Expert Systems with Applications 36 (2009) 10896–10904

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Using support vector machine with a hybrid feature selection method to the stock trend prediction

Ming-Chi Lee *

Department of Computer Science and Information Engineering, National Pingtung Institute of Commerce, No. 51 Minsheng E. Rd., Pingtung 900, Taiwan, ROC

ARTICLE INFO

Keywords:
Support vector machine
Feature selection
Stock index

ABSTRACT

In this paper, we developed a prediction model based on support vector machine (SVM) with a hybrid feature selection method to predict the trend of stock markets. This proposed hybrid feature selection method, named *F-score and Supported Sequential Forward Search* (F_SSFS), combines the advantages of filter methods and wrapper methods to select the optimal feature subset from original feature set. To evaluate the prediction accuracy of this SVM-based model combined with F_SSFS, we compare its performance with back-propagation neural network (BPNN) along with three commonly used feature selection methods including Information gain, Symmetrical uncertainty, and Correlation-based feature selection via paired *t*-test. The grid-search technique using 5-fold cross-validation is used to find out the best parameter value of kernel function of SVM. In this study, we show that SVM outperforms BPN to the problem of stock trend prediction. In addition, our experimental results show that the proposed SVM-based model combined with F_SSFS has the highest level of accuracies and generalization performance in comparison with the other three feature selection methods. With these results, we claim that SVM combined with F_SSFS can serve as a promising addition to the existing stock trend prediction methods.

1. Introduction

In modern finance, derivatives such as futures and options play increasingly prominent roles not only in risk management activities but also in price speculative activities. Owing to the high-leverage characteristic involved in derivative tradings, investors can gain enormous profits with a small amount of capital if they can accurately predict the market's direction. However, predicting the financial market's movements is quite difficult (Tsaih, Hsu, & Lai, 1998).

Increasingly, according to academic investigations, movements in market prices are not random. Rather, they behave in a highly nonlinear, dynamic manner. The standard random walk assumption of futures prices may merely be a veil of randomness that shrouds a noisy nonlinear process (Tsaih et al., 1998). To remove this veil and to make the forecasting of futures prices more reliable, the application of artificial intelligence (AI), such as expert systems (Kee & Koh, 1994), fuzzy system (Chang & Liu, 2008), and neural network (Chiang, Urban, & Baldridge, 1996) has received extensive attention. In particular, back-propagation neural network (BPNN) is frequently used in stock market since the power of prediction is known to be better than the others; however, it has been commonly reported that BPNN models require a large amount of training data to estimate the distribution of input pattern, and they have difficulties of generalizing the results because of their overfitting nature. In addition, it fully depends on researchers' experience or knowledge to preprocess data in order to select control parameters including relevant input variables, hidden layer size, learning rate, and momentum (Lawrence, Giles, & Tsoi, 1997; Min & Lee, 2005; Weigend, 1994).

Support vector machine (SVM), developed by Vapnik (1995), is a relatively new machine learning technique and is gaining popularity due to many attractive features and excellent generalization performance on a wide range of problems. It captures geometric characteristics of feature space without deriving weights of networks from the training data; it is capable of extracting the optimal solution with the small training set size. Also, SVM embodies the structural risk minimization principle (SRM), which has been shown to be superior to traditional empirical risk minimization principle (ERM) employed by conventional neural networks. SRM minimizes an upper bound of generalization error as opposed to ERM that minimizes the error on training data. Therefore, the solution of SVM may be global optimum while other neural-network models tend to fall into a local optimal solution, and over-fitting is unlikely to occur with SVM (Cristianini & Shawe-Taylor, 2000; Kim, 2003; Min & Lee, 2005). However, despite the fact that SVM has outstanding performance, its classification performance and classifier's generalization ability is often influenced by its

* Fax: +886 8 7223962.
E-mail address: lmc@npic.edu.tw.

0957-4174/$ - see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2009.02.038

dimension or number of feature variables. In the stock prediction applications, high-dimensional feature vectors impose a high computational cost as well as the risk of “over-fitting”. Feature selection addresses the dimensionality reduction problem by determining a subset of available features which is most essential for classification (Liu & Zheng, 2006). In this study, we developed a prediction model based on support vector machine with a hybrid feature selection method to predict stock trend. This proposed hybrid feature selection method, which combines filter method and wrapper method to find the suboptimal features for the original input features. In addition, bearing in mind that the optimal parameter search plays a crucial role to build a stock trend prediction model with high prediction accuracy and stability, we employ a grid-search technique using 5-fold cross-validation to find out the optimal parameter values of kernel function of SVM. To evaluate the prediction accuracy of this proposed SVM-based model with the hybrid feature selection method, we also compare its performance with back-propagation neural network (BPNN) along with three commonly used feature selection methods such as Information Gain, Symmetrical uncertainty, and Correlation-based feature selection via paired *t*-test.

This paper is organized as follows. Section 2 describes feature selection methods. Section 3 introduces support vector machine. Section 4 proposes the SVM-based prediction model. Section 5 presents the experimental results and analysis. Section 6 gives remarks and provides a conclusion.

2. Feature selection

In general, feature selection algorithms designed with different evaluation criteria broadly fall into two categories: the filter methods (Dash et al., 2002; Hall, 2000; Liu & Setiono, 1996; Yu & Liu, 2003), and the wrapper methods (Caruana & Freitag, 1994; Dy & Brodley, 2000; Kim, Street, & Menczer, 2000; Kohavi & John, 1997). Fig. 1a and b gives the outline of the two methods. Acquiring no feed back from classifier, the filter methods estimate the classification performance by some indirect assessments, such as distance measures which reflect how well the classes separate from each other (Huang, Dian-Xiu, & Chuang, 2007). The wrapper methods, on the contrary, are classifier-dependent. Based on the classification accuracy, the methods evaluate the “goodness” of the selected feature subset directly, which should intuitively yield better performance. As a matter of fact, many experimental results reported so far are in favor of the wrapper methods (Chiang et al., 1996; Lawrence et al., 1997; Min & Lee, 2005). In spite of the good performance, the wrapper methods have limited applications due to the high computational complexity involved. To take advantage of the filter and wrapper methods and avoid the high computational complexity of wrapper methods, this paper presents a hybrid feature selection method named *F-score and supported sequential forward search* (F_SSFS) in the context of support vector machine, where *F*-score plays the role of filter and *supported sequential forward search* (SSFS) plays the role of wrapper. This proposed hybrid method makes use of both an independent measure and a classifier to evaluate feature subsets, where the F_SSFS uses the *F*-score measure to decide the best subsets for a given feature set and uses the SVM classifier to select the final best subset among the best subsets across different feature sets. Fig. 1c gives the outline of the proposed method. By introducing a filtering process for each SSFS iteration, F_SSFS reduces the number of features that has to be tested through the training of SVM. Then the pre-selected features are considered “informative”, and are evaluated by the accuracy of classification as in the conventional wrapper method. In this way, we are able to reduce the unnecessary computation time spent on the testing of the “noninformative” features while maintaining the good performance delivered by the wrapper method. In the following, *F*-score and SSFS are introduced.

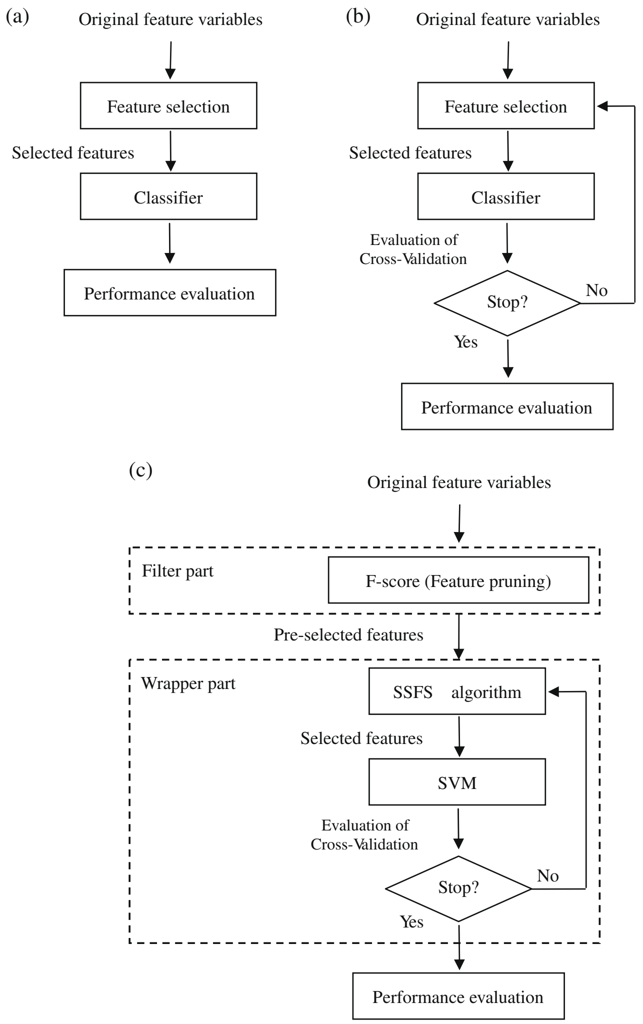


Fig. 1. The comparison among (a) filter methods, (b) wrapper methods, and (c) hybrid methods for feature selection.

2.1. F-score

F-score is a simple filter technique which measures the discrimination of two sets of real numbers (Chen & Lin, 2005; Huang et al., 2007). Given training vectors x_k, $k = 1, 2, \ldots, m$, if the number of positive and negative instances are n_+ and n_-, respectively, then the *F*-score of the *i*th feature is defined as follows:

$$F(i) \equiv \frac{(\bar{x}_i^{(+)} - \bar{x}_i)^2 + (\bar{x}^{(-)} - \bar{x}_i)^2}{\frac{1}{n_+ - 1}\sum_{k=1}^{n_+}(x_{k,i}^{(+)} - \bar{x}_i^{(+)})^2 + \frac{1}{n_- - 1}\sum_{k=1}^{n_-}(x_{k,i}^{(-)} - \bar{x}_i^{(-)})^2}, \qquad (1)$$

where $\bar{x}_i, \bar{x}_i^{(+)}$, and $\bar{x}^{(-)}$ are the averages of the *i*th feature of the whole, positive, and negative data sets, respectively; $x_{k,i}^{(+)}$ is the *i*th feature of the *k*th positive instance, and $x_{k,i}^{(-)}$ is the *i*th feature of the *k*th negative instance. The numerator indicates the discrimination between the positive and negative sets, and the denominator indicates the one within each of the two sets. The larger the *F*-score is, the more likely this feature is more discriminative (Chen & Lin, 2005).

2.2. Supported sequential forward search

A number of search methods have been developed, starting with sequential backward search (SBS) and its "bottom up" counterpart known as sequential forward search (SFS). SFS algorithm is breadth search algorithms executing through a pruned tree (Somol, 1999). Supported sequential forward search (SSFS) is a kind of wrapper method, which is basically a variation of the sequential forward search (SFS) algorithm that is specially tailored to SVM to expedite the feature searching process (Liu & Zheng, 2006). Recall that in SVM, there is a special group of training samples named "support vectors", whose corresponding coefficients are non-zeros. In other words, training samples other than support vectors have no contribution to determine the decision boundary. Since the number of support vectors is relatively small, SSFS could train SVM just by using the support vectors. Following this idea, this research apply the SSFS algorithm, which dynamically maintains an active subset as the candidates of the support vectors, and trains SVM using this reduced subset rather than the entire training set. In this way, SSFS is able to find the boundary with less computational cost.

Consider the binary classification scenario, which has input vectors denoted as $X \in R^k$ and their corresponding class labels denoted as $Y \in \{1, -1\}$. Let

$$F = \{f_1, f_2, \ldots, f_k\} \tag{2}$$

be the set of all features under examination, and let

$$\begin{aligned} S &= \{(X(l), Y(l)|l = 1, 2, \cdots, N\} \\ &= \{[x_1(l)x_2(l)\cdots x_k(l)]^T, Y(l)|1, \ldots, N\} \end{aligned} \tag{3}$$

denotes the training set containing N training pairs, where $x_i(l)$ is the numerical value of feature f_i for the lth training sample.

The goal of feature selection is to find a minimal set of features $F_s = \{f_{s1}, f_{s2}, \ldots, f_{sd}\}$ to represent the input vector X in a lower dimensional feature space as

$$x_s = [x_{s1}x_{s2}\cdots x_{sd}], \tag{4}$$

where $d < k$, while the classifier obtained in the low dimensional representation still yields the acceptable classification accuracy.

The procedure of SSFS is described as follows. The first step is to choose the best single features among the k possible choices. To do so, SSFS trains SVM k times, each of which uses all the training samples available but with only one feature f_i. Mathematically the initial feature combination set is

$$F_1^i = f_i, \quad f_i \in F \tag{5}$$

and the active training set V_1^i, which is the entire training set, is

$$V_1^i = \{1, 2, \ldots, N\}. \tag{6}$$

Although, every training sample in S is involved in this initial training task, the computational complexity is not high because the input vector is just one-dimensional (1-D). After the training, each single-feature combination F_1^i is associated with a value M_1^i, which is the minimum of the objective function, and a group of support vectors v_i. The feature that yields the smallest M_1^i

$$j = \arg\min_{i \in \{1,2,\ldots,N\}} M_1^i \tag{7}$$

is chosen as the best one. Thus SSFS obtains the initial feature combination $F_1 = \{F_j\}$ and its active training set $v_1 = \{v_j\}$.

At step n, SSFS has already obtained the feature combination F_n that contains n features, and the active training set V_n. To add one more feature into the feature combination set, SSFS tests each remaining feature f_i one by one and constructs the corresponding active training set for every new feature combination as follows:

$$\begin{aligned} F_{n+1}^i &= F_n \cup \{f_i\} \text{ for } f_i \in F_n^{av}, \\ V_{n+1}^i &= V_n \cup \{v_i\}, \end{aligned} \tag{8}$$

where $F_n^{av} = \{f_r | f_r \in F \text{ and } f_r \notin F_n\}$ is the collection of the available features to be selected from.

For each F_{n+1}^i, SSFS trains SVM just by using the samples in V_{n+1}^i. The resulting minimum of the objective functions and the collection of the support vectors are denoted as M_{n+1}^i and SV_{n+1}^i, respectively. Then the feature that yields the combination with the least M_{n+1}^i

$$j = \arg\min_{f_i \in F_n^{av}} M_{n+1}^i \tag{9}$$

is selected, and accordingly the new feature combination F_{n+1}, and new active training set V_{n+1} are obtained as follows:

$$\begin{aligned} F_{n+1} &= F_{n+1}^j, \\ V_{n+1} &= SV_{n+1}^j. \end{aligned} \tag{10}$$

The SSFS process continues until no significant reduction of M_n^i is found or the desired number of features has been obtained.

2.3. Related methods of feature selection

We apply three feature selection methods to compare the proposed hybrid feature selection. Three feature selection methods are statistical methods for selecting feature (Huang et al., 2007), such as:

(1) Information Gain: This method measures the importance of a feature by measuring the Information Gain with the respect to the class. Information Gain is given by:

$$InfoGain = H(Y) - H(Y|X), \tag{11}$$

where X and Y are features and

$$\begin{aligned} H(Y) &= -\sum_{y \in Y} p(y)\log_2(p(y)), \\ H(Y|X) &= -\sum_{x \in X} p(y) \sum_{y \in Y} p(y|x)\log_2(p(y|x)). \end{aligned} \tag{12}$$

(2) Symmetrical uncertainty: This method measures the importance of a feature by measuring the Symmetrical uncertainty with respect to the class, and the balances for the Information Gain's bias is:

$$SU = 2.0 \times \frac{InfoGain}{H(Y) + H(X)}. \tag{13}$$

(3) Correlation-based feature selection: CFS evaluates a subset of features by considering the individual predictive ability of each feature along with the degree of redundancy among them:

$$CFS_s = \frac{k\bar{r}_{cf}}{\sqrt{k + k(k-1)\bar{r}_{ff}}}, \tag{14}$$

where CFS_s is the score of a feature subset S containing k features, $\bar{r}_{cf}$ is the average feature to class correlation ($f \in S$), and $\bar{r}_{ff}$ is the average feature to feature correlation. The difference between normal filter algorithm and CFS is that while normal filter provide scores for each feature independently, CFS gives a heuristic "merit" of a feature subset and reports the best subset it finds.

3. Support vector machine

In this section we will describe the basic SVM concepts for typical two-class classification problems. These concepts can also be found in Kecman (2001), Scho'lkopf and Smola (2000), Cristianini and Shawe-Taylor (2000), Huang, Chen, and Wang (2007). Given a training set of instance-label pairs (x_i, y_i), $i = 1, 2, \ldots, m$ where $x_i \in R^n$ and $y_i \in \{+1, -1\}$, SVM finds an optimal separating hyperplane with the maximum margin by solving the following optimization problem:

$$\begin{aligned} &\min_{w,b} \quad \frac{1}{2} w^T w \\ &\text{subject to}: \quad y_i(\langle w \cdot x_i \rangle + b) - 1 \geqslant 0. \end{aligned} \tag{15}$$

It is known that to solve this quadratic optimization problem one must find the saddle point of the Lagrange function:

$$L_p(w, b, \alpha) = \frac{1}{2} w^T w - \sum_{i=1}^{m} (\alpha_i y_i (\langle w \cdot x_i \rangle + b) - 1), \tag{16}$$

where the α_i denotes Lagrange multipliers, hence $\alpha_i \geqslant 0$. The search for an optimal saddle point is necessary because the L_p must be minimized with respect to the primal variables w and b and maximized with respect to the non-negative dual variable α_i. By differentiating with respect to w and b, and introducing the Karush Kuhn–Tucker (KKT) condition for the optimum constrained function, then L_p is transformed to the dual Lagrangian $L_D(\alpha)$:

$$\begin{aligned} &\max_{\alpha} \quad L_D(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j \langle x_i \cdot x_j \rangle \\ &\text{subject to}: \quad \alpha_i \geqslant 0 \quad i = 1, \ldots, m \text{ and } \sum_{i=1}^{m} \alpha_i y_i = 0. \end{aligned} \tag{17}$$

To find the optimal hyperplane, a dual Lagrangian $L_D(\alpha)$ must be maximized with respect to non-negative α_i. The solution α_i for the dual optimization problem determines the parameters $w^*$and b^* of the optimal hyperplane. Thus, the optimal hyperplane decision function $f(x) = \text{sgn}(\langle w^* \cdot x \rangle + b^*)$ can be written as

$$f(x) = \text{sgn}\left(\sum_{i=1}^{m} y_i \alpha_i^* \langle x_i \cdot x \rangle + b^* \right). \tag{18}$$

In a typical classification task, only a small subset of the Lagrange multipliers α_i usually tends to be greater than zero. Geometrically, these vectors are the closest to the optimal hyperplane. The respective training vectors having non-zero α_i are called support vectors, as the optimal decision hyperplane $f(x,\alpha^*,b^*)$ depends on them exclusively.

The above concepts can also be extended to the non-separable case (linear generalized SVM). In terms of these introduced slack variables, the problem of finding the hyperplane that provides the minimum number of training errors (i.e., to keep the constraint violation as small as possible) has the formal expression as follows:

$$\begin{aligned} &\min_{w,b,\xi} \quad \frac{1}{2} w^T w + C \sum_{i=1}^{m} \xi_i \\ &\text{subject to}: \quad y_i(\langle w \cdot x_i \rangle + b) + \xi - 1 \geqslant 0 \text{ and } \xi_i \geqslant 0, \end{aligned} \tag{19}$$

where C is a penalty parameter on the training error, and ξ_i is the non-negative slack variable. SVM finds the hyperplane that provides the minimum number of training errors (i.e., to keep the constraint violation as small as possible).

This optimization model can be solved using the Lagrangian method, which is almost equivalent to the method for solving the optimization problem in the separable case. One must maximize the dual variables Lagrangian:

$$\begin{aligned} &\max_{\alpha} \quad L_D(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j \langle x_i \cdot x_j \rangle \\ &\text{subject to}: \quad 0 \leqslant \alpha_i \leqslant C \quad i = 1, \ldots, m \text{ and } \sum_{i=1}^{m} \alpha_i y_i = 0. \end{aligned} \tag{20}$$

To find the optimal hyperplane, a dual Lagrangian $L_D(\alpha)$ must be maximized with respect to non-negative α_i under the constrains $\sum_{i=1}^{m} \alpha_i y_i = 0$ and $0 \leqslant \alpha_i \leqslant C$. The penalty parameter C, which is now the upper bound on α_i, is determined by the user. Finally, the form of optimal hyperplane decision function is the same as (18). The nonlinear SVM maps the training samples from the input space into a higher-dimensional feature space via a mapping function Φ. In the dual Lagrange (20), the inner products are replaced by the kernel function (21), and the nonlinear SVM dual Lagrangian $L_D(\alpha)$ (22) is similar with that in the linear generalized case

$$(\Phi(x_i) \cdot \Phi(x_j)) := k(x_i, x_j) \tag{21}$$

$$\begin{aligned} &L_D(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j k(x_i \cdot x_j) \\ &\text{subject to}: 0 \leqslant \alpha_i \leqslant C \quad i = 1, \ldots, m \text{ and } \sum_{i=1}^{m} \alpha_i y_i = 0. \end{aligned} \tag{22}$$

Followed by the steps described in the linear generalized case, we obtain decision function of the following form:

$$\begin{aligned} f(x) &= \text{sgn}\left(\sum_{i=1}^{m} y_i \alpha_i^* \langle \Phi(x) \cdot \Phi(x_i) \rangle + b^* \right) \\ &\quad \text{sgn}\left(\sum_{i=1}^{m} y_i \alpha_i^* \langle k(x \cdot x_i) \rangle + b^* \right). \end{aligned} \tag{23}$$

Radial basis function (RBF) is a common kernel function as follows:

$$k(x_i, x_j) = \exp\left(-\gamma \| x_i - x_j \|^2 \right). \tag{24}$$

4. Research design

4.1. Data collection and preprocessing

This paper wants to predict the direction of the NASDAQ Index. An overwhelming number of studies have examined the price discovery process involving stock indexes and their futures contracts. Many studies find that index futures lead the spot index and contend that this sequence supports the notion that trading costs and response time are correlated (Abhyankar, 1995; Cornell & French, 1983; Iihara, Kato, & Tokunaga, 1996; Tse & Booth, 1996). In this paper, we use 29 technical indices as the whole features set and the direction of change in the daily NASDAQ index as the prediction target. Since this work predicts the direction of daily stock price index, we use '1' and '−1' to denote that the next day's index is higher or lower than today's index, respectively. We collect the raw data series consisting of closing prices of 20 futures contracts and 9 spot indexes. Nine of the 20 futures contracts are on commodities (silver, platinum, palladium, heating oil, copper, gold, crude oil, coal, natural gas), and 11 are on foreign currencies (Swiss frank, yen, mark, Canadian dollar, British pound, Euro dollar, Renminbi, Australia dollar, South Korea dollar, Hong Kong dollar, Singapore dollar). The nine spot indexes are DJIA Index, NYSE Composite Index, Philadelphia Semiconductor Index, UTIL Index, DJCOMP Index, TRAN Index, AMEX Composite Index, Russell 2000 Index, and S&P 500 Index. We also use 1-day lagged NASDAQ Index as an additional explanatory variable for a total of 30 variables. We select these variables because of availability of 6 years of daily data on them—from November 8, 2001, to November 8, 2007—with a total of 1065 observations per variable. The data set was obtained from Taiwan Economic Journal database (TEJD, 2008). For each futures contract series, we extract forecasts at one point in the life of the series. One forecast serves for one horizon: 15 trading days to maturity of a futures contract. The forecast date is 15 days before the maturity of nearest futures contracts. We obtain these forecast dates and horizon using the American futures options pricing model (Barone-Adessi & Whaley, 1987; Hamid & Iqbal, 2004).

In this study, the data set is partitioned into five folds, of which four folds (80%) are used as the training set for building up the forecast model, and the remaining holdout fold (20%) as the forecasts (test set) for justifying the generalization performance of the model. Each fold is randomly constructed. We obtain 200 forecasts. The original data are scaled into the range of (0, 1). The goal of linear scaling is to independently normalize each feature component to the specified range. It ensures the larger value input attributes do not overwhelm smaller value inputs; hence helps to reduce prediction errors (Hsu, Chang, & Lin, 2004).

4.2. SVM-based model with F_SSFS

As mentioned earlier, we propose a hybrid feature selection method, namely "F_SSFS". Fig. 2 gives the outline of the proposed hybrid method which combines the advantage of both the filter and the wrapper methods. In the filtering part, acting in the generic way similar to a filter method, calculates *F*-score for every feature and ranks features without involving the classifier. We sort *F*-score and select *K* (threshold) highest scored features to construct the feature subset F_n^{av}. The features with relatively high ranks are considered as "informative" feature candidates and then are restudied by the wrapper part that further investigates their contributions to a specific SVM.

In the wrapper part, each selected feature f_i does the 5-fold cross-validation and calculates the average accuracy of the 5-fold cross validation. The highest average accuracy of the 5-fold cross-validation that is the least minimum of the objective functions determines the feature to be added in the best feature subset F_n. We choose the next feature using $F_{n+1}^i = F_n \cup \{f_i\}$ for $f_i \in F_n^{av}$ and calculate the average validation accuracy of the 5-fold cross-validation. Thus, we determine the feature to be added using $j = \arg\min_{i\in\{1,2,\ldots,N\}} M_1^i$ and update the active training set using $F_{n+1} = F_{n+1}^j$ and $V_{n+1} = SV_{n+1}^j$. The SSFS process continues until no significant increasing accuracy of cross-validation is found or the desired number of features has been obtained.

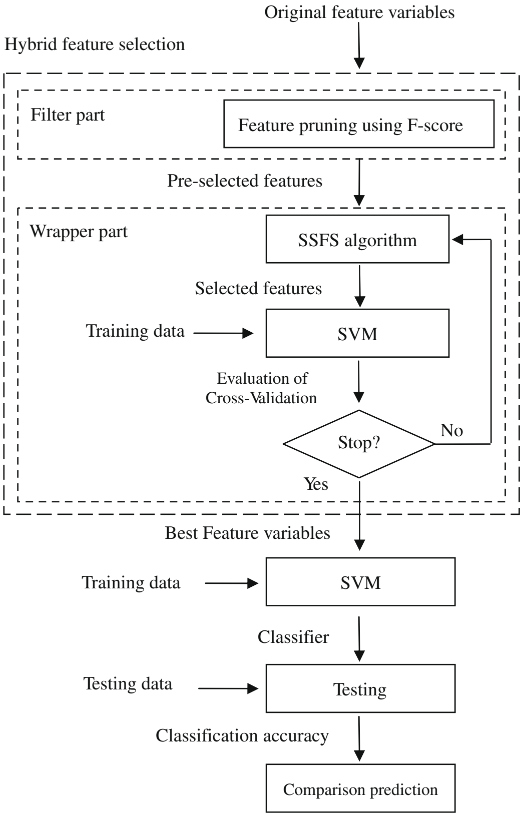


Fig. 2. Procedure of SVM-based model with F_SSFS.

After wrapper processing, we rerun SVM training on the training data to obtain a trained SVM classifier using the best feature combination. Based on the trained SVM model, it measure classification accuracy of stock trend prediction on the test set. Finally, we compare the prediction accuracy of SVM and BPNN using the features selected by Information Gain, Symmetrical uncertainty, CFS and F_SSFS. The procedure and algorithm of the proposed SVM-based model plus F_SSFS is described in Figs. 2 and 3.

4.3. Modeling for support vector machine

Model selection and parameter search play a crucial role in the performance of SVM. However, there is no general guidance for selection of SVM kernel function and parameters so far. In general, the radial basis function (RBF) is suggested for SVM. The RBF kernel nonlinearly maps the samples into the high-dimensional space, so it can handle nonlinear problem. For the nonlinear SVM, there are additional parameters, the kernel parameters γ to tune. Improper selection of the penalty parameter *C* and kernel parameters can cause over-fitting or under-fitting problems (Hsu et al., 2004; Tay & Cao, 2001). Currently, some kinds of parameter search approach are employed such as cross-validation via parallel grid-search, heuristics search, and inference of model parameters within the Bayesian evidence framework. The performance is generally evaluated by cost, e.g. classification accuracy or mean square error (MSE).

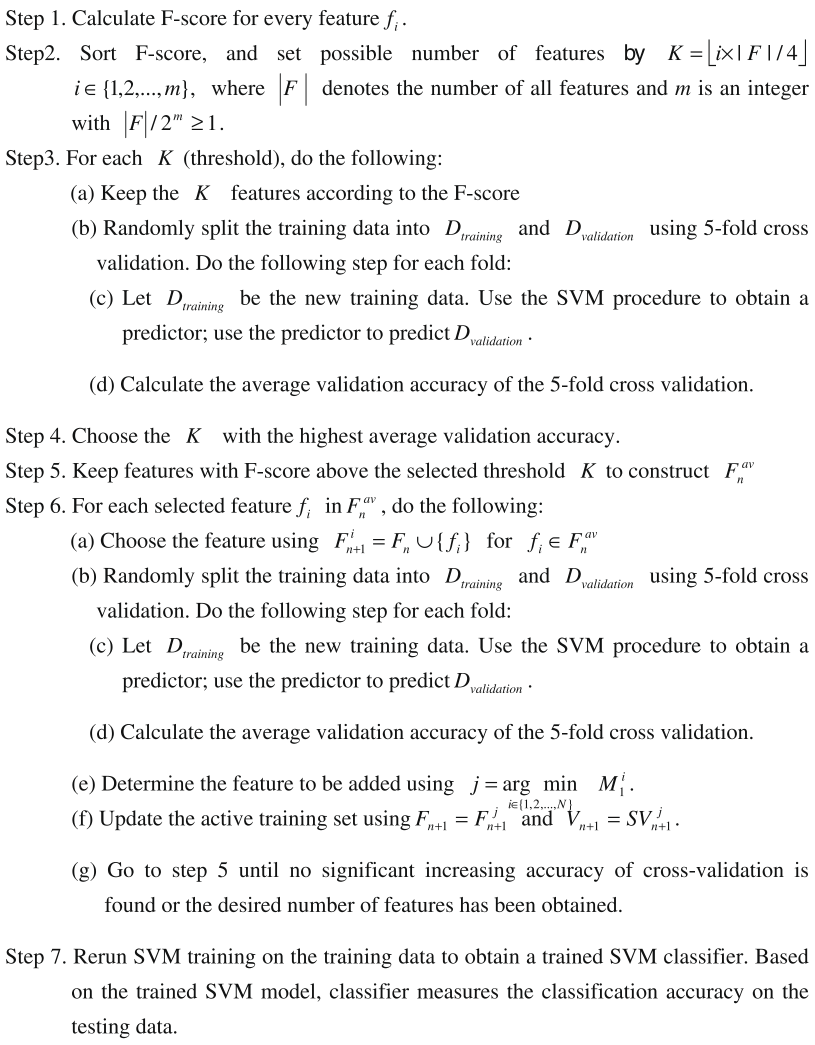
We prefer a grid-search on (C, γ) using 5-fold cross-validation for the following reasons. Firstly, the cross-validation procedure can prevent the over-fitting problem. Secondly, computational time to find good parameters by grid-search is not much more than that by other methods. Furthermore, the grid-search can be easily parallelized because each (C, γ) is independent. In order to increase efficiency, we try exponentially growing sequences of (C, γ) to identify good parameters. We set the $C = 2^{-5}, 2^{-3}, \ldots, 2^{15}$ and kernel parameter $\gamma = 2^{-15}, 2^{-13}, \ldots, 2^{3}$. The final performance of classifier is evaluated by mean costs of v folds subsets. In grid-search process, pairs of (C, γ) are tried and the one with the best cross-validation accuracy is picked up (Ding, Song, & Zen, 2007; Hsu et al., 2004). We use LIBSVM software to conduct SVM experiment (Hsu et al., 2004).

5. Experimental results and analysis

In the experiments, we apply the proposed SVM-based method to predict NASDAQ Index direction. First, we show the features selected by the proposed F_SSFS method. Second, we employ a grid-search technique using 5-fold cross-validation to find out the optimal parameter values of kernel function of SVM. Thirdly, we compare the prediction accuracy of SVM and BPNN using the features selected by Information Gain, Symmetrical uncertainty, CFS and F_SSFS. Finally, we apply relative importance (RI) to explain the relative contributions of features for stock trend prediction.

5.1. Experimental result of F_SSFS

The threshold *K* determines how many features we want to keep after the filtering. One extreme case is that *K* is equal to the number of all original features. In this scenario, the filter part does not contribute at all. Similarly, if *K* is equal to 1, the wrapper meth-



```
Step 1. Calculate F-score for every feature $f_i$.
Step2. Sort F-score, and set possible number of features by $K = \lfloor i \times |F| / 4 \rfloor$
       $i \in \{1,2,...,m\}$, where $|F|$ denotes the number of all features and $m$ is an integer
       with $|F| / 2^m \geq 1$.
Step3. For each $K$ (threshold), do the following:
       (a) Keep the $K$ features according to the F-score
       (b) Randomly split the training data into $D_{training}$ and $D_{validation}$ using 5-fold cross
           validation. Do the following step for each fold:
          (c) Let $D_{training}$ be the new training data. Use the SVM procedure to obtain a
              predictor; use the predictor to predict $D_{validation}$.
          (d) Calculate the average validation accuracy of the 5-fold cross validation.
Step 4. Choose the $K$ with the highest average validation accuracy.
Step 5. Keep features with F-score above the selected threshold $K$ to construct $F_n^{av}$
Step 6. For each selected feature $f_i$ in $F_n^{av}$, do the following:
       (a) Choose the feature using $F_{n+1}^i = F_n \cup \{f_i\}$ for $f_i \in F_n^{av}$
       (b) Randomly split the training data into $D_{training}$ and $D_{validation}$ using 5-fold cross
           validation. Do the following step for each fold:
          (c) Let $D_{training}$ be the new training data. Use the SVM procedure to obtain a
              predictor; use the predictor to predict $D_{validation}$.
          (d) Calculate the average validation accuracy of the 5-fold cross validation.
       (e) Determine the feature to be added using $j = \arg\min_{i \in \{1,2,...,N\}} M_1^i$.
       (f) Update the active training set using $F_{n+1} = F_{n+1}^j$ and $V_{n+1} = SV_{n+1}^j$.
       (g) Go to step 5 until no significant increasing accuracy of cross-validation is
           found or the desired number of features has been obtained.
Step 7. Rerun SVM training on the training data to obtain a trained SVM classifier. Based
        on the trained SVM model, classifier measures the classification accuracy on the
        testing data.
```

Fig. 3. Algorithm of SVM-based model with F_SSFS.

od is unnecessary. K is usually chosen between the two extremes and thus works as a tuning parameter to balance between the performance and the complexity of the algorithm (Liu & Zheng, 2006). We show the experiment result of different values of the parameter K for the prediction of stock. We set $|F| = 30$ in this threshold experiment. Different values of the parameter K are tried for the prediction of stock trend, and the prediction accuracy obtained are listed in Table 1. Not to our surprise, with the increasing of K the accuracy of the prediction increases but the selection process takes longer time, which confirms that the performance and complexity of the algorithm can be balanced by tuning K.

Table 1
Comparison of the prediction accuracy with different values of the threshold K.

	Prediction accuracy	
	Training (%)	Testing (%)
$K = \left\lfloor \frac{\lvert F \rvert}{4} \right\rfloor = 7$	76.1	75.5
$K = \left\lfloor \frac{\lvert F \rvert}{2} \right\rfloor = 15$	86.8	84.5
$K = \left\lfloor \frac{3\lvert F \rvert}{4} \right\rfloor = 22$	88.0	87.5
$K = \lvert F \rvert = 30$	88.4	87.5

We get a threshold $K = 22$ to construct the feature subset F_n^{av}. The wrapper criterion of feature selection employed in our model is the average cross-validation accuracy of SVM. After the process of wrapper part, 17 feature variables turned out to have contributions with the average cross-validation accuracy rate of 87.7%. The feature selection list of 17 feature variables and their F-score are shown in Table 2.

5.2. Results of SVM model selection

We must first decide which kernels to select for implementing SVM; and then the penalty parameter C and kernel parameters are chosen. For the nonlinear SVM, there are two parameters associated with the RBF kernels: C and γ. It is not known beforehand which values of C and γ are the best for one problem; consequently, some kind of model selection (parameter search) approach must be employed (Kim et al., 2000). This study conducts a grid-search to find the best values of C and γ using 5-fold cross-validation. Pairs of (C, γ) are tried and the one with the best cross-validation accuracy is picked. After conducting the grid-search for training data,

Table 2
The results of *F*-score of selected features and average accuracy rate.

No	Feature variables	*F*-score	Average accuracy rate (%)
1	Yen	14.70	62.3
2	Philadelphia Semiconductor Index	12.30	67.2
3	Renminbi	8.87	71.5
4	Swiss frank	3.41	75.6
5	Australia dollar	2.90	77.4
6	British pound	2.13	78.5
7	Nasdaq 1-day-lagged	1.91	80.0
8	Gold	0.90	81.3
9	DJCOMP Index	0.90	82.1
10	Russell 2000 Index	0.86	83.2
11	Silver	0.76	84.4
12	Palladium	0.70	85.2
13	Coal	0.54	85.8
14	Platinum	0.53	86.4
15	Singapore dollar	0.42	86.8
16	Hong Kong dollar	0.40	87.3
17	Canadian dollar	0.22	87.7

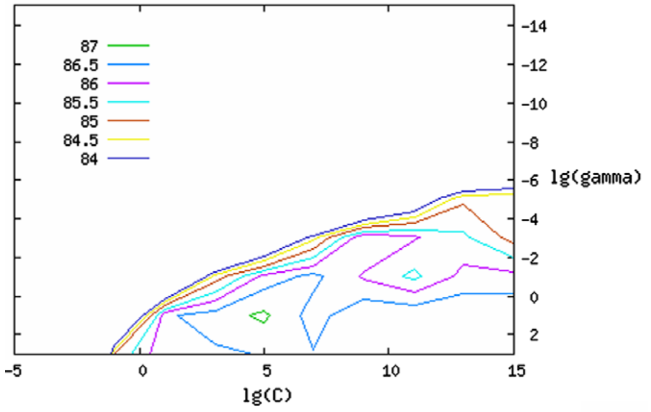


Fig. 4. Grid-search for RBF kernel (C,γ).

we found that the optimal (C,γ) was $(2^5,2^1)$ with cross-validation rate of 87.1% (see Fig. 4). Table 3 summarizes the results of the grid-search using 5-fold cross-validation.

5.3. Experimental result of SVM

In this experiment, SVM achieves accuracy ranging from 83.2% to 87.2% in predicting stock trend with an average of 85.4%. Along with four different feature selection methods, their prediction accuracies are shown in Table 4. SVM achieves accuracy ranging from 85.5% to 88.5% when using the “F_SSFS” with an average of 87.3%, 78% to 83% when using the Information Gain with an average of 80.1%, 78% to 82.5% when using the Symmetrical uncertainty with an average of 80.1%, and 75% to 78% when using CFS with an average of 76.3%. Notably, every fold gives an accuracy of 85% or higher when using “F_SSFS”. Compared to other feature selection methods, it can be seen that “F_SSFS” achieves the best accuracy. Apparently, SVM plus F_SSFS outperform the SVM integrated with the other three feature selection methods.

5.4. Experimental result of BPNN

In this study, a three-layer BPNN is used as benchmark. In BPNN, the number of its hidden nodes is 17 according to the number of input variables suggested by Shin, Lee, and Kim (2005). For the stopping criteria of BPNN, this study allows 1000 learning epoch per one training example. The learning rate is set to 0.1 and the momentum term is to 0.5. The hidden nodes use the hyperbolic tangent transfer function and the output node uses the same transfer function. We use Matlab 7 to perform the BPNN experiments.

Table 5 exhibits the classification accuracy when the model is trained using BPNN. First, BPNN achieves accuracy ranging from 70.2% to 73.1% with an average of 71.5%. Together with four different feature selection methods, BPNN achieves accuracy ranging from 71.5% to 74% when using the “F_SSFS” with an average of 72.5%, 65% to 68.5% when using the Information Gain with an average of 66.8%, 65% to 68.5% when using the Symmetrical uncertainty with an average of 66.8%, and 62% to 64% when using CFS with an average of 63%. Notably, every fold gives an accuracy of 71% or higher when using “F_SSFS”. Compared to the other three feature selection methods, it can be seen that “F_SSFS” achieves the best accuracy. Apparently, BPNN plus F_SSFS outperform BPNN combined with the other three feature selection methods.

5.5. Comparing prediction accuracy of SVM models and BPNN

The comparison of NASDAQ Index trend prediction accuracy using SVM and BPNN classifiers along with different feature selection methods is given in Table 6. For each feature selection, SVM outperforms BPNN for the test data. In comparison with BPNN’s 72.5%, 66.8%, 66.8%, and 63% accuracies in terms of using “F_SSFS”, Information Gain, Symmetrical uncertainty and CFS methods, respectively, SVM’s 87.3%, 80.1%, 80.1% and 76.3% performance is definitely superior to BPNN performance. A paired-samples *t*-test is conducted to evaluate the significance regarding to SVM and BPNN performance. As shown in Table 6, the hypothesis, the mean accuracy of BPNN is equal to the mean accuracy of the SVM, has been significantly rejected with $\alpha = 0.05$, which proves that the SVM outperforms BPNN based on statistical analysis.

Table 3
The result of grid-search using 5-fold cross-validation.

C	γ									
	2^{-15}	2^{-13}	2^{-11}	2^{-9}	2^{-7}	2^{-5}	2^{-3}	2^{1}	2^{-1}	2^{3}
2^{-5}	62.4855	62.4855	62.4855	62.4855	62.4855	62.4855	62.4855	62.4855	62.4855	62.4855
2^{-3}	62.4855	62.4855	62.4855	62.4855	62.4855	62.4855	63.2985	74.3322	67.712	68.8734
2^{-1}	62.4855	62.4855	62.4855	62.4855	62.4855	63.9954	71.6609	80.7201	74.7967	85.0174
2^{1}	62.4855	62.4855	62.4855	62.4855	63.9954	70.4994	74.7967	86.4111	80.7201	86.4111
2^{3}	62.4855	62.4855	62.4855	63.4146	67.9443	73.0546	78.2811	86.7596	84.669	86.4111
2^{5}	62.4855	62.4855	63.5308	67.4797	71.5447	75.1452	81.8815	87.108	86.1789	86.5273
2^{7}	62.4855	63.6469	67.4797	70.151	72.9384	77.4681	84.3206	86.295	86.6434	86.5273
2^{9}	63.6469	67.3635	69.8026	72.0093	75.7259	80.8362	86.4111	86.8757	85.9466	86.5273
2^{11}	67.3635	69.6864	70.7317	73.403	77.4681	83.043	86.0627	86.8757	85.3659	86.5273
2^{13}	69.5703	70.2671	71.777	75.6098	80.0232	84.9013	85.5981	86.8757	86.1789	86.5273
2^{15}	70.0348	70.6156	73.7515	77.5842	81.3008	84.9013	84.7851	86.8757	86.1789	86.5273

Table 4
The performance of SVM model with four different feature selection methods using 5-fold cross-validation.

Classifier + feature selection methods	Accuracy of 5-fold cross-validation					
	Set 1	Set 2	Set 3	Set 4	Set 5	Average
SVM + F_SSFS	85.5	87.0	87.0	88.5	88.5	87.3
SVM + Information Gain	78.0	79.0	79.0	81.5	83.0	80.1
SVM + Symmetrical uncertainty	78.0	79.5	79.5	81.0	82.5	80.1
SVM + CFS	75.0	75.0	76.5	77.0	78.0	76.3
SVM	83.2	85.3	86.2	85.3	87.2	85.4

Table 5
The performance of BPNN model with four different feature selection methods using 5-fold cross-validation.

Classifier + feature selection methods	Accuracy of 5-fold cross-validation					
	Set 1	Set 2	Set 3	Set 4	Set 5	Average (%)
BPNN + F_SSFS	71.5	71.5	72.0	73.5	74.0	72.5
BPNN + Information Gain	65.0	65.0	67.0	68.5	68.5	66.8
BPNN + Symmetrical uncertainty	65.0	65.0	66.5	68.5	69.0	66.8
BPNN + CFS	62.0	62.5	63.0	63.5	64.0	63.0
BPNN	70.2	71.1	70.7	72.2	73.1	71.5

Table 6
Paired *t*-test and Mann–Whitney nonparametric test comparison between BPNN and SVM.

Feature selection methods	Classifier	Accuracy (%)	Paired *t*-test		Mann–Whitney nonparametric test	
			T statistic	*p* (two-tailed)	*Z* statistics	*p* (two-tailed)
F_SSFS	SVM	87.3	19.268	0.001**	−3.496	0.001**
	BPNN	72.5				
Information Gain	SVM	80.1	10.951	0.001**	−3.046	0.002**
	BPNN	66.8				
Symmetrical uncertainty	SVM	80.1	11.665	0.001**	−3.046	0.002**
	BPNN	66.8				
CFS	SVM	76.3	19.504	0.001**	−2.922	0.003**
	BPNN	63.0				

** *p*-Value < 0.01.

Since the standard *t*-tests assume normal distribution of the data and our data may not be normally distributed, we also perform a nonparametric test—the Mann–Whitney test. This test performs a two sample rank test for the difference between two population medians. In the results of Mann–Whitney nonparametric tests, SVM forecasts show significant differences from BPNN forecasts in the case of all feature selection methods in two-tailed tests. That means, whereas BPNN has provided good forecasts, SVM forecasts have been good in the case of all feature selection methods. From the results of experiments, it clearly implies that for the data in our sample, the prediction forecasts from SVM provided superior results compared to BPNN.

5.6. Experimental result of feature selection

A key deficiency of neural-network models for stock trend prediction applications is the difficulty in selecting the discriminative features and explaining the rationale for the stock trend prediction (Huang et al., 2007). We apply Garson's index (Garson, 1991) to estimate the relative contributions of input features. After performing 5-fold cross-validation, for each attribute, we calculate its average Garson's index (BPNN), and the average *F*-score ("SVM + F_SSFS"). Fig. 5 illustrates the relative importance of each feature with the form of its relative percentage.

Garson (1991) found a method to solve the relative importance for each input parameters. The function of relative importance is to correlate the input neuron and output neuron. The equation can be written as:

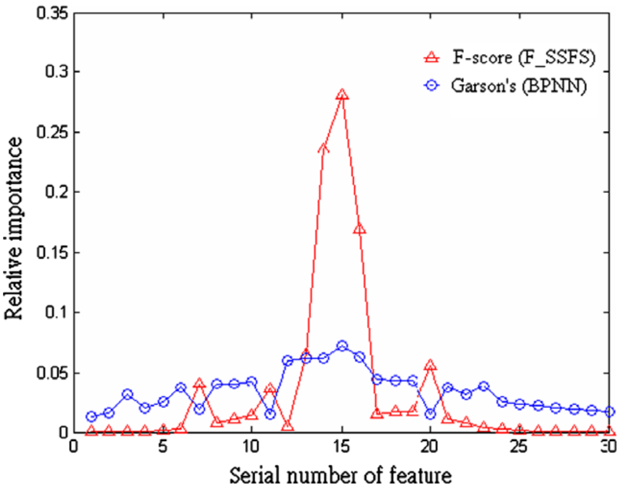


Fig. 5. Relative importance of features.

$$RI_i = \frac{\sum_j^{n_H}\left[\frac{I_{vj}}{\sum_k^{n_v} I_{vj}} O_j\right]}{\sum_i^{n_v}\left[\sum_j^{n_H}\left[\frac{I_{vj}}{\sum_k^{n_v} I_{vj}} O_j\right]\right]}, \tag{26}$$

where n_H is the number of hidden layer's neuron, n_v is the number of input layer's neuron, v is the neuron associate to the 'v' input layers, j is the neuron associate to the 'j' hidden layers, I_{vj} is the weight value between the 'v' input layers to the 'j' hidden layers, and O_j is the weight value between the output layers to the 'j' hidden layers.

For each attribute, we calculated the relative importance (RI) of all features for F-score ("SVM + F_SSFS"). The relative importance of F-score is defined as follows:

$$RI_i = \frac{F_i}{\sum_{i=1}^{n} F_i}, \tag{27}$$

where F_i is the F-score of ith feature, and $\sum_{i=1}^{n} F_i$ is the sum F-score of all features where n is the number of features, and $i = 1,2,\ldots,n$. Fig. 5 illustrates them with the form of their relative percentage.

In this study, only about 17 of the 30 input variables are finally present in the stock trend prediction model for the case of the sample data. For "SVM + F_SSFS" model, the 15th feature is the most important one for index prediction. Some features do not contribute to the "SVM + F_SSFS" model (e.g., features 1 to 5–25 to 30). For BPNN model, all input variables seem to contribute to the output decision.

6. Conclusion

This study proposed a prediction model based on support vector machine with a hybrid feature selection method to the problem of stock trend prediction. Fundamentally this hybrid feature selection method, which is named F_SSFS, is a more efficient version of a wrapper/SFS approach. F_SSFS introduces a feature pruning process into the wrapper part such that some "noninformative" features are filtered out and consequently the number of SVM training is reduced. Furthermore, during the SSFS searching process, an active training set is maintained as the candidates of the support vectors. In this way, the number of samples participating in a single optimization procedure decreases, reducing high computational cost as well as the risk of "over-fitting". Our experimentation results demonstrate that the proposed SVM-based model with F_SSFS has the highest level of accuracies and better generalization performance than BPN. With these results, we claim that the SVM-based model with F_SSFS can serve as a promising addition to the existing stock trend prediction methods.

Our study has the following limitation that needs further research. First, in SVM, The choice of feature variables has a critical impact on the performance of the resulting system. We also need to investigate to develop a structured method of selecting an optimal value of the parameters in SVM for the best prediction performance. The second issue for future research relates to the generalization of SVM on the basis of the appropriate level of the training set size and gives a guideline to measure the generalization performance.

References

Abhyankar, A. H. (1995). Return and volatility dynamics in the FTSE100 stock index and stock index futures markets. *The Journal of Futures Markets, 15*(4), 457–488.

Barone-Adessi, G., & Whaley, R. E. (1987). Efficient analytic approximation of American option values. *Journal of Finance*, 42, 301–320.

Caruana, R., & Freitag, D., (1994). Greedy attribute selection. In *Proceedings of the 11th international conference on machine learning* (pp. 28–36).

Chang, P.-C., & Liu, C.-H. (2008). A TSK type fuzzy rule based system for stock price prediction. *Expert Systems with Applications, 34*(1), 135–144.

Chen, Y.-W., & Lin, C.-J., (2005). Combining SVMs with various feature selection strategies. Available from: <http://www.csie.ntu.edu.tw/~cjlin/papers/features.pdf>.

Chiang, W.-C., Urban, T., & Baldridge, G. (1996). A neural network fund net asset approach to mutual value forecasting. *Omega*.

Cornell, B., & French, K. R. (1983). The pricing of stock index futures. *The Journal of Futures Markets, 3*(1), 1–14.

Cristianini, N., & Shawe-Taylor, J. (2000). *An introduction to support vector machines*. Cambridge: Cambridge University Press.

Dash, M. et al., (2002). Feature selection for clustering – a filter solution. In *Proceedings of the second international conference on data mining* (pp. 115–122).

Ding, Y., Song, X., & Zen, Y. (2007). Forecasting financial condition of Chinese listed companies based on support vector machine. *Expert Systems with Applications, 34*(4), 3081–3089.

Dy, J. G., & Brodley, C. E., (2000). Feature subset selection and order identification for unsupervised learning. In *Proceedings of the 17th international conference on machine learning* (pp. 247–254).

Garson, G. D. (1991). Interpreting neural-network connection weights. *AI Expert*, 47–51.

Hall, M. A., (2000). Correlation-based feature selection for discrete and numeric class machine learning. In *Proceedings of the 17th international conference on machine learning* (pp. 359–366).

Hamid, S. A., & Iqbal, Z. (2004). Using neural networks for forecasting volatility of S&P 500 Index futures prices. *Journal of Business Research, 57*(10), 1116–1125.

Hsu, C.-W., Chang, C.-C., & Lin, C.-J., (2004). *A practical guide to support vector classification. Technical report*. Department of Computer Science and Information Engineering, National Taiwan University.

Huang, C.-L., Chen, M.-C., & Wang, C.-J. (2007). Credit scoring with a data mining approach based on support vector machines. *Expert Systems with Applications, 33*(4), 847–856.

Huang, C.-J., Dian-Xiu & Chuang, Y.-T. (2007). Application of wrapper approach and composite classifier to the stock trend prediction. *Expert Systems with Applications, 34*(4), 2870–2878.

Iihara, Y., Kato, K., & Tokunaga, T. (1996). Intraday return dynamics between the cash and the futures markets in Japan. *The Journal of Futures Markets, 16*(2), 147–162.

Kecman, V. (2001). *Learning and soft computing*. Cambridge, MA: The MIT Press.

Kee, W. Y., & Koh, A. (1994). Technical analysis of Nikkei 225 stock index futures using an expert system advisor. In *Proceedings of the CBOT conference*.

Kim, K.-J. (2003). Financial time series forecasting using support vector machines. *Neurocomputing, 55*(1–2), 307–319.

Kim, Y., Street, W., & Menczer, F. (2000). Feature selection for unsupervised learning via evolutionary search. In *Proceedings of the sixth ACM SIGKDD international conference knowledge discovery and data mining* (pp. 365–369).

Kohavi, R., & John, G. H. (1997). Wrappers for feature subset selection. *Artificial Intelligence, 97*(1–2), 273–324.

Lawrence, S., Giles, C. L., & Tsoi, A.-C. (1997). Lessons in neural network training: Overfitting may be harder than expected. In *Proceedings of the fourteenth national conference on artificial intelligence, AAAI-97* (pp. 540–545).

Liu, H., & Setiono, R., (1996). A probabilistic approach to feature selection – a filter solution. In *Proceedings of the 13th international conference on machine learning* (pp. 319–327).

Liu, Y., & Zheng, Y. F. (2006). FS_SFS: A novel feature selection method for support vector machines. *Pattern Recognition, 39*(7), 1333–1345.

Min, J. H., & Lee, Y.-C. (2005). Bankruptcy prediction using support vector machine with optimal choice of kernel function parameters. *Expert Systems with Applications, 28*(4), 603–614.

Scho'lkopf, B., & Smola, A. J. (2000). *Statistical learning and kernel methods*. Cambridge, MA: MIT Press.

Shin, K.-S., Lee, T. S., & Kim, H.-J. (2005). An application of support vector machines in bankruptcy prediction model. *Expert Systems with Applications, 28*(1), 127–135.

Somol, P. et al. (1999). Adaptive floating search methods in feature selection. *Pattern Recognition Letters, 20*(11–13), 1157–1163.

Tay, F. E. H., & Cao, L. (2001). Application of support vector machines in financial time series forecasting. *Omega, 29*(4), 309–317.

TEJD. (2008). *Taiwan economic journal database*. Available form <http://61.30.108.162/image/readme1.htm>.

Tsaih, R., Hsu, Y., & Lai, C. C. (1998). Forecasting S&P 500 stock index futures with a hybrid AI system. *Decision Support Systems, 23*(2), 161–174.

Tse, Y., & Booth, G. (1996). Common volatility and volatility spillovers between US. and Eurodollar interest rates: Evidence from the futures market. *Journal of Econometrics and Business, 48*, 299–312.

Vapnik, V. N. (1995). *The nature of statistical learning theory*. New York: Springer-Verlag.

Weigend, A., (1994). On overfitting and the effective number of hidden units. In *Proceedings of the 1993 connectionist models summer school* (pp. 335–3420).

Yu, L., & Liu, H., (2003). Feature selection for high-dimensional data: A fast correlation-based filter solution. In *Proceedings of the 20th international conference on machine learning* (pp. 856–863).