STOCK MARKET PREDICTON USING SUPPORT VECTOR MACHINES

[1]Mohd. Noor Md. Sap, [2]A. Majid Awan

Faculty of Computer Sci. & Information Systems, University Technology Malaysia
Skudai 81310, Johor, Malaysia

[1]mohdnoor@fsksm.utm.my, [2]awanmajid@hotmail.com

Abstract: *The stock market is a complex, nonstationary, chaotic and non-linear dynamical system. Therefore, predicting stock price movements is quite difficult. A novel type of learning machine called support vector machine (SVM) has been receiving increasing interest in areas ranging from its original application in pattern recognition to other applications such as regression estimation due to remarkable generalization performance. This paper deals with the application of SVM in financial time series forecasting. Some results for stock price prediction are also presented. Analysis of the experimental results proved that it is advantageous to apply SVMs to forecast financial time series.*

Keywords: financial time series forecasting, nonstationarity, support vector machines (SVM)

1. Introduction

Financial time series forecasting is one of the most challenging applications of modern time series forecasting. Financial time series are inherently noisy, nonstationary and deterministically chaotic [1,2]. These characteristics suggest that there is no complete information that could be obtained from the past behaviour of financial markets to fully capture the dependency between the future price and that of the past.

There are two main categories in financial time series forecasting: univariate analysis and multivariate analysis. In multivariate analysis, any indicator, whether it is related to the output directly or not, can be incorporated as the input variable, while in univariate analysis, the input variables are restricted to the time series being forecasted. A general univariate model that is commonly used is based on the AutoRegressive Integrated Moving Average (ARIMA) method. Compared to other multivariate models, the performance of ARIMA is not satisfactory because this model is parametric, and additionally, it is developed on the assumption that the time series being forecasted are linear and stationary. These constraints are not consistent with the characteristics of financial time series. Therefore, Artificial Neural Network (ANN) assisted multivariate analysis has become a dominant and popular tool in recent years. The prediction performance is greatly improved by the use of a neural network both in terms of prediction metrics and trading metrics [3–6]. Multivariate models can rely on greater information, where not only the lagged time series being forecasted, but also technical indicators, fundamental indicators or inter-market indicators are combined to act as predicators. Moreover, a neural network is more effective in describing the dynamics of non-stationary time series due to its unique non-parametric, non-assumable, noise-tolerant and adaptive properties. Neural networks are universal function approximators that can map any nonlinear function without *a priori* assumptions about the data.

However, a critical issue concerning neural networks is the over-fitting problem. It can be attributed to the fact that a neural network captures not only useful information contained in the given data, but also unwanted noise. This usually leads to a poor level of generalization. The performance of neural networks in terms of generalization for the out-of-sample data–the data that are not used in training the network–is always inferior to that of the training data. Therefore, the development of neural networks and the tasks related to architecture selection, learning

parameter estimation and training, require substantial care in order to achieve the desired level of generalization. The significance of good generalization is critical when using neural networks for financial time series forecasting.

The issue of generalization has long been a concern to researchers, who have explored a variety of procedures for enhancing the generalization ability of neural networks. A typical approach uses crossvalidation, where the data given is divided into three sub-samples. The first sample is for training, the second one for testing, while the third one for validating. This approach involves a substantial amount of computation that is often referred to as a weakness in theory. Nonparametric probability density estimation is another statistical tool used to improve generalization [7], but its underlying assumption–that the distribution for both patterns and noise remains the same over the model estimation and forecasting period–could often not be satisfied in financial time series. Other techniques of enhancing the generalization ability of neural networks include modifying training algorithms [8], pruning the connections or hidden nodes of the network [9], using adaptive learning parameters, and selecting significant variables [10,11].

Recently, SVMs developed by Vapnik [12] have provided another novel approach to improve the generalization property of neural networks. Originally, SVMs were developed for pattern recognition problems. Recently, with the introduction of ε-insensitive loss function, SVMs have been extended to solve non-linear regression problems. Unlike most of the traditional learning machines that adopt the Empirical Risk Minimization Principle, SVMs implement the Structural Risk Minimization Principle, which seeks to minimize an upper bound of the generalization error rather than minimize the training error. This will result in better generalization than that with conventional techniques.

The objective of this paper is to examine the functional characteristics of SVMs in financial forecasting. This paper consists of five sections. Section 2 provides a brief introduction to SVMs while Section 3 contains the experimental data. Section 4 discusses the experimental results followed by the conclusions drawn from this study in the last section.

2. SVM for Regression Estimation

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

Given a set of data points $\{(x_1, y_1),....,(x_t, y_t)\}$, where each $x_i \subset R^n$ denotes the input space of the sample and has a corresponding target value $y_i \subset R$ for $i=1,..., l$ where l corresponds to the size of the training data [12, 22]. The idea of the regression problem is to determine a function that can approximate future values accurately.

The generic SVR estimating function takes the form:

$$f(x) = (w \cdot \Phi(x)) + b \tag{1}$$

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

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

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

$$w = \sum_{i=1}^{\ell} (\alpha_i - \alpha_i^*)\Phi(x_i) \tag{3}$$

By substituting equation (3) into equation (1), the generic equation can be rewritten as:

$$f(x) = \sum_{i=1}^{\ell} (\alpha_i - \alpha_i^*)(\Phi(x_i) \cdot \Phi(x)) + b = \sum_{i=1}^{\ell} (\alpha_i - \alpha_i^*)k(x_i, x) + b \tag{4}$$

In equation (4) the dot product can be replaced with function $k(x_i, x)$, known as the kernel function. Kernel functions enable dot product to be performed in high-dimensional feature space using low dimensional space data input without knowing the transformation Φ. The elegance of using the kernel function is that one can deal with feature spaces of arbitrary dimensionality without having to compute the map explicitly. Any function that satisfies Mercer's condition [12] can be used as the kernel function. Common examples of the kernel function are the polynomial kernel and the Gaussian kernel (radial basis function). The radial basis function (RBF) is commonly used as the kernel for regression:

$$K(x_i, x_j) = \exp(-\|x_i - x_j\|^2 / 2\sigma^2) \tag{5}$$

Some common kernels are shown in Table 1.

Table 1.Common kernel functions

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

The ε-insensitive loss function is the most widely used cost function 0. The function is in the form:

$$\Gamma(f(x) - y) = \begin{cases} |f(x) - y| - \varepsilon, & for \quad |f(x) - y| \geq \varepsilon \\ 0 & otherwise \end{cases} \tag{6}$$

By solving the quadratic optimization problem in (7), the regression risk in equation (2) and the ε-insensitive loss function (6) can be minimized:

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

subject to

$$\sum_{i=1}^{l} \alpha_i - \alpha_i^* = 0, \quad \alpha_i, \alpha_i^* \in [0, C] \tag{7}$$

The Lagrange multipliers, α_i *and* α_i^*, represent solutions to the above quadratic problem that act as forces pushing predictions towards target value y_i. Only the non-zero values of the Lagrange multipliers in equation (7) are useful in forecasting the regression line and are known as support vectors. For all points inside the ε -tube, the Lagrange multipliers equal to zero do not contribute to the regression function. Only if the requirement $|f(x) - y| \geq \varepsilon$ (See Figure 1) is fulfilled, Lagrange multipliers may be non-zero values and used as support vectors.

The constant C introduced in equation (2) determines penalties to estimation errors. A large C assigns higher penalties to errors so that the regression is trained to minimize error with lower generalization while a small C assigns fewer penalties to errors; this allows the minimization of margin with errors, thus higher generalization ability. If C goes to infinitely large, SVR would not allow the occurrence of any error and result in a complex model, whereas when C goes to zero, the result would tolerate a large amount of errors and the model would be less complex.

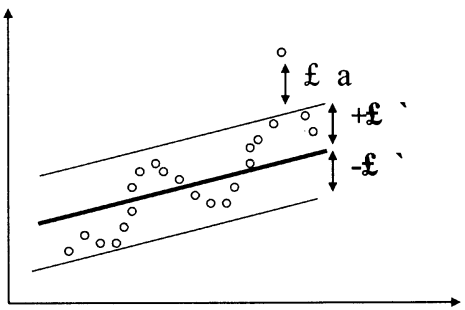

Figure 1. Support vector regression to fit a tube with radius ε to the data and positive slack variables ζ_i measuring the points lying outside of the tube.

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

$$\begin{aligned} \alpha_i(\varepsilon + \zeta_i - y_i + (w, x_i) + b) &= 0 \\ \alpha_i^*(\varepsilon + \zeta_i^* + y_i - (w, x_i) - b) &= 0 \end{aligned} \tag{8}$$

and

$$\begin{aligned} (C - \alpha_i)\zeta_i &= 0 \\ (C - \alpha_i^*)\zeta_i^* &= 0 \end{aligned} \tag{9}$$

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

$$b = y_i - (w, x_i) - \varepsilon \quad for\ \alpha_i \in (0, C)$$
$$b = y_i - (w, x_i) + \varepsilon \quad for\ \alpha_i^* \in (0, C) \tag{10}$$

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

From the implementation point of view, training SVM is equivalent to solving the linearly constrained quadratic programming problem (6) with the number of variables twice as that of the number of training data points. The sequential minimal optimization (SMO) algorithm extended by Scholkopf and Smola [14], [15] is very effective in training SVM for solving the regression estimation problem.

In the next section, we apply our inferential result of SVR based on the general type of ε-insensitive loss function to the regression of financial data, for example, indices and stock prices. By applying regression to the data, we can build a dynamic system to model the data and hence use the system for predicting future prices.

3. Experimental Settings

In this section, we conduct experiments to illustrate the usefulness of SVM for financial prediction. In our experiment, we use the daily closing prices of Kuala Lumpur Stock Exchange (KLSE) Index from 01 April 2002 up to the end of December 2004. Among these data points, the training data points cover the period from 01 April 2002 up to the end of December 2003, while the data points staring from 01 March 2004 up to the end of December 2004 are used as the test data. We set the length of the shift window to 100. The dynamic system is modeled as $\hat{I}_t = f(I_{t-4}, I_{t-3}, I_{t-2}, I_{t-1})$, where I_t is the real stock price at time t, and $\hat{I}_t$ is the predictive value at time t. Therefore, the first training data set is a total of 100 days' of KLSE index. For example the closing stock prices of a selected company for 100 are shown in Figure 2. We use these to predict the next day's index. This window is then shifted and an entire training is performed again to predict the following day's index for the remaining test data.

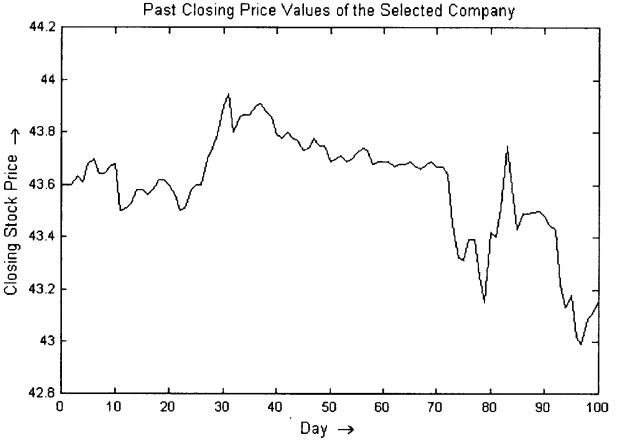

Figure 2. Closing stock prices of a selected company for 100 days.

4. Experimental Results

In this investigation, the Gaussian function is used as the kernel function of the SVMs because Gaussian kernels tend to give good performance under general smoothness assumptions. Consequently, they are especially useful if no additional knowledge of the data is available [15]. The SVR algorithm used in our experiment is modified from LibSVM [23]. Before running the algorithm, we need to determine some parameters. They are C, the cost of error; parameter of kernel function, and the margins. After performing a crossvalidation in the first training data, we set values of parameters. Our experiments show that a width value, σ, of the Gaussian function of 10 is found to produce the best possible results. C and ε are chosen to be 10 and 10^{-3} respectively, because these values produced the best possible results according to the validation set.

The prediction performance is evaluated using the statistical metrics: Mean Squared Error (MSE). MSE is a measure of the deviation between actual and predicted values. The smaller the values of MSE, the closer are the predicted time series values to that of the actual values. The experiments are conducted on a Pentium 4, with 1.7 GHZ, 1 GB RAM and WindowsXP. With these configurations, the prediction results are obtained within seconds.

The behaviour of the MSE (Mean Squared Error) is illustrated in Figure 3 and Figure 5. It is evident that the MSE on the training set is very low during the entire training period as shown in Figure 3. The graphs of actual and predicted stock price values for the training period of 100 days are almost overlapping. However, the MSE on the test set fluctuates and is higher than the MSE for the training period, as shown in Figure 5. It can be seen in Figure 5, the graphs of actual and predicted prices for the test period of 101 to 110 days are deviating from each other, whereas as the graphs for the training period until 100 days are overlapping.

Figure 3. Actual and Predicted stock prices for a training period of 100 days

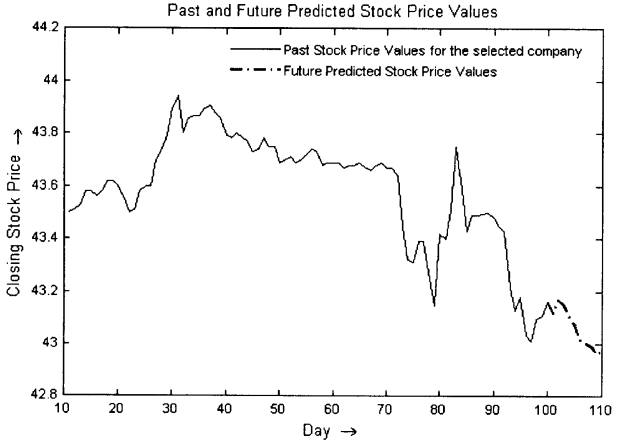

Figure 4. Predicted stock prices for a training period of 100 days and test period of day 101-110

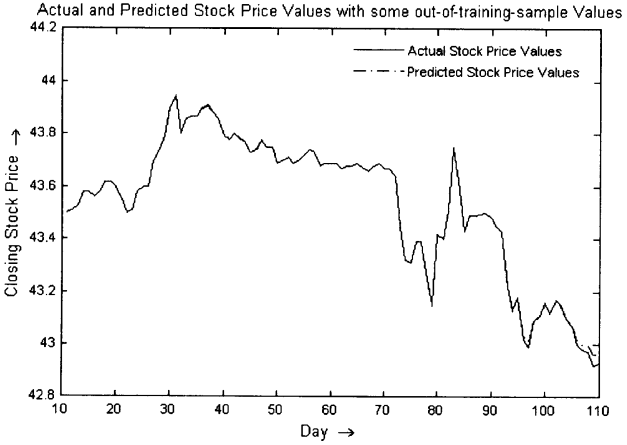

Figure 5. Actual and Predicted stock prices for training period of 100 days and test period of day 101-110

From Figure 5 it is evident that the predicted values obtained from SVMs are closer to the actual values, indicating that there is a smaller deviation between the actual and predicted values. All the above results are found to be consistent with the statistical learning theory.

As already mentioned above, before running the algorithm, we need to determine some parameters. They are C, the cost of error; parameter of kernel function, and the margins. After performing a crossvalidation in the first training data, we set values of parameters. Our experiments show that a width value, σ, of the Gaussian function of 10 is found to produce the best possible results. C and ε are chosen to be 10 and 10^{-3} respectively, because these values produced the best possible results according to the validation set. Also, we use different values for the kernel parameter. Specifically, σ = 0.90, 2.5, 5, 7.5, 10, 12.5, 15. As we increase σ, the accuracy of the prediction also increases. This can be seen from Table 2.

Table 2: MSE values for different values of σ for the test data

σ	0.9	2.5	5	7.5	10	12.5	15
MSE	35.73	23.32	7.31	5.12	4.10	3.69	4.01

5. Conclusions

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

References

[1] Deboeck GJ. Trading on the Edge: Neural, Genetic, and Fuzzy Systems for Chaotic Financial Markets. New York, Wiley, 1994.

[2] Yaser SAM, Atiya AF. Introduction to financial forecasting. Applied Intelligence 1996; 6: 205–213

[3] Abecasis SM, Lapenta ES. Modeling multivariate time series with a neural network: comparison with regression analysis. Proceedings of INFONOR'96: IX International Symposium in Informatic Applications, Antofagasta, Chile, 1996, pp. 18–22

[4] Cheng W, Wanger L, Lin CH. Forecasting the 30-year US treasury bond with a system of neural networks. J Computational Intelligence in Finance 1996; 4: 10–16

[5] Sharda R, Patil RB. A connectionsist approach to time series prediction: an empirical test. Neural Networks in Finance and Investing, 1993; 451–464

[6] Van E, Robert J. The Application of Neural Networks in the Forecasting of Share Prices. Finance & Technology Publishing, Haymarket, VA, USA, 1997

[7] Pan ZH, Wang XD. Wavelet-based density estimator model for forecasting. J Computational Intelligence in Finance 1998; 6: 6–13

[8] Kimoto T, Asakawa K, Yoda M, Takeoka M. Stock market prediction system with modular neural networks. Neural Networks in Finance and Investing 1993; 343–357

[9] Dorsey R, Sexton R. The use of parsimonious neural networks for forecasting financial time series. J Computational Intelligence in Finance 1998; 6: 24–30

[10] Abecasis SM, Lapenta SL. Modeling the MERVAL index with neural networks and the discrete wavelet transform. J Computational Intelligence in Finance 1997; 5: 15–19

[11] VanAussem A, Campbell J, Murtagh F. Wavelet-based feature extraction and decomposition strategies for financial forecasting. J Computational Intelligence in Finance 1998; 6: 5–12

[12] Vapnik VN. The Nature of Statistical Learning Theory. New York, Springer-Verlag, 1995

[13] H. W. Kuhn and A. W. Tucker, "Nonlinear programming," in *Proc. 12th Berkeley Symp. Mathematical Statistics and Probabilistics*. Berkeley, CA: Univ. California Press, 1951, 481–492.

[14] A. J. Smola and B. Scholkopf, "A tutorial on support vector regression," Royal Holloway College, London, U.K., NeuroCOLT Tech. Rep. TR, 1998.

[15] A. J. Smola, "Learning with kernels," Ph.D. dissertation, GMD, Birlinghoven, Germany, 1998.

[16] Thomason M. The practitioner methods and tool. J Computational Intelligence in Finance 1999; 7: 36–45

[17] Thomason M. The practitioner methods and tool. J Computational Intelligence in Finance 1999; 7: 35–45

[18] Scholkopf B, Burges CJC, Smola AJ. Advances in Kernel Methods. The MIT Press, London, England, 1999

[19] Smola AJ, Scholkopf B. A tutorial on support vector regression. NeuroCOLT Technical Report TR, Royal Holloway College, London, UK, 1998

[20] Smola AJ. Learning with Kernels. PhD Thesis, GMD, Birlinghoven, Germany, 1998

[21] Haykin, S. Neural Networks: a Comprehensive Foundation. Prentice Hall International, 1999

[22] K.R. Muller, A. Smola, G. Ratch, B. Scholkopf, J. Kohlmorgen, and V Vapnik, "Using Support Vector Support Machines for Time Series Prediction", Image Processing Services Research Lab, AT&T Labs

[23] Chih-Chung Chang and Chih-Jen Lin, LIBSVM : a library for support vector machines, 2001. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm