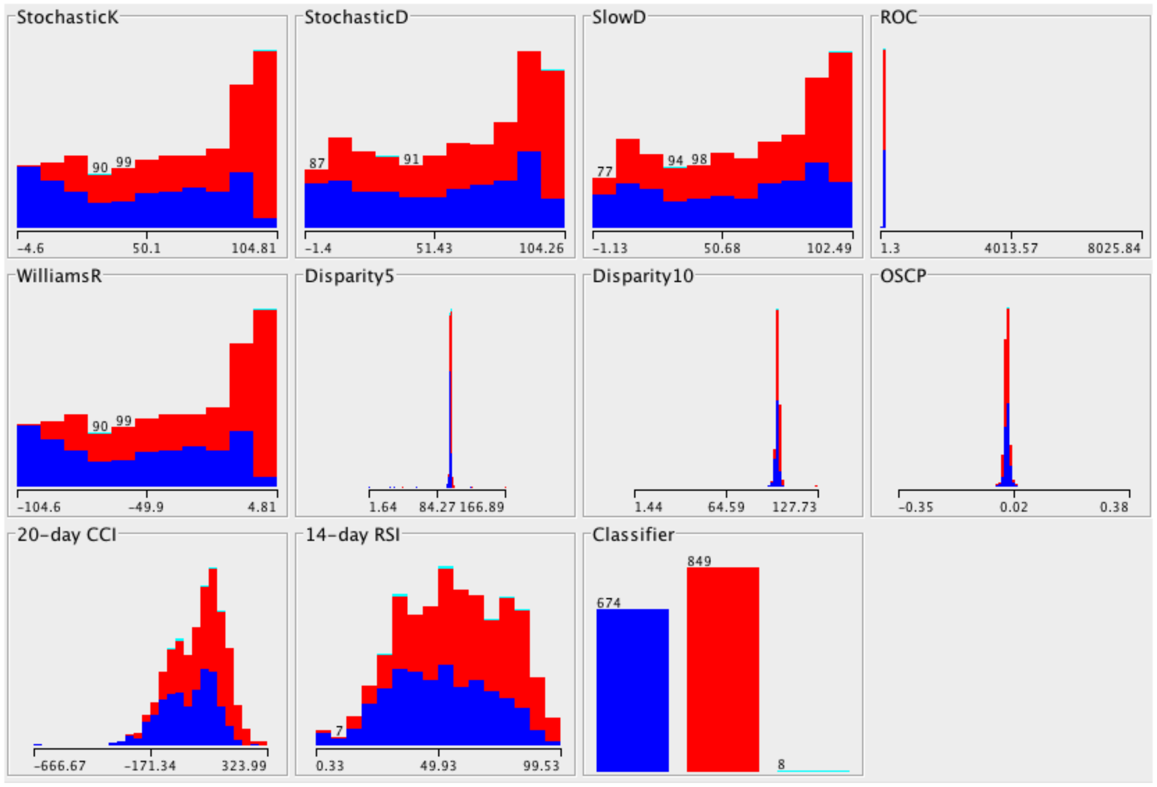
BHM


StochasticK
-175.72
76.45
328.61
StochasticD
-78.95
65.42
209.79
SlowD
-64.04
69.01
202.06
ROC
91.78
98.42
105.06
WilliamsR
-275.72
-23.55
228.61
Disparity5
97.29
99.62
101.96
Disparity10
95
99.16
103.32
OSCP
-0.02
0
0.02
20-day CCI
-359.73
23.47
406.67
14-day RSI
7
3.95
51.52
99.08
Classifier
469
505
14

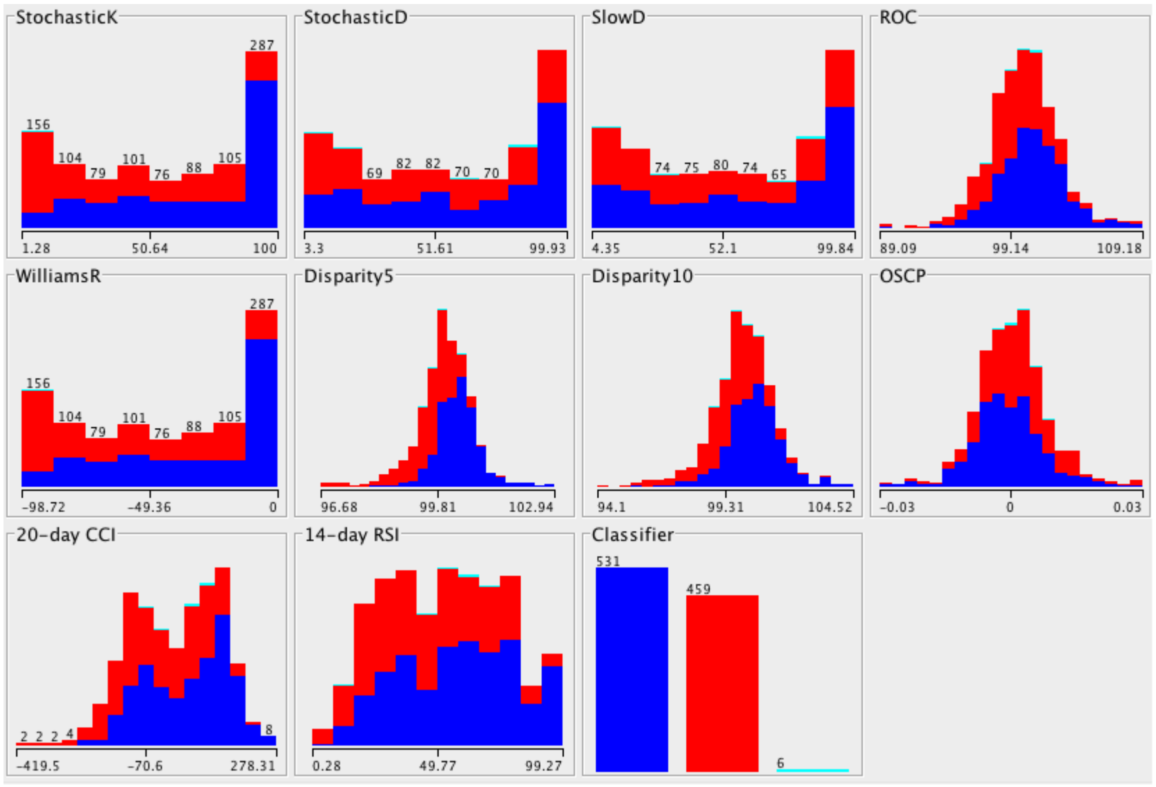
DFM


StochasticK
99 94 93 80 79 98
0
50
100
StochasticD
89 80 78
2.74
51.37
100
SlowD
83 72 77
3.14
51.11
99.08
ROC
83.69
100.27
116.85
WilliamsR
99 94 93 80 79 98
-100
-50
0
Disparity5
94.18
99.96
105.75
Disparity10
91.7
99.57
107.44
OSCP
3 3 8
-0.04
0
0.03
20-day CCI
2 4 5 5 4 1
-323.67
18.77
361.21
14-day RSI
3
1.98
49.28
96.59
Classifier
486
514
3

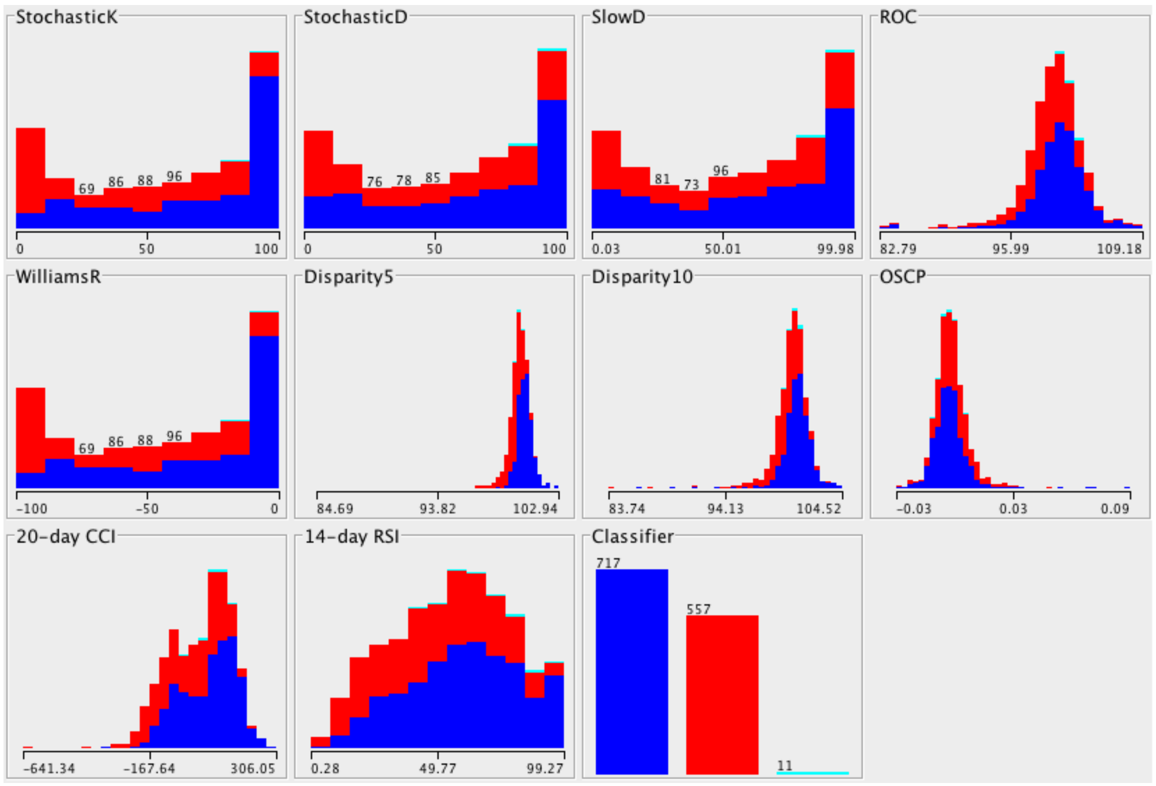
DSM


StochasticK
90
99
-4.6
50.1
104.81
StochasticD
87
91
-1.4
51.43
104.26
SlowD
77
94
98
-1.13
50.68
102.49
ROC
1.3
4013.57
8025.84
WilliamsR
90
99
-104.6
-49.9
4.81
Disparity5
1.64
84.27
166.89
Disparity10
1.44
64.59
127.73
OSCP
-0.35
0.02
0.38
20-day CCI
-666.67
-171.34
323.99
14-day RSI
7
0.33
49.93
99.53
Classifier
674
849
8

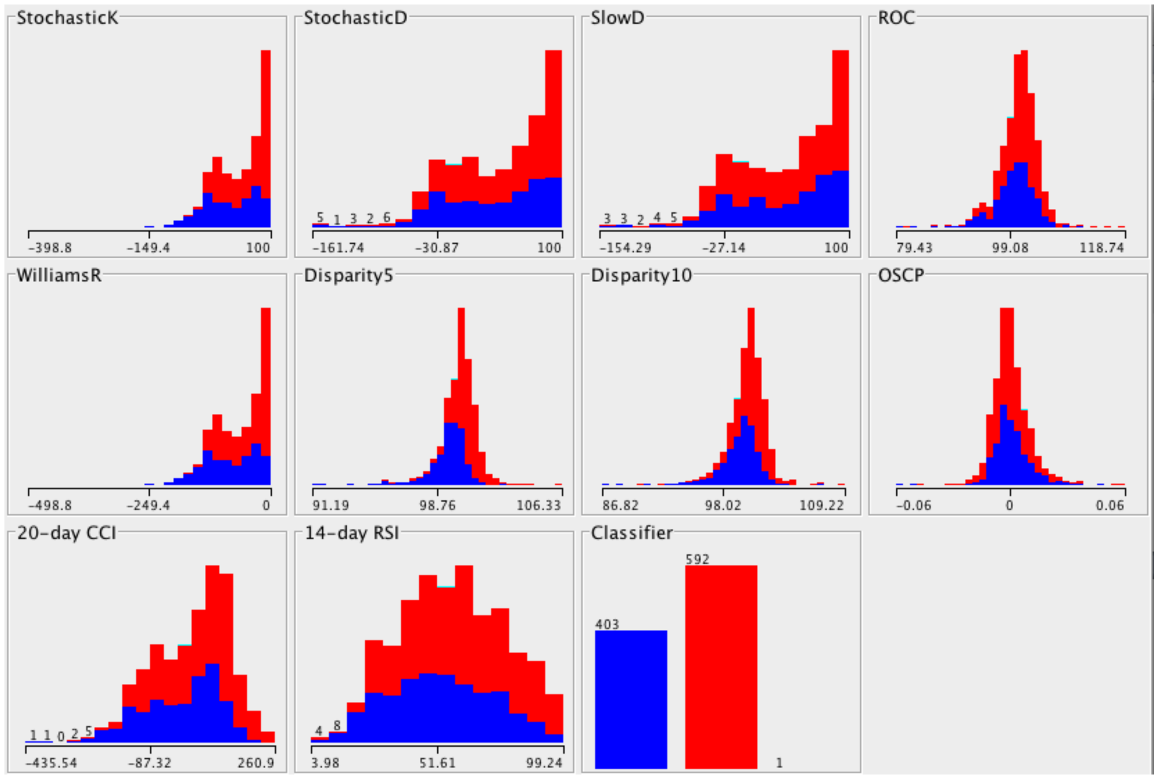
KSE


StochasticK
287
156
104
79
101
76
88
105
1.28
50.64
100
StochasticD
69
82
82
70
70
3.3
51.61
99.93
SlowD
74
75
80
74
65
4.35
52.1
99.84
ROC
89.09
99.14
109.18
WilliamsR
287
156
104
79
101
76
88
105
-98.72
-49.36
0
Disparity5
96.68
99.81
102.94
Disparity10
94.1
99.31
104.52
OSCP
-0.03
0
0.03
20-day CCI
2 2 2 4
8
-419.5
-70.6
278.31
14-day RSI
0.28
49.77
99.27
Classifier
531
459
6

MSM


StochasticK
69
86
88
96
0
50
100
StochasticD
76
78
85
0
50
100
SlowD
81
73
96
0.03
50.01
99.98
ROC
82.79
95.99
109.18
WilliamsR
69
86
88
96
-100
-50
0
Disparity5
84.69
93.82
102.94
Disparity10
83.74
94.13
104.52
OSCP
-0.03
0.03
0.09
20-day CCI
-641.34
-167.64
306.05
14-day RSI
0.28
49.77
99.27
Classifier
717
557
11

TADAWUL


StochasticK
-398.8
-149.4
100
StochasticD
5 1 3 2 6
-161.74
-30.87
100
SlowD
3 3 2 4 5
-154.29
-27.14
100
ROC
79.43
99.08
118.74
WilliamsR
-498.8
-249.4
0
Disparity5
91.19
98.76
106.33
Disparity10
86.82
98.02
109.22
OSCP
-0.06
0
0.06
20-day CCI
1 1 0 2 5
-435.54
-87.32
260.9
14-day RSI
4 8
3.98
51.61
99.24
Classifier
403
592
1

Appendix 6

Appendix shows the weights for all attributes for Support Vector Machine forecasting model using 10-fold cross-validation testing method as experiment 1 which producing the prediction model. The sign “+” refer to the positive relation of each indicator and the sign “-“refer to the negative relation of each indicator.

MARKET	ADX	BHM	DFM	DSM	KSE	MSM	TADAWUL
StochasticK	(+)5.6444	(+)-5.1875	(+)4.75	(+)5.8765	(+)-4.6135	(+)-5.4018	(+)5.2576
StochasticD	(+)-4.8096	(+)2.3273	(+)-5.4106	(+)-6.1534	(+)5.2953	(+)5.8021	(+)-2.6748
SlowD	(+)-5.923	(+)3.5658	(+)-3.8161	(+)-5.2764	(+)4.0122	(+)4.76	(+)-4.5851
ROC	(+)0.9175	(+)0.4173	(+)0.6433	(+)0.9225	(+)-0.2734	(+)0.0486	(+)0.3666
WilliamsR	(+)5.6444	(+)-5.1875	(+)4.75	(+)5.8765	(+)-4.6135	(+)-5.4018	(+)5.2576
Disparity5	(+)0.4441	(+)-6.1262	(+)2.4948	(+)0.1802	(+)-3.2042	(+)-3.2633	(+)5.6921
Disparity10	(+)0.3915	(+)-2.5035	(+)1.0968	(+)0.4583	(+)-0.9058	(+)-1.3975	(+)2.6435
OSCP	(+)0.4348	(+)-1.9568	(+)1.5338	(+)0.022	(+)-2.0076	(+)-2.622	(+)2.2831
20-day CCI	(+)2.4413	(+)0.9107	(+)-0.673	(+)2.0584	(+)-0.7672	(+)-2.0495	(+)1.549
14-day RSI	(+)1.1645	(+)-1.6003	(+)1.9977	(+)1.7482	(+)-1.6893	(+)-1.029	(+)1.4533
b	(-)3.3378	(+)8.0091	(-)4.0442	(-)2.9221	(+)4.6624	(+)6.3258	(-)11.3991

Appendix 7

Appendix shows the overall accuracy hit ratios of Support Vector Machine (SVM) model in GCC stock markets under study

Market		ADX	BHM	DFM	DSM	KSE	MSM	TADAWUL
Evaluation on test split	Correctly Classified Instances	81.03%	72.06%	82.07%	81.20%	79.92%	79.75%	75.50%
	Incorrectly Classified Instances	18.97%	27.94%	17.93%	18.80%	20.08%	20.25%	24.50%
10-fold cross-validation	Correctly Classified Instances	82.11%	76.72%	81.16%	81.25%	80.62%	*83.42%	77.51%
	Incorrectly Classified Instances	17.89%	23.28%	18.84%	18.75%	19.38%	16.58%	22.49%
	Average Accuracy	***82%**	**74%**	***82%**	**81%**	**80%**	***82%**	**77%**

Appendix 8

Appendix 8 shows the confusion matrix for 10-fold cross-validation of SVM for 7 GCC stock indexes

Market	Down	Up	FALSE	Total	Market	Down	Up	FALSE	Total
ADX	93	25	0	118	ADX	369	96	0	465
	20	112	0	132		76	462	0	538
	3	0	0	3		4	5	0	9
Total	**116**	**137**	**0**	**253**	**Total**	**449**	**563**	**0**	**1012**
BHM	83	31	0	114	BHM	345	124	0	469
	33	95	0	128		92	413	0	505
	2	3	0	5		6	8	0	14
Total	**118**	**129**	**0**	**247**	**Total**	**443**	**545**	**0**	**988**
DFM	102	22	0	124	DFM	396	90	0	486
	22	104	0	126		96	418	0	514
	0	1	0	1		1	2	0	3
Total	**124**	**127**	**0**	**251**	**Total**	**493**	**510**	**0**	**1003**
DSM	123	49	0	172	DSM	488	186	0	674
	21	188	0	209		93	756	0	849
	1	1	0	2		2	6	0	8
Total	**145**	**238**	**0**	**383**	**Total**	**583**	**948**	**0**	**1531**
KSE	110	22	0	132	KSE	452	79	0	531
	28	89	0	117		108	351	0	459
	0	0	0	0		1	5	0	6
Total	**138**	**111**	**0**	**249**	**Total**	**561**	**435**	**0**	**996**
MSM	155	28	0	183	MSM	633	84	0	717
	34	101	0	135		118	439	0	557
	2	1	0	3		8	3	0	11
Total	**191**	**130**	**0**	**321**	**Total**	**759**	**526**	**0**	**1285**
TADAWUL	50	52	0	102	TADAWUL	231	172	0	403
	9	138	0	147		51	541	0	592
	0	0	0	0		1	0	0	1
Total	**59**	**190**	**0**	**249**	**Total**	**283**	**713**	**0**	**996**

Market	Down	Up	FALSE	Total	Market	Down	Up	FALSE	Total
ADX	79%	21%	0%	118	ADX	79%	21%	0%	465
	15%	*85%	0%	132		14%	*86%	0%	538
	100%	0%	0%	3		44%	56%	0%	9
Total	**116**	**137**	**0**	**253**	**Total**	**449**	**563**	**0**	**1012**
BHM	73%	27%	0%	114	BHM	74%	26%	0%	469
	26%	*74%	0%	128		18%	*82%	0%	505
	40%	60%	0%	5		43%	57%	0%	14
Total	**118**	**129**	**0**	**247**	**Total**	**443**	**545**	**0**	**988**
DFM	82%	18%	0%	124	DFM	81%	19%	0%	486
	17%	*83%	0%	126		19%	*81%	0%	514
	0%	100%	0%	1		33%	67%	0%	3
Total	**124**	**127**	**0**	**251**	**Total**	**493**	**510**	**0**	**1003**
DSM	72%	28%	0%	172	DSM	72%	28%	0%	674
	10%	*90%	0%	209		11%	*89%	0%	849
	50%	50%	0%	2		25%	75%	0%	8
Total	**145**	**238**	**0**	**383**	**Total**	**583**	**948**	**0**	**1531**
KSE	*83%	17%	0%	132	KSE	*85%	15%	0%	531
	24%	76%	0%	117		24%	76%	0%	459
	NA	NA	NA	0		17%	83%	0%	6
Total	**138**	**111**	**0**	**249**	**Total**	**561**	**435**	**0**	**996**
MSM	*85%	15%	0%	183	MSM	*88%	12%	0%	717
	25%	75%	0%	135		21%	79%	0%	557
	67%	33%	0%	3		73%	27%	0%	11
Total	**191**	**130**	**0**	**321**	**Total**	**759**	**526**	**0**	**1285**
TADAWUL	49%	51%	0%	102	TADAWUL	57%	43%	0%	403
	6%	*94%	0%	147		9%	*91%	0%	592
	NA	NA	NA	0		100%	0%	0%	1
Total	**59**	**190**	**0**	**249**	**Total**	**283**	**713**	**0**	**996**

Appendix 9

Appendix 9 compares and analyzes the performance of Support Vector Machine with other classifications algorithm using split 75.0% train, remainder test experiment.

Test mode: split 75.0% train, remainder test

	Market	Accuracy hit Ratio	Mean Absolute Error	Root Mean Squared Error
SVM	ADX	81.03%	0.267	0.3448
	BHM	72.06%	0.2888	0.375
	DFM	*82.07%	*0.2629	*0.3388
	DSM	81.20%	0.2652	0.3421
	KSE	79.92%	0.2668	0.3445
	MSM	79.75%	0.2693	0.3481
	TADAWUL	75.50%	0.2767	0.3585
	MAX	**82.07%**	**28.88%**	**37.50%**
	MIN	**72.06%**	**26.29%**	**33.88%**
Logit	ADX	82.61%	0.1597	0.2882
	BHM	75.71%	0.2008	0.3253
	DFM	*85.66%	*0.1389	*0.2724
	DSM	82.77%	0.1527	0.2847
	KSE	85.54%	0.1358	0.2497
	MSM	85.36%	0.1352	0.2623
	TADAWUL	80.32%	0.1704	0.2991
	MAX	**85.66%**	**20.08%**	**32.53%**
	MIN	**75.71%**	**13.52%**	**24.97%**
Random Forest	ADX	77.08%	0.1755	0.3174
	BHM	74.09%	0.2148	0.3435
	DFM	80.48%	0.175	0.3022
	DSM	*81.98%	*0.1629	*0.296
	KSE	78.71%	0.1775	0.3106
	MSM	81.62%	0.1607	0.3046
	TADAWUL	79.92%	0.1692	0.311
	MAX	**81.98%**	**21.48%**	**34.35%**
	MIN	**74.09%**	**16.07%**	**29.60%**

Appendix 10

Appendix 10 shows the comparison results of the three models. It is showing the accuracy hit ratios for classifying the direction of each market index into "UP" and "DWON" and it is showing also the Mean Absolute Error and Root Mean Square Error for the three models.

	Market	Accuracy hit Ratio	Mean Absolute Error	Root Mean Squared Error
SVM	ADX	82.11%	0.2639	0.3403
	BHM	76.72%	0.2771	0.3591
	DFM	81.16%	0.2648	0.3415
	DSM	81.25%	0.265	0.3419
	KSE	80.62%	0.2666	0.3442
	MSM	83.42%	0.261	0.3359
	TADAWUL	82.13%	0.1664	0.2872
	MAX	**83.42%**	**27.71%**	**35.91%**
	MIN	**76.72%**	**16.64%**	**28.72%**
Logit	ADX	84.19%	0.1516	0.2769
	BHM	80.67%	0.183	0.3032
	DFM	85.24%	0.1454	0.2709
	DSM	83.15%	0.1581	0.282
	KSE	85.24%	0.1361	0.2619
	MSM	87.47%	0.1325	0.2552
	TADAWUL	82.13%	0.1664	0.2872
	MAX	**87.47%**	**18.30%**	**30.32%**
	MIN	**80.67%**	**13.25%**	**25.52%**
Random Forest	ADX	81.32%	0.1663	0.3041
	BHM	74.60%	0.2075	0.3415
	DFM	78.66%	0.1749	0.3125
	DSM	80.34%	0.1675	0.3035
	KSE	78.41%	0.1803	0.3189
	MSM	82.96%	0.1523	0.286
	TADAWUL	81.22%	0.1607	0.2975
	MAX	**82.96%**	**20.75%**	**34.15%**
	MIN	**74.60%**	**15.23%**	**28.60%**

Appendix 11

Appendix 11 provides the actual output of SVM evaluation spooled from Weka Software for both types of experiments for each market

ADX

```
=== Run information ===

Scheme:weka.classifiers.functions.SMO -C 1.0 -L 0.0010 -P 1.0E-12 -N 0 -V -1 -W 1 -K
"weka.classifiers.functions.supportVector.PolyKernel -C 250007 -E 1.0"
Relation:     ADX_31122009_31122013
Instances:    1012
Attributes:   11
              StochasticK
              StochasticD
              SlowD
              ROC
              WilliamsR
              Disparity5
              Disparity10
              OSCP
              20-day CCI
              14-day RSI
              Classifier
Test mode:10-fold cross-validation

=== Classifier model (full training set) ===

SMO

Kernel used:
  Linear Kernel: K(x,y) = <x,y>

Classifier for classes: DOWN, UP

BinarySMO

Machine linear: showing attribute weights, not support vectors.

        5.6444 * (normalized) StochasticK
 +     -4.8096 * (normalized) StochasticD
 +     -5.923  * (normalized) SlowD
 +      0.9175 * (normalized) ROC
 +      5.6444 * (normalized) WilliamsR
 +      0.4441 * (normalized) Disparity5
 +      0.3915 * (normalized) Disparity10
 +      0.4348 * (normalized) OSCP
 +      2.4413 * (normalized) 20-day CCI
 +      1.1645 * (normalized) 14-day RSI
 -      3.3378

Number of kernel evaluations: 72473 (73.626% cached)

Classifier for classes: DOWN, FALSE

BinarySMO

Machine linear: showing attribute weights, not support vectors.

        0.0008 * (normalized) StochasticK
 +     -0.0085 * (normalized) StochasticD
 +      0.0041 * (normalized) SlowD
 +      0.0012 * (normalized) ROC
 +      0.0008 * (normalized) WilliamsR
 +      0.0049 * (normalized) Disparity5
 +     -0.0107 * (normalized) Disparity10
 +     -0.0057 * (normalized) OSCP
 +       0.0085 * (normalized) 20-day CCI
 +       0.0001 * (normalized) 14-day RSI
 -       0.9954

Number of kernel evaluations: 23933 (72.567% cached)

Classifier for classes: UP, FALSE

BinarySMO

Machine linear: showing attribute weights, not support vectors.

        -0.0056 * (normalized) StochasticK
 +       0.0245 * (normalized) StochasticD
 +      -0.0152 * (normalized) SlowD
 +      -0.0016 * (normalized) ROC
 +      -0.0056 * (normalized) WilliamsR
 +      -0.0065 * (normalized) Disparity5
 +      -0.0024 * (normalized) Disparity10
 +      -0.0134 * (normalized) OSCP
 +      -0.0011 * (normalized) 20-day CCI
 +       0.0014 * (normalized) 14-day RSI
 -       0.9845

Number of kernel evaluations: 10852 (65.894% cached)



Time taken to build model: 0.23 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         831               82.1146 %
Incorrectly Classified Instances       181               17.8854 %
Kappa statistic                          0.6426
Mean absolute error                      0.2639
Root mean squared error                  0.3403
Relative absolute error                 78.1354 %
Root relative squared error             82.8423 %
Total Number of Instances             1012

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.794     0.146      0.822     0.794     0.807      0.824    DOWN
                 0.859     0.213      0.821     0.859     0.839      0.823    UP
                 0         0          0         0         0          0.5      FALSE
Weighted Avg.    0.821     0.18       0.814     0.821     0.817      0.82

=== Confusion Matrix ===

   a   b   c   <-- classified as
 369  96   0 |   a = DOWN
  76 462   0 |   b = UP
   4   5   0 |   c = FALSE
```

```
=== Run information ===

Scheme:weka.classifiers.functions.SMO -C 1.0 -L 0.0010 -P 1.0E-12 -N 0 -V -1 -W 1 -K
"weka.classifiers.functions.supportVector.PolyKernel -C 250007 -E 1.0"
Relation:     ADX_31122009_31122013
Instances:    1012
Attributes:   11
              StochasticK
              StochasticD
              SlowD
```

```
            ROC
            WilliamsR
            Disparity5
            Disparity10
            OSCP
            20-day CCI
            14-day RSI
            Classifier
Test mode:split 75.0% train, remainder test

=== Classifier model (full training set) ===

SMO

Kernel used:
  Linear Kernel: K(x,y) = <x,y>

Classifier for classes: DOWN, UP

BinarySMO

Machine linear: showing attribute weights, not support vectors.

        5.6444 * (normalized) StochasticK
 +     -4.8096 * (normalized) StochasticD
 +     -5.923  * (normalized) SlowD
 +      0.9175 * (normalized) ROC
 +      5.6444 * (normalized) WilliamsR
 +      0.4441 * (normalized) Disparity5
 +      0.3915 * (normalized) Disparity10
 +      0.4348 * (normalized) OSCP
 +      2.4413 * (normalized) 20-day CCI
 +      1.1645 * (normalized) 14-day RSI
 -      3.3378

Number of kernel evaluations: 72473 (73.626% cached)

Classifier for classes: DOWN, FALSE

BinarySMO

Machine linear: showing attribute weights, not support vectors.

        0.0008 * (normalized) StochasticK
 +     -0.0085 * (normalized) StochasticD
 +      0.0041 * (normalized) SlowD
 +      0.0012 * (normalized) ROC
 +      0.0008 * (normalized) WilliamsR
 +      0.0049 * (normalized) Disparity5
 +     -0.0107 * (normalized) Disparity10
 +     -0.0057 * (normalized) OSCP
 +      0.0085 * (normalized) 20-day CCI
 +      0.0001 * (normalized) 14-day RSI
 -      0.9954

Number of kernel evaluations: 23933 (72.567% cached)

Classifier for classes: UP, FALSE

BinarySMO

Machine linear: showing attribute weights, not support vectors.

       -0.0056 * (normalized) StochasticK
 +      0.0245 * (normalized) StochasticD
 +     -0.0152 * (normalized) SlowD
 +     -0.0016 * (normalized) ROC
 +     -0.0056 * (normalized) WilliamsR
 +     -0.0065 * (normalized) Disparity5
 +     -0.0024 * (normalized) Disparity10
 +      -0.0134 * (normalized) OSCP
 +      -0.0011 * (normalized) 20-day CCI
 +       0.0014 * (normalized) 14-day RSI
 -       0.9845

Number of kernel evaluations: 10852 (65.894% cached)



Time taken to build model: 0.05 seconds

=== Evaluation on test split ===
=== Summary ===

Correctly Classified Instances         205               81.0277 %
Incorrectly Classified Instances        48               18.9723 %
Kappa statistic                          0.6233
Mean absolute error                      0.267
Root mean squared error                  0.3448
Relative absolute error                 78.8103 %
Root relative squared error             83.5899 %
Total Number of Instances              253

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.788     0.17       0.802     0.788     0.795      0.809    DOWN
                 0.848     0.207      0.818     0.848     0.833      0.821    UP
                 0         0          0         0         0          0.5      FALSE
Weighted Avg.    0.81      0.187      0.8       0.81      0.805      0.812

=== Confusion Matrix ===

   a   b   c   <-- classified as
  93  25   0 |   a = DOWN
  20 112   0 |   b = UP
   3   0   0 |   c = FALSE
```

BHM

```
=== Run information ===

Scheme:weka.classifiers.functions.SMO -C 1.0 -L 0.0010 -P 1.0E-12 -N 0 -V -1 -W 1 -K
"weka.classifiers.functions.supportVector.PolyKernel -C 250007 -E 1.0"
Relation:     BHM_31122009_31122013
Instances:    988
Attributes:   11
              StochasticK
              StochasticD
              SlowD
              ROC
              WilliamsR
              Disparity5
              Disparity10
              OSCP
              20-day CCI
              14-day RSI
              Classifier
Test mode:10-fold cross-validation

=== Classifier model (full training set) ===

SMO

Kernel used:
  Linear Kernel: K(x,y) = <x,y>

Classifier for classes: UP, DOWN
```

```
BinarySMO

Machine linear: showing attribute weights, not support vectors.

       -5.1875 * (normalized) StochasticK
 +      2.3273 * (normalized) StochasticD
 +      3.5658 * (normalized) SlowD
 +      0.4173 * (normalized) ROC
 +     -5.1875 * (normalized) WilliamsR
 +     -6.1262 * (normalized) Disparity5
 +     -2.5035 * (normalized) Disparity10
 +     -1.9568 * (normalized) OSCP
 +      0.9107 * (normalized) 20-day CCI
 +     -1.6003 * (normalized) 14-day RSI
 +      8.0091

Number of kernel evaluations: 34106 (63.129% cached)

Classifier for classes: UP, FALSE

BinarySMO

Machine linear: showing attribute weights, not support vectors.

       -0.0105 * (normalized) StochasticK
 +      0.0239 * (normalized) StochasticD
 +     -0.0188 * (normalized) SlowD
 +     -0.0014 * (normalized) ROC
 +     -0.0105 * (normalized) WilliamsR
 +     -0.0051 * (normalized) Disparity5
 +     -0.0008 * (normalized) Disparity10
 +     -0.0056 * (normalized) OSCP
 +      0.0036 * (normalized) 20-day CCI
 +      0.0016 * (normalized) 14-day RSI
 -      0.9895

Number of kernel evaluations: 12087 (70.966% cached)

Classifier for classes: DOWN, FALSE

BinarySMO

Machine linear: showing attribute weights, not support vectors.

        0.0029 * (normalized) StochasticK
 +      0.0109 * (normalized) StochasticD
 +     -0.0127 * (normalized) SlowD
 +     -0.0006 * (normalized) ROC
 +      0.0029 * (normalized) WilliamsR
 +     -0.0019 * (normalized) Disparity5
 +      0.0015 * (normalized) Disparity10
 +     -0.0028 * (normalized) OSCP
 +     -0.0016 * (normalized) 20-day CCI
 +      0      * (normalized) 14-day RSI
 -      0.9997

Number of kernel evaluations: 30245 (67.563% cached)



Time taken to build model: 0.04 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         758               76.7206 %
Incorrectly Classified Instances       230               23.2794 %
Kappa statistic                          0.5392
Mean absolute error                      0.2771
Root mean squared error                  0.3591
```

```
Relative absolute error                80.9096 %
Root relative squared error            86.8215 %
Total Number of Instances              988

=== Detailed Accuracy By Class ===

              TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                0.736     0.189      0.779     0.736     0.757      0.773    UP
                0.818     0.273      0.758     0.818     0.787      0.772    DOWN
                0         0          0         0         0          0.5      FALSE
Weighted Avg.    0.767     0.229      0.757     0.767     0.761      0.769

=== Confusion Matrix ===

   a   b   c   <-- classified as
 345 124   0 |   a = UP
  92 413   0 |   b = DOWN
   6   8   0 |   c = FALSE
```

```
=== Run information ===

Scheme:weka.classifiers.functions.SMO -C 1.0 -L 0.0010 -P 1.0E-12 -N 0 -V -1 -W 1 -K
"weka.classifiers.functions.supportVector.PolyKernel -C 250007 -E 1.0"
Relation:     BHM_31122009_31122013
Instances:    988
Attributes:   11
              StochasticK
              StochasticD
              SlowD
              ROC
              WilliamsR
              Disparity5
              Disparity10
              OSCP
              20-day CCI
              14-day RSI
              Classifier
Test mode:split 75.0% train, remainder test

=== Classifier model (full training set) ===

SMO

Kernel used:
  Linear Kernel: K(x,y) = <x,y>

Classifier for classes: UP, DOWN

BinarySMO

Machine linear: showing attribute weights, not support vectors.

        -5.1875 * (normalized) StochasticK
 +       2.3273 * (normalized) StochasticD
 +       3.5658 * (normalized) SlowD
 +       0.4173 * (normalized) ROC
 +      -5.1875 * (normalized) WilliamsR
 +      -6.1262 * (normalized) Disparity5
 +      -2.5035 * (normalized) Disparity10
 +      -1.9568 * (normalized) OSCP
 +       0.9107 * (normalized) 20-day CCI
 +      -1.6003 * (normalized) 14-day RSI
 +       8.0091

Number of kernel evaluations: 34106 (63.129% cached)

Classifier for classes: UP, FALSE
```

```
BinarySMO

Machine linear: showing attribute weights, not support vectors.

       -0.0105 * (normalized) StochasticK
 +      0.0239 * (normalized) StochasticD
 +     -0.0188 * (normalized) SlowD
 +     -0.0014 * (normalized) ROC
 +     -0.0105 * (normalized) WilliamsR
 +     -0.0051 * (normalized) Disparity5
 +     -0.0008 * (normalized) Disparity10
 +     -0.0056 * (normalized) OSCP
 +      0.0036 * (normalized) 20-day CCI
 +      0.0016 * (normalized) 14-day RSI
 -      0.9895

Number of kernel evaluations: 12087 (70.966% cached)

Classifier for classes: DOWN, FALSE

BinarySMO

Machine linear: showing attribute weights, not support vectors.

        0.0029 * (normalized) StochasticK
 +      0.0109 * (normalized) StochasticD
 +     -0.0127 * (normalized) SlowD
 +     -0.0006 * (normalized) ROC
 +      0.0029 * (normalized) WilliamsR
 +     -0.0019 * (normalized) Disparity5
 +      0.0015 * (normalized) Disparity10
 +     -0.0028 * (normalized) OSCP
 +     -0.0016 * (normalized) 20-day CCI
 +      0      * (normalized) 14-day RSI
 -      0.9997

Number of kernel evaluations: 30245 (67.563% cached)



Time taken to build model: 0.04 seconds

=== Evaluation on test split ===
=== Summary ===

Correctly Classified Instances         178               72.0648 %
Incorrectly Classified Instances        69               27.9352 %
Kappa statistic                          0.451
Mean absolute error                      0.2888
Root mean squared error                  0.375
Relative absolute error                 84.019  %
Root relative squared error             90.2141 %
Total Number of Instances              247

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.728     0.263      0.703     0.728     0.716      0.732    UP
                 0.742     0.286      0.736     0.742     0.739      0.728    DOWN
                 0         0          0         0         0          0.5      FALSE
Weighted Avg.    0.721     0.27       0.706     0.721     0.713      0.726

=== Confusion Matrix ===

  a  b  c   <-- classified as
 83 31  0 |  a = UP
 33 95  0 |  b = DOWN
  2  3  0 |  c = FALSE
```

DFM

```
=== Run information ===

Scheme:weka.classifiers.functions.SMO -C 1.0 -L 0.0010 -P 1.0E-12 -N 0 -V -1 -W 1 -K
"weka.classifiers.functions.supportVector.PolyKernel -C 250007 -E 1.0"
Relation:     DFM_31122009_31122013
Instances:    1003
Attributes:   11
              StochasticK
              StochasticD
              SlowD
              ROC
              WilliamsR
              Disparity5
              Disparity10
              OSCP
              20-day CCI
              14-day RSI
              Classifier
Test mode:10-fold cross-validation

=== Classifier model (full training set) ===

SMO

Kernel used:
  Linear Kernel: K(x,y) = <x,y>

Classifier for classes: DOWN, UP

BinarySMO

Machine linear: showing attribute weights, not support vectors.

         4.75   * (normalized) StochasticK
 +      -5.4106 * (normalized) StochasticD
 +      -3.8161 * (normalized) SlowD
 +       0.6433 * (normalized) ROC
 +       4.75   * (normalized) WilliamsR
 +       2.4948 * (normalized) Disparity5
 +       1.0968 * (normalized) Disparity10
 +       1.5338 * (normalized) OSCP
 +      -0.673  * (normalized) 20-day CCI
 +       1.9977 * (normalized) 14-day RSI
 -       4.0442

Number of kernel evaluations: 67904 (68.522% cached)

Classifier for classes: DOWN, FALSE

BinarySMO

Machine linear: showing attribute weights, not support vectors.

        -0.0004 * (normalized) StochasticK
 +      -0.0047 * (normalized) StochasticD
 +       0.007  * (normalized) SlowD
 +      -0.0044 * (normalized) ROC
 +      -0.0004 * (normalized) WilliamsR
 +       0.0051 * (normalized) Disparity5
 +       0.0055 * (normalized) Disparity10
 +       0.0027 * (normalized) OSCP
 +      -0.0075 * (normalized) 20-day CCI
 +       0.0011 * (normalized) 14-day RSI
 -       1.0033

Number of kernel evaluations: 22364 (74.783% cached)

Classifier for classes: UP, FALSE
```

```
BinarySMO

Machine linear: showing attribute weights, not support vectors.

       -0.0071 * (normalized) StochasticK
 +      0.0316 * (normalized) StochasticD
 +     -0.0076 * (normalized) SlowD
 +     -0.0089 * (normalized) ROC
 +     -0.0071 * (normalized) WilliamsR
 +      0.0041 * (normalized) Disparity5
 +      0.0047 * (normalized) Disparity10
 +      0.003  * (normalized) OSCP
 +     -0.019  * (normalized) 20-day CCI
 +      0.0006 * (normalized) 14-day RSI
 -      1.0001

Number of kernel evaluations: 12983 (75.943% cached)



Time taken to build model: 0.04 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         814               81.1565 %
Incorrectly Classified Instances       189               18.8435 %
Kappa statistic                          0.6241
Mean absolute error                      0.2648
Root mean squared error                  0.3415
Relative absolute error                 78.9329 %
Root relative squared error             83.4295 %
Total Number of Instances             1003

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.815     0.188      0.803     0.815     0.809      0.814    DOWN
                 0.813     0.188      0.82      0.813     0.816      0.813    UP
                 0         0          0         0         0          0.5      FALSE
Weighted Avg.    0.812     0.187      0.809     0.812     0.81       0.812

=== Confusion Matrix ===

   a   b   c   <-- classified as
 396  90   0 |   a = DOWN
  96 418   0 |   b = UP
   1   2   0 |   c = FALSE
```

```
=== Run information ===

Scheme:weka.classifiers.functions.SMO -C 1.0 -L 0.0010 -P 1.0E-12 -N 0 -V -1 -W 1 -K
"weka.classifiers.functions.supportVector.PolyKernel -C 250007 -E 1.0"
Relation:     DFM_31122009_31122013
Instances:    1003
Attributes:   11
              StochasticK
              StochasticD
              SlowD
              ROC
              WilliamsR
              Disparity5
              Disparity10
              OSCP
              20-day CCI
              14-day RSI
```

```
            Classifier
Test mode:split 75.0% train, remainder test

=== Classifier model (full training set) ===

SMO

Kernel used:
  Linear Kernel: K(x,y) = <x,y>

Classifier for classes: DOWN, UP

BinarySMO

Machine linear: showing attribute weights, not support vectors.

         4.75   * (normalized) StochasticK
 +      -5.4106 * (normalized) StochasticD
 +      -3.8161 * (normalized) SlowD
 +       0.6433 * (normalized) ROC
 +       4.75   * (normalized) WilliamsR
 +       2.4948 * (normalized) Disparity5
 +       1.0968 * (normalized) Disparity10
 +       1.5338 * (normalized) OSCP
 +      -0.673  * (normalized) 20-day CCI
 +       1.9977 * (normalized) 14-day RSI
 -       4.0442

Number of kernel evaluations: 67904 (68.522% cached)

Classifier for classes: DOWN, FALSE

BinarySMO

Machine linear: showing attribute weights, not support vectors.

        -0.0004 * (normalized) StochasticK
 +      -0.0047 * (normalized) StochasticD
 +       0.007  * (normalized) SlowD
 +      -0.0044 * (normalized) ROC
 +      -0.0004 * (normalized) WilliamsR
 +       0.0051 * (normalized) Disparity5
 +       0.0055 * (normalized) Disparity10
 +       0.0027 * (normalized) OSCP
 +      -0.0075 * (normalized) 20-day CCI
 +       0.0011 * (normalized) 14-day RSI
 -       1.0033

Number of kernel evaluations: 22364 (74.783% cached)

Classifier for classes: UP, FALSE

BinarySMO

Machine linear: showing attribute weights, not support vectors.

        -0.0071 * (normalized) StochasticK
 +       0.0316 * (normalized) StochasticD
 +      -0.0076 * (normalized) SlowD
 +      -0.0089 * (normalized) ROC
 +      -0.0071 * (normalized) WilliamsR
 +       0.0041 * (normalized) Disparity5
 +       0.0047 * (normalized) Disparity10
 +       0.003  * (normalized) OSCP
 +      -0.019  * (normalized) 20-day CCI
 +       0.0006 * (normalized) 14-day RSI
 -       1.0001

Number of kernel evaluations: 12983 (75.943% cached)
```

```
Time taken to build model: 0.05 seconds

=== Evaluation on test split ===
=== Summary ===

Correctly Classified Instances         206               82.0717 %
Incorrectly Classified Instances        45               17.9283 %
Kappa statistic                          0.6428
Mean absolute error                      0.2629
Root mean squared error                  0.3388
Relative absolute error                 78.2867 %
Root relative squared error             82.6411 %
Total Number of Instances              251

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.823     0.173      0.823     0.823     0.823      0.825    DOWN
                 0.825     0.184      0.819     0.825     0.822      0.821    UP
                 0         0          0         0         0          0.5      FALSE
Weighted Avg.    0.821     0.178      0.817     0.821     0.819      0.821

=== Confusion Matrix ===

   a   b   c   <-- classified as
 102  22   0 |   a = DOWN
  22 104   0 |   b = UP
   0   1   0 |   c = FALSE
```

DSM

```
=== Run information ===

Scheme:weka.classifiers.functions.SMO -C 1.0 -L 0.0010 -P 1.0E-12 -N 0 -V -1 -W 1 -K
"weka.classifiers.functions.supportVector.PolyKernel -C 250007 -E 1.0"
Relation:     DSM_31122009_31122013
Instances:    1531
Attributes:   11
              StochasticK
              StochasticD
              SlowD
              ROC
              WilliamsR
              Disparity5
              Disparity10
              OSCP
              20-day CCI
              14-day RSI
              Classifier
Test mode:10-fold cross-validation

=== Classifier model (full training set) ===

SMO

Kernel used:
  Linear Kernel: K(x,y) = <x,y>

Classifier for classes: DOWN, UP

BinarySMO

Machine linear: showing attribute weights, not support vectors.

         5.8765 * (normalized) StochasticK
 +      -6.1534 * (normalized) StochasticD
 +      -5.2764 * (normalized) SlowD
 +       0.9225 * (normalized) ROC
 +       5.8765 * (normalized) WilliamsR
 +       0.1802 * (normalized) Disparity5
 +       0.4583 * (normalized) Disparity10
 +       0.022  * (normalized) OSCP
 +       2.0584 * (normalized) 20-day CCI
 +       1.7482 * (normalized) 14-day RSI
 -       2.9221

Number of kernel evaluations: 136400 (70.789% cached)

Classifier for classes: DOWN, FALSE

BinarySMO

Machine linear: showing attribute weights, not support vectors.

         0.0037 * (normalized) StochasticK
 +      -0.0242 * (normalized) StochasticD
 +       0.0159 * (normalized) SlowD
 +       0.0004 * (normalized) ROC
 +       0.0037 * (normalized) WilliamsR
 +       0.0015 * (normalized) Disparity5
 +       0.0025 * (normalized) Disparity10
 +      -0.0029 * (normalized) OSCP
 +       0.0064 * (normalized) 20-day CCI
 +      -0.0018 * (normalized) 14-day RSI
 -       1.0048

Number of kernel evaluations: 31738 (69.848% cached)

Classifier for classes: UP, FALSE

BinarySMO

Machine linear: showing attribute weights, not support vectors.

       -0.0004 * (normalized) StochasticK
 +       0      * (normalized) StochasticD
 +      -0.0013 * (normalized) SlowD
 +      -0.0014 * (normalized) ROC
 +      -0.0004 * (normalized) WilliamsR
 +      -0.0002 * (normalized) Disparity5
 +      -0.0076 * (normalized) Disparity10
 +      -0.0067 * (normalized) OSCP
 +       0.005  * (normalized) 20-day CCI
 +       0.0018 * (normalized) 14-day RSI
 -       0.9945

Number of kernel evaluations: 32513 (65.22% cached)



Time taken to build model: 0.28 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        1244               81.2541 %
Incorrectly Classified Instances       287               18.7459 %
Kappa statistic                          0.6166
Mean absolute error                      0.265
Root mean squared error                  0.3419
Relative absolute error                 79.6686 %
Root relative squared error             83.8598 %
Total Number of Instances             1531

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
```

```
                0.724    0.111     0.837    0.724    0.776     0.807    DOWN
                0.89     0.282     0.797    0.89     0.841     0.804    UP
                0        0         0        0        0         0.5      FALSE
Weighted Avg.   0.813    0.205     0.811    0.813    0.808     0.804

=== Confusion Matrix ===

   a   b   c   <-- classified as
 488 186   0 |   a = DOWN
  93 756   0 |   b = UP
   2   6   0 |   c = FALSE
```

```
=== Run information ===

Scheme:weka.classifiers.functions.SMO -C 1.0 -L 0.0010 -P 1.0E-12 -N 0 -V -1 -W 1 -K
"weka.classifiers.functions.supportVector.PolyKernel -C 250007 -E 1.0"
Relation:     DSM_31122009_31122013
Instances:    1531
Attributes:   11
              StochasticK
              StochasticD
              SlowD
              ROC
              WilliamsR
              Disparity5
              Disparity10
              OSCP
              20-day CCI
              14-day RSI
              Classifier
Test mode:split 75.0% train, remainder test

=== Classifier model (full training set) ===

SMO

Kernel used:
  Linear Kernel: K(x,y) = <x,y>

Classifier for classes: DOWN, UP

BinarySMO

Machine linear: showing attribute weights, not support vectors.

         5.8765 * (normalized) StochasticK
 +      -6.1534 * (normalized) StochasticD
 +      -5.2764 * (normalized) SlowD
 +       0.9225 * (normalized) ROC
 +       5.8765 * (normalized) WilliamsR
 +       0.1802 * (normalized) Disparity5
 +       0.4583 * (normalized) Disparity10
 +       0.022  * (normalized) OSCP
 +       2.0584 * (normalized) 20-day CCI
 +       1.7482 * (normalized) 14-day RSI
 -       2.9221

Number of kernel evaluations: 136400 (70.789% cached)

Classifier for classes: DOWN, FALSE

BinarySMO

Machine linear: showing attribute weights, not support vectors.

         0.0037 * (normalized) StochasticK
 +      -0.0242 * (normalized) StochasticD
 +      0.0159 * (normalized) SlowD
 +      0.0004 * (normalized) ROC
 +      0.0037 * (normalized) WilliamsR
 +      0.0015 * (normalized) Disparity5
 +      0.0025 * (normalized) Disparity10
 +     -0.0029 * (normalized) OSCP
 +      0.0064 * (normalized) 20-day CCI
 +     -0.0018 * (normalized) 14-day RSI
 -      1.0048

Number of kernel evaluations: 31738 (69.848% cached)

Classifier for classes: UP, FALSE

BinarySMO

Machine linear: showing attribute weights, not support vectors.

       -0.0004 * (normalized) StochasticK
 +      0      * (normalized) StochasticD
 +     -0.0013 * (normalized) SlowD
 +     -0.0014 * (normalized) ROC
 +     -0.0004 * (normalized) WilliamsR
 +     -0.0002 * (normalized) Disparity5
 +     -0.0076 * (normalized) Disparity10
 +     -0.0067 * (normalized) OSCP
 +      0.005  * (normalized) 20-day CCI
 +      0.0018 * (normalized) 14-day RSI
 -      0.9945

Number of kernel evaluations: 32513 (65.22% cached)



Time taken to build model: 0.09 seconds

=== Evaluation on test split ===
=== Summary ===

Correctly Classified Instances         311               81.201  %
Incorrectly Classified Instances        72               18.799  %
Kappa statistic                          0.617
Mean absolute error                      0.2652
Root mean squared error                  0.3421
Relative absolute error                 79.5772 %
Root relative squared error             83.7231 %
Total Number of Instances              383

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.715     0.104      0.848     0.715     0.776      0.805    DOWN
                 0.9       0.287      0.79      0.9       0.841      0.806    UP
                 0         0          0         0         0          0.5      FALSE
Weighted Avg.    0.812     0.204      0.812     0.812     0.808      0.804

=== Confusion Matrix ===

   a   b   c   <-- classified as
 123  49   0 |   a = DOWN
  21 188   0 |   b = UP
   1   1   0 |   c = FALSE
```

KSE

```
=== Run information ===

Scheme:weka.classifiers.functions.SMO -C 1.0 -L 0.0010 -P 1.0E-12 -N 0 -V -1 -W 1 -K
"weka.classifiers.functions.supportVector.PolyKernel -C 250007 -E 1.0"
Relation:     KSE_31122009_31122013
Instances:    996
Attributes:   11
              StochasticK
              StochasticD
              SlowD
              ROC
              WilliamsR
              Disparity5
              Disparity10
              OSCP
              20-day CCI
              14-day RSI
              Classifier
Test mode:10-fold cross-validation

=== Classifier model (full training set) ===

SMO

Kernel used:
  Linear Kernel: K(x,y) = <x,y>

Classifier for classes: UP, DOWN

BinarySMO

Machine linear: showing attribute weights, not support vectors.

        -4.6135 * (normalized) StochasticK
 +       5.2953 * (normalized) StochasticD
 +       4.0122 * (normalized) SlowD
 +      -0.2734 * (normalized) ROC
 +      -4.6135 * (normalized) WilliamsR
 +      -3.2042 * (normalized) Disparity5
 +      -0.9058 * (normalized) Disparity10
 +      -2.0076 * (normalized) OSCP
 +      -0.7672 * (normalized) 20-day CCI
 +      -1.6893 * (normalized) 14-day RSI
 +       4.6624

Number of kernel evaluations: 78616 (72.812% cached)

Classifier for classes: UP, FALSE

BinarySMO

Machine linear: showing attribute weights, not support vectors.

        -0.009  * (normalized) StochasticK
 +       0.0301 * (normalized) StochasticD
 +      -0.0126 * (normalized) SlowD
 +       0.0001 * (normalized) ROC
 +      -0.009  * (normalized) WilliamsR
 +      -0.0072 * (normalized) Disparity5
 +      -0.0019 * (normalized) Disparity10
 +      -0.0025 * (normalized) OSCP
 +       0.0066 * (normalized) 20-day CCI
 +      -0.0016 * (normalized) 14-day RSI
 -       0.9972

Number of kernel evaluations: 12104 (77.007% cached)

Classifier for classes: DOWN, FALSE

BinarySMO
```

```
Machine linear: showing attribute weights, not support vectors.

       0.0027 * (normalized) StochasticK
 +     -0.019  * (normalized) StochasticD
 +      0.0151 * (normalized) SlowD
 +      0.0024 * (normalized) ROC
 +      0.0027 * (normalized) WilliamsR
 +      0.0018 * (normalized) Disparity5
 +      0.0009 * (normalized) Disparity10
 +      0.0044 * (normalized) OSCP
 +      0.0037 * (normalized) 20-day CCI
 +     -0.0044 * (normalized) 14-day RSI
 -      1.0059

Number of kernel evaluations: 15652 (81.113% cached)



Time taken to build model: 0.05 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         803               80.6225 %
Incorrectly Classified Instances       193               19.3775 %
Kappa statistic                          0.6112
Mean absolute error                      0.2666
Root mean squared error                  0.3442
Relative absolute error                 79.3666 %
Root relative squared error             84.0287 %
Total Number of Instances              996

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.851     0.234      0.806     0.851     0.828      0.808    UP
                 0.765     0.156      0.807     0.765     0.785      0.804    DOWN
                 0         0          0         0         0          0.5      FALSE
Weighted Avg.    0.806     0.197      0.801     0.806     0.803      0.805

=== Confusion Matrix ===

   a   b   c   <-- classified as
 452  79   0 |   a = UP
 108 351   0 |   b = DOWN
   1   5   0 |   c = FALSE
```

```
=== Run information ===

Scheme:weka.classifiers.functions.SMO -C 1.0 -L 0.0010 -P 1.0E-12 -N 0 -V -1 -W 1 -K
"weka.classifiers.functions.supportVector.PolyKernel -C 250007 -E 1.0"
Relation:     KSE_31122009_31122013
Instances:    996
Attributes:   11
              StochasticK
              StochasticD
              SlowD
              ROC
              WilliamsR
              Disparity5
              Disparity10
              OSCP
              20-day CCI
              14-day RSI
              Classifier
Test mode:split 75.0% train, remainder test
```

```
=== Classifier model (full training set) ===

SMO

Kernel used:
  Linear Kernel: K(x,y) = <x,y>

Classifier for classes: UP, DOWN

BinarySMO

Machine linear: showing attribute weights, not support vectors.

       -4.6135 * (normalized) StochasticK
 +      5.2953 * (normalized) StochasticD
 +      4.0122 * (normalized) SlowD
 +     -0.2734 * (normalized) ROC
 +     -4.6135 * (normalized) WilliamsR
 +     -3.2042 * (normalized) Disparity5
 +     -0.9058 * (normalized) Disparity10
 +     -2.0076 * (normalized) OSCP
 +     -0.7672 * (normalized) 20-day CCI
 +     -1.6893 * (normalized) 14-day RSI
 +      4.6624

Number of kernel evaluations: 78616 (72.812% cached)

Classifier for classes: UP, FALSE

BinarySMO

Machine linear: showing attribute weights, not support vectors.

       -0.009  * (normalized) StochasticK
 +      0.0301 * (normalized) StochasticD
 +     -0.0126 * (normalized) SlowD
 +      0.0001 * (normalized) ROC
 +     -0.009  * (normalized) WilliamsR
 +     -0.0072 * (normalized) Disparity5
 +     -0.0019 * (normalized) Disparity10
 +     -0.0025 * (normalized) OSCP
 +      0.0066 * (normalized) 20-day CCI
 +     -0.0016 * (normalized) 14-day RSI
 -      0.9972

Number of kernel evaluations: 12104 (77.007% cached)

Classifier for classes: DOWN, FALSE

BinarySMO

Machine linear: showing attribute weights, not support vectors.

        0.0027 * (normalized) StochasticK
 +     -0.019  * (normalized) StochasticD
 +      0.0151 * (normalized) SlowD
 +      0.0024 * (normalized) ROC
 +      0.0027 * (normalized) WilliamsR
 +      0.0018 * (normalized) Disparity5
 +      0.0009 * (normalized) Disparity10
 +      0.0044 * (normalized) OSCP
 +      0.0037 * (normalized) 20-day CCI
 +     -0.0044 * (normalized) 14-day RSI
 -      1.0059

Number of kernel evaluations: 15652 (81.113% cached)
```

```
Time taken to build model: 0.13 seconds

=== Evaluation on test split ===
=== Summary ===

Correctly Classified Instances         199               79.9197 %
Incorrectly Classified Instances        50               20.0803 %
Kappa statistic                          0.5958
Mean absolute error                      0.2668
Root mean squared error                  0.3445
Relative absolute error                 79.6747 %
Root relative squared error             84.5228 %
Total Number of Instances              249

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.833     0.239      0.797     0.833     0.815      0.797    UP
                 0.761     0.167      0.802     0.761     0.781      0.797    DOWN
                 0         0          0         0         0          ?        FALSE
Weighted Avg.    0.799     0.205      0.799     0.799     0.799      0.797

=== Confusion Matrix ===

   a   b   c   <-- classified as
 110  22   0 |   a = UP
  28  89   0 |   b = DOWN
   0   0   0 |   c = FALSE
```

MSM

```
=== Run information ===

Scheme:weka.classifiers.functions.SMO -C 1.0 -L 0.0010 -P 1.0E-12 -N 0 -V -1 -W 1 -K
"weka.classifiers.functions.supportVector.PolyKernel -C 250007 -E 1.0"
Relation:     MSM_31122009_31122014
Instances:    1285
Attributes:   11
              StochasticK
              StochasticD
              SlowD
              ROC
              WilliamsR
              Disparity5
              Disparity10
              OSCP
              20-day CCI
              14-day RSI
              Classifier
Test mode:10-fold cross-validation

=== Classifier model (full training set) ===

SMO

Kernel used:
  Linear Kernel: K(x,y) = <x,y>

Classifier for classes: UP, DOWN

BinarySMO

Machine linear: showing attribute weights, not support vectors.

        -5.4018 * (normalized) StochasticK
 +       5.8021 * (normalized) StochasticD
 +       4.76   * (normalized) SlowD
 +       0.0486 * (normalized) ROC
 +      -5.4018 * (normalized) WilliamsR
 +      -3.2633 * (normalized) Disparity5
 +      -1.3975 * (normalized) Disparity10
 +      -2.622  * (normalized) OSCP
 +      -2.0495 * (normalized) 20-day CCI
 +      -1.029  * (normalized) 14-day RSI
 +       6.3258

Number of kernel evaluations: 132059 (73.509% cached)

Classifier for classes: UP, FALSE

BinarySMO

Machine linear: showing attribute weights, not support vectors.

        -0.0007 * (normalized) StochasticK
 +       0.0105 * (normalized) StochasticD
 +      -0.0082 * (normalized) SlowD
 +       0.0062 * (normalized) ROC
 +      -0.0007 * (normalized) WilliamsR
 +       0.0001 * (normalized) Disparity5
 +      -0.0042 * (normalized) Disparity10
 +       0.0022 * (normalized) OSCP
 +      -0.0014 * (normalized) 20-day CCI
 +      -0.0014 * (normalized) 14-day RSI
 -       1.0005

Number of kernel evaluations: 32495 (69.652% cached)

Classifier for classes: DOWN, FALSE

BinarySMO

Machine linear: showing attribute weights, not support vectors.

         0.0045 * (normalized) StochasticK
 +      -0.024  * (normalized) StochasticD
 +       0.0203 * (normalized) SlowD
 +       0.0044 * (normalized) ROC
 +       0.0045 * (normalized) WilliamsR
 +       0.0098 * (normalized) Disparity5
 +       0.0009 * (normalized) Disparity10
 +       0.0131 * (normalized) OSCP
 +      -0.003  * (normalized) 20-day CCI
 +      -0.0025 * (normalized) 14-day RSI
 -       1.0151

Number of kernel evaluations: 16380 (74.585% cached)



Time taken to build model: 0.07 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        1072               83.4241 %
Incorrectly Classified Instances       213               16.5759 %
Kappa statistic                          0.6638
Mean absolute error                      0.261
Root mean squared error                  0.3359
Relative absolute error                 78.1113 %
Root relative squared error             82.2141 %
Total Number of Instances             1285

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.883     0.222      0.834     0.883     0.858      0.831    UP
                 0.788     0.12       0.835     0.788     0.811      0.834    DOWN
                 0        0         0        0        0         0.5     FALSE
Weighted Avg.    0.834    0.176     0.827    0.834    0.83      0.829

=== Confusion Matrix ===

   a   b   c   <-- classified as
 633  84   0 |   a = UP
 118 439   0 |   b = DOWN
   8   3   0 |   c = FALSE
```

```
=== Run information ===

Scheme:weka.classifiers.functions.SMO -C 1.0 -L 0.0010 -P 1.0E-12 -N 0 -V -1 -W 1 -K
"weka.classifiers.functions.supportVector.PolyKernel -C 250007 -E 1.0"
Relation:     MSM_31122009_31122014
Instances:    1285
Attributes:   11
              StochasticK
              StochasticD
              SlowD
              ROC
              WilliamsR
              Disparity5
              Disparity10
              OSCP
              20-day CCI
              14-day RSI
              Classifier
Test mode:split 75.0% train, remainder test

=== Classifier model (full training set) ===

SMO

Kernel used:
  Linear Kernel: K(x,y) = <x,y>

Classifier for classes: UP, DOWN

BinarySMO

Machine linear: showing attribute weights, not support vectors.

        -5.4018 * (normalized) StochasticK
 +       5.8021 * (normalized) StochasticD
 +       4.76   * (normalized) SlowD
 +       0.0486 * (normalized) ROC
 +      -5.4018 * (normalized) WilliamsR
 +      -3.2633 * (normalized) Disparity5
 +      -1.3975 * (normalized) Disparity10
 +      -2.622  * (normalized) OSCP
 +      -2.0495 * (normalized) 20-day CCI
 +      -1.029  * (normalized) 14-day RSI
 +       6.3258

Number of kernel evaluations: 132059 (73.509% cached)

Classifier for classes: UP, FALSE

BinarySMO

Machine linear: showing attribute weights, not support vectors.

        -0.0007 * (normalized) StochasticK
 +       0.0105 * (normalized) StochasticD
 +      -0.0082 * (normalized) SlowD
 +       0.0062 * (normalized) ROC
 +      -0.0007 * (normalized) WilliamsR
 +       0.0001 * (normalized) Disparity5
 +      -0.0042 * (normalized) Disparity10
 +       0.0022 * (normalized) OSCP
 +      -0.0014 * (normalized) 20-day CCI
 +      -0.0014 * (normalized) 14-day RSI
 -       1.0005

Number of kernel evaluations: 32495 (69.652% cached)

Classifier for classes: DOWN, FALSE

BinarySMO

Machine linear: showing attribute weights, not support vectors.

         0.0045 * (normalized) StochasticK
 +      -0.024  * (normalized) StochasticD
 +       0.0203 * (normalized) SlowD
 +       0.0044 * (normalized) ROC
 +       0.0045 * (normalized) WilliamsR
 +       0.0098 * (normalized) Disparity5
 +       0.0009 * (normalized) Disparity10
 +       0.0131 * (normalized) OSCP
 +      -0.003  * (normalized) 20-day CCI
 +      -0.0025 * (normalized) 14-day RSI
 -       1.0151

Number of kernel evaluations: 16380 (74.585% cached)



Time taken to build model: 0.07 seconds

=== Evaluation on test split ===
=== Summary ===

Correctly Classified Instances         256               79.7508 %
Incorrectly Classified Instances        65               20.2492 %
Kappa statistic                          0.5871
Mean absolute error                      0.2693
Root mean squared error                  0.3481
Relative absolute error                 80.7026 %
Root relative squared error             85.3775 %
Total Number of Instances              321

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.847     0.261      0.812     0.847     0.829      0.793    UP
                 0.748     0.156      0.777     0.748     0.762      0.796    DOWN
                 0         0          0         0         0          0.5      FALSE
Weighted Avg.    0.798     0.214      0.789     0.798     0.793      0.792

=== Confusion Matrix ===

   a   b   c   <-- classified as
 155  28   0 |   a = UP
  34 101   0 |   b = DOWN
   2   1   0 |   c = FALSE
```

TADAWUL

```
=== Run information ===

Scheme:weka.classifiers.functions.SMO -C 1.0 -L 0.0010 -P 1.0E-12 -N 0 -V -1 -W 1 -K
"weka.classifiers.functions.supportVector.PolyKernel -C 250007 -E 1.0"
Relation:     Tadauwl_30122009_31122013
Instances:    996
Attributes:   11
              StochasticK
              StochasticD
              SlowD
              ROC
              WilliamsR
              Disparity5
              Disparity10
              OSCP
              20-day CCI
              14-day RSI
              Classifier
Test mode:10-fold cross-validation

=== Classifier model (full training set) ===

SMO

Kernel used:
  Linear Kernel: K(x,y) = <x,y>

Classifier for classes: DOWN, UP

BinarySMO

Machine linear: showing attribute weights, not support vectors.

         5.2576 * (normalized) StochasticK
 +      -2.6748 * (normalized) StochasticD
 +      -4.5851 * (normalized) SlowD
 +       0.3666 * (normalized) ROC
 +       5.2576 * (normalized) WilliamsR
 +       5.6921 * (normalized) Disparity5
 +       2.6435 * (normalized) Disparity10
 +       2.2831 * (normalized) OSCP
 +       1.549  * (normalized) 20-day CCI
 +       1.4533 * (normalized) 14-day RSI
 -      11.3991

Number of kernel evaluations: 36051 (66.46% cached)

Classifier for classes: DOWN, FALSE

BinarySMO

Machine linear: showing attribute weights, not support vectors.

         0.0045 * (normalized) StochasticK
 +      -0.0177 * (normalized) StochasticD
 +       0.0126 * (normalized) SlowD
 +       0.0043 * (normalized) ROC
 +       0.0045 * (normalized) WilliamsR
 +       0.0014 * (normalized) Disparity5
 +      -0.0042 * (normalized) Disparity10
 +       0.0097 * (normalized) OSCP
 +       0.0048 * (normalized) 20-day CCI
 +       0      * (normalized) 14-day RSI
 -       1.0132

Number of kernel evaluations: 563 (62.789% cached)

Classifier for classes: UP, FALSE

BinarySMO
```

```
Machine linear: showing attribute weights, not support vectors.

        0.0003 * (normalized) StochasticK
 +      -0.007  * (normalized) StochasticD
 +       0.0047 * (normalized) SlowD
 +       0.0059 * (normalized) ROC
 +       0.0003 * (normalized) WilliamsR
 +      -0.0048 * (normalized) Disparity5
 +      -0.0045 * (normalized) Disparity10
 +       0.0027 * (normalized) OSCP
 +       0.0082 * (normalized) 20-day CCI
 +      -0.0032 * (normalized) 14-day RSI
 -       1.0023

Number of kernel evaluations: 1744 (66.507% cached)



Time taken to build model: 0.03 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         772               77.51   %
Incorrectly Classified Instances       224               22.49   %
Kappa statistic                          0.5106
Mean absolute error                      0.2724
Root mean squared error                  0.3525
Relative absolute error                 84.4958 %
Root relative squared error             87.8566 %
Total Number of Instances              996

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.573     0.088      0.816     0.573     0.673      0.743    DOWN
                 0.914     0.426      0.759     0.914     0.829      0.744    UP
                 0         0          0         0         0          0.5      FALSE
Weighted Avg.    0.775     0.289      0.781     0.775     0.765      0.743

=== Confusion Matrix ===

   a   b   c   <-- classified as
 231 172   0 |   a = DOWN
  51 541   0 |   b = UP
   1   0   0 |   c = FALSE
```

```
=== Run information ===

Scheme:weka.classifiers.functions.SMO -C 1.0 -L 0.0010 -P 1.0E-12 -N 0 -V -1 -W 1 -K
"weka.classifiers.functions.supportVector.PolyKernel -C 250007 -E 1.0"
Relation:     Tadauwl_30122009_31122013
Instances:    996
Attributes:   11
              StochasticK
              StochasticD
              SlowD
              ROC
              WilliamsR
              Disparity5
              Disparity10
              OSCP
              20-day CCI
              14-day RSI
              Classifier
Test mode:split 75.0% train, remainder test
```

```
=== Classifier model (full training set) ===

SMO

Kernel used:
  Linear Kernel: K(x,y) = <x,y>

Classifier for classes: DOWN, UP

BinarySMO

Machine linear: showing attribute weights, not support vectors.

        5.2576 * (normalized) StochasticK
 +     -2.6748 * (normalized) StochasticD
 +     -4.5851 * (normalized) SlowD
 +      0.3666 * (normalized) ROC
 +      5.2576 * (normalized) WilliamsR
 +      5.6921 * (normalized) Disparity5
 +      2.6435 * (normalized) Disparity10
 +      2.2831 * (normalized) OSCP
 +      1.549  * (normalized) 20-day CCI
 +      1.4533 * (normalized) 14-day RSI
 -     11.3991

Number of kernel evaluations: 36051 (66.46% cached)

Classifier for classes: DOWN, FALSE

BinarySMO

Machine linear: showing attribute weights, not support vectors.

        0.0045 * (normalized) StochasticK
 +     -0.0177 * (normalized) StochasticD
 +      0.0126 * (normalized) SlowD
 +      0.0043 * (normalized) ROC
 +      0.0045 * (normalized) WilliamsR
 +      0.0014 * (normalized) Disparity5
 +     -0.0042 * (normalized) Disparity10
 +      0.0097 * (normalized) OSCP
 +      0.0048 * (normalized) 20-day CCI
 +      0      * (normalized) 14-day RSI
 -      1.0132

Number of kernel evaluations: 563 (62.789% cached)

Classifier for classes: UP, FALSE

BinarySMO

Machine linear: showing attribute weights, not support vectors.

        0.0003 * (normalized) StochasticK
 +     -0.007  * (normalized) StochasticD
 +      0.0047 * (normalized) SlowD
 +      0.0059 * (normalized) ROC
 +      0.0003 * (normalized) WilliamsR
 +     -0.0048 * (normalized) Disparity5
 +     -0.0045 * (normalized) Disparity10
 +      0.0027 * (normalized) OSCP
 +      0.0082 * (normalized) 20-day CCI
 +     -0.0032 * (normalized) 14-day RSI
 -      1.0023

Number of kernel evaluations: 1744 (66.507% cached)
```

```
Time taken to build model: 0.03 seconds

=== Evaluation on test split ===
=== Summary ===

Correctly Classified Instances         188               75.502  %
Incorrectly Classified Instances        61               24.498  %
Kappa statistic                          0.4586
Mean absolute error                      0.2767
Root mean squared error                  0.3585
Relative absolute error                 85.7453 %
Root relative squared error             89.2747 %
Total Number of Instances              249

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.49      0.061      0.847     0.49      0.621      0.714    DOWN
                 0.939     0.51       0.726     0.939     0.819      0.714    UP
                 0         0          0         0         0          ?        FALSE
Weighted Avg.    0.755     0.326      0.776     0.755     0.738      0.714

=== Confusion Matrix ===

   a   b   c   <-- classified as
  50  52   0 |   a = DOWN
   9 138   0 |   b = UP
   0   0   0 |   c = FALSE
```