Available online at www.sciencedirect.com
ScienceDirect
Expert Systems with Applications 34 (2008) 2870–2878
Expert Systems with Applications
www.elsevier.com/locate/eswa

Application of wrapper approach and composite classifier to the stock trend prediction

Chenn-Jung Huang *, Dian-Xiu Yang, Yi-Ta Chuang

Department of Computer and Information Science, College of Science, National Hualien University of Education, Taiwan

Abstract

The research on the stock market prediction has been more popular in recent years. Numerous researchers tried to predict the immediate future stock prices or indices based on technical indices with various mathematical models and machine learning techniques such as artificial neural networks (ANN), support vector machines (SVM) and ARIMA models. Although some researches in the literature exhibit satisfactory prediction achievement when the average percentage error and root mean square error are used as the performance metrics, the prediction accuracy of whether stock market goes or down is seldom analyzed. This paper employs wrapper approach to select the optimal feature subset from original feature set composed of 23 technical indices and then uses voting scheme that combines different classification algorithms to predict the trend in Korea and Taiwan stock markets. Experimental result shows that wrapper approach can achieve better performance than the commonly used feature filters, such as χ^2-Statistic, Information gain, ReliefF, Symmetrical uncertainty and CFS. Moreover, the proposed voting scheme outperforms single classifier such as SVM, *k*th nearest neighbor, back-propagation neural network, decision tree, and logistic regression.

Keywords: Stock prediction; Wrapper; Voting; Feature selection; Classification

1. Introduction

Stock market prediction is regarded as a challenging task of financial time-series prediction. There have been many studies using artificial neural networks (ANNs) in this area. A large number of successful applications have shown that ANN can be a very useful tool for time-series modeling and forecasting. The early researchers focused on application of ANNs to stock market prediction. Recent research tends to hybridize several artificial intelligence (AI) techniques. Kim and Han proposed a genetic algorithms approach to feature discretization (Kim & Han, 2000) and the determination of connection weights for ANN to predict the stock price index. They suggested that their approach reduced the dimensionality of the feature space and enhanced the prediction performance.

Some of these studies, however, showed that ANN had some drawback in learning the patterns because stock market data has enormous noise and complex dimensionality. Thus, ANN exhibits inconsistent and unpredictable performance on noisy data. Nevertheless, back-propagation (BP) neural network, the most popular neural network model, suffers from difficulty in selecting a large number of controlling parameters which include relevant input variables, hidden layer size, learning rate and momentum term.

Recently, a support vector machine (SVM), a novel neural network algorithm, was developed by Vapnik (1998). Many traditional neural network models had implemented the empirical risk minimization principle, while SVM implements the structural risk minimization principle. The former seeks to minimize the misclassification error or deviation from correct solution of the training data but the latter searches to minimize an upper bound of generalization error. In addition, the solution of SVM may be global optimum while other neural network models may

* Corresponding author.
E-mail address: cjhuang@mail.nhlue.edu.tw (C.-J. Huang).

0957-4174/$ - see front matter © 2007 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2007.05.035

tend to fall into a local optimal solution. Thus, over-fitting is unlikely to occur with SVM.

Kim (2003) proposed a SVM approach to predict the direction of the stock price. Eleven technical indices were taken as the inputs in Kim (2003) and the best prediction rate is up to 57%. To tackle this challenge, we attempt to use an appropriate feature selection method to select the most relevant technical indices from 23 commonly used ones and then feed the chosen technical indices into the SVM classifier to predict future stock trend in Taiwan and Korea markets. Moreover, we propose a new voting scheme which combines different classification algorithms with the feature set selected by wrapper approach to each classifier. The difference between the ordinary voting scheme, named stacking, and our proposed voting scheme is that the ordinary stacking scheme only combines several different classifiers to make the consensus. In our scheme, we further use wrapper feature selection algorithm to find the finest feature set for each specified classifier we employ in the voting scheme.

The remainder of this paper is organized as follows. In Section 2, we describe the feature selection methods in data mining domain. The voting scheme plus wrapper approach for feature selection is present in Section 3. In Section 4, wrapper approach is compared with the commonly used feature selection methods as described in Section 2 and the comparison of the proposed voting scheme with different single classifiers is also presented. Conclusions are given in Section 5.

2. Related works

In many practical situations, there are far too many features related to stock trend classification. Some of them are irrelevant and some are redundant from the viewpoint of machine learning domain. It is well-known that the inclusion of irrelevant and redundant information may cause incorrect result of some machine learning algorithms.

Feature subset selection can be seen as a search through the space of feature subsets. There are many approaches for feature selection proposed in the literature, such as:

(1) χ^2-Statistic: This method measures the importance of a feature by computing the value of the χ^2-statistic with respect to the class.
(2) 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:

$$\text{InfoGain} = H(Y) - H(Y|X), \tag{1}$$

where X and Y are features and

$$H(Y) = -\sum_{y\in Y} p(y)\log_2(p(y)), \tag{2}$$

$$H(Y|X) = -\sum_{x\in X} p(x)\sum_{y\in Y} p(y|x)\log_2(p(y|x)). \tag{3}$$

Both, the information gain and the χ^2-statistic, are biased in favor of features with higher dispersion.
(3) 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:

$$\text{SU} = 2.0 \times \frac{\text{InfoGain}}{H(Y) + H(X)}. \tag{4}$$

(4) ReliefF: This method is feature weighting algorithm that is sensitive to feature interaction. The key idea of ReliefF is to rate features according to how well their values distinguish among instances of different classes and how well they cluster instances of the same class. To this end, ReliefF repeatedly chooses a single instance at random from data, and then locates the nearest instances of the same class and the nearest instances pertaining to different classes. The feature values of these instances are used to update the scores for each feature.
(5) 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:

$$\text{CFS}_S = \frac{k\bar{r}_{cf}}{\sqrt{k + k(k-1)\bar{r}_{ff}}}, \tag{5}$$

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. Wrapper approach plus voting machine technique

3.1. Wrapper feature selection method

The wrapper approach searches for an optimal feature subset tailored to the particular algorithm (Kohavi & John, 1995), whereas the filter approaches attempt to measure values of features from the data set. The concept of wrapper approach is shown in Fig. 1. In the wrapper approach, the feature subset selection is done by induction algorithm as a black box. The feature subset selection algorithm conducts a search for a good subset using the induction algorithm itself as part of the evaluation function. The accuracy of induced classifiers is estimated by accuracy estimation technique. The classification algorithm itself is used to determine the attribute subset. Since the wrapper approach optimizes the evaluation measure of the classification algorithm while removing features, it mostly leads to greater accuracy than the so-called filter approaches as described in Section 2.

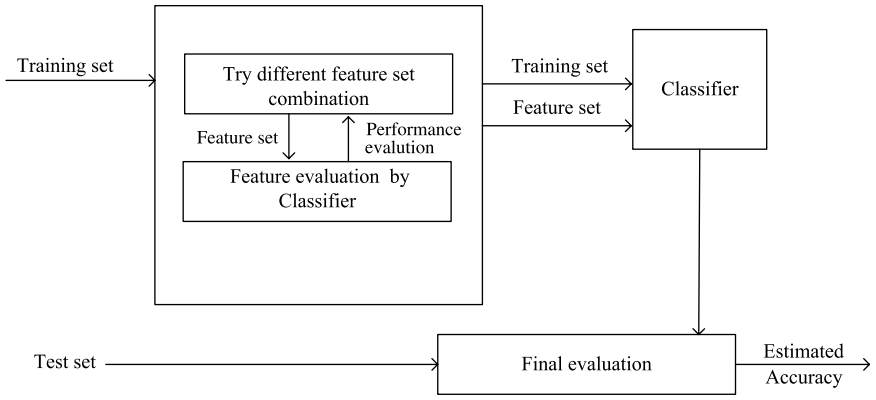

Fig. 1. The wrapper approach.

3.2. Voting machine technique

Voting is a well-known aggregation procedure that combines different opinions of voters into consensus. In the simplest form of voting method, each data item is assigned a number of votes. Since there are advantages and disadvantages for different classification algorithms, we thus try to combine support vector machine (SVM), *k*th nearest neighbors, back-propagation neural network, decision tree and logistic regression into the voting scheme in this paper in order to predict the direction of change in daily Taiwan and Korea stock price indices. Meanwhile, we adopt different features selected by wrapper method for different classification algorithms since we believe that it may not be suitable to use the same feature set for different algorithms.

3.2.1. Support vector machines

In a decade, support vector machines (SVMs) (Burges, 1998; Chang & Lin, 2001; Cristianini & Shawe-Taylor, 2000; Haykin, 1994; Vapnik, 1995) have attracted much attention as a new classification technique with good generalization ability. The basic idea of SVMs is to map input vectors into a high-dimensional feature space and linearly separate the feature vectors with an optimal hyper-plane in terms of margins, i.e. distances of given examples from a separating hyper-plane.

Support vector machines (SVMs) are promising methods for the prediction of financial time-series because they use a risk function consisting of the empirical error and a regularized term which is derived from the structural risk minimization principle. Given a training dataset represented by the X-matrix $(X_1,\ldots,X_m)$ divided into two linearly separable classes with class labels (+1 and −1) stored in the Y-vector $(y_1,\ldots,y_m)$ as given in Fig. 1, the maximum margin plane can be found by minimizing ($\|w\|_2$):

$$\|w\|_2 = w \cdot w = \sum_{i=1}^{d} w_i^2, \tag{6}$$

with constraints:

$$y_i(w \cdot x_i + b) \geqslant 1, \tag{7}$$

where $i = 1,\ldots,m$, $b \in R$, and $x_i \in R^d$.

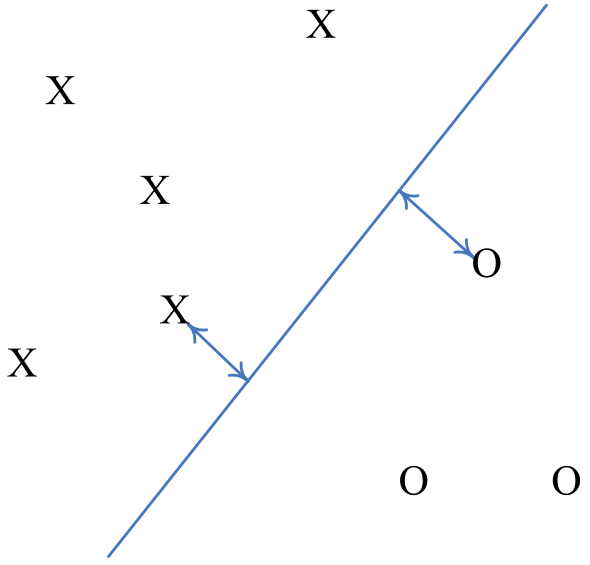

Fig. 2. Support vectors and margin.

Fig. 2 shows the simplest case of two linearly separable classes. The concept of support vectors is illustrated by the points closest to the surface separating two classes, and the margin is given by the distance between support vectors and separating surface.

The decision function takes the form $f(x) = \mathrm{sgn}(w \cdot x + b)$, where sgn(·) is simply a sign function which returns +1 for positive arguments and −1 for negative arguments. This simple classification problem is generalized to a non-separable case by introducing slack variables x_i and minimizing the following quantity:

$$\frac{1}{2} w \cdot w = C \sum_{i=1}^{m} \xi_i, \tag{8}$$

where $y_i(w \cdot x_i + b) \geqslant 1 - \xi_i$ and $\xi_i > 0$.

The above quadratic optimization problem with constraints can be reformulated by introducing Lagrangian multipliers α, v, and the following Lagrangian is formed,

$$L(w,b,\xi,\alpha,v) = \frac{1}{2}w\cdot w + C\sum_{i=1}^{m}\xi_i - \sum_{i=1}^{m}\alpha_i[y_i(w\cdot x_i+b)-1+\xi_i] - \sum_{i=1}^{m}v_i\xi_i. \tag{9}$$

Stationary points of this Lagrangian can be obtained by:

$$\frac{\partial L}{\partial w} = w - \sum_{i=1}^{m}\alpha_i y_i x_i = 0, \tag{10}$$

$$\frac{\partial L}{\partial b} = \sum_{i=1}^{m}\alpha_i y_i = 0, \tag{11}$$

$$\frac{\partial L}{\partial \xi_i} = \alpha_i + v_i - C = 0. \tag{12}$$

Two remaining derivatives $\left(\frac{\partial L}{\partial \alpha}, \frac{\partial L}{\partial v}\right)$ recover the constraint equations. By substituting the expression $w = \sum_{i=1}^{m}\alpha_i y_i x_i$ back into the Lagrangian, we obtain this simpler dual formulation

$$W(\alpha) = \frac{1}{2}\sum_{i=1}^{m}\alpha_i\alpha_j y_i y_j x_i\cdot x_j - \sum_{i=1}^{m}\alpha_i, \tag{13}$$

with the constraints $0 \leqslant \alpha_i \leqslant C$ and $\sum\alpha_i y_i = 0$.

Given a mapping

$$x \to \phi(x), \tag{14}$$

the dot product in the final space can be replaced by a Mercer kernel

$$\phi(x)\phi(y) \to K(x,y). \tag{15}$$

Since $\phi(\cdot)$ typically maps x into the space of much higher dimensionality, it is usually specified by defining the kernel implicitly. The above dual formulation thus becomes a preferred approach due to highly dimensional feature space induced by $\phi(\cdot)$ mapping. The decision function for classification problems is then given by:

$$f(x) = \operatorname{sgn}\left(\sum_{sv}\alpha_i y_i K(x_i,x) + b\right). \tag{16}$$

3.2.2. kth nearest neighboring

The kth nearest neighbor (KNN) algorithm (Kelly et al., 1991; Peterson, Doom, & Raymer, 2005) is a classification algorithm based on closest training example feature space. The training phase of the algorithm consists of storing the feature vectors and class labels of the training samples. In the actual classification phase, the same features are computed as before for the test sample whose class is not known. Distances from the new vector to all stored vectors are computed and k closest samples are selected. The new point is predicted to belong to the most numerous class within the set.

The KNN method is a simple yet effective method for classification in the areas of pattern recognition, machine learning, data mining, and information retrieval. It has been successfully used in a variety of real-world applications. The performance of KNN can be comparable with the state-of-the-art classification methods with simpler computation (Atkeson, Moore, & Schaal, 1997).

Given an instance with unknown classification, instances with known classification that are closer to this instance are given more weights. The distance or similarity between instances is typically determined by the Euclidean distance. However, other distance functions can also be applied. The Euclidean distance defines the dissimilarity or distance $d(i,j)$between instances i and j as:

$$d(i,j) = \sqrt{\left(\frac{|x_{i1-j1}|}{R_1}\right)^2 + \left(\frac{|x_{i2-j2}|}{R_2}\right)^2 + \cdots + \left(\frac{|x_{ip-jp}|}{R_p}\right)^2}, \tag{17}$$

where $R_f = \max_h x_{hf} - \max_h x_{hf}$, which denotes the range of attribute f.

Once k nearest neighbors with known classification are selected for an unclassified instance p, a classification combination method that combines the classifications from the k nearest neighbors predicts the classification for p. The simplest classification combination method is the voting method. The classification for p is assigned as the majority class in the k nearest neighbors. The second classification combination method eliminates the effect of unequal instances of different classes in the k nearest neighbors by taking the average distance for each class. Thus, the instance p is classified as belonging to class Y if:

$$\frac{1}{k_1}\sum_{i\in Y(p,k)} d(i,p) < \frac{1}{k_2}\sum_{i\in N(p,k)} d(i,p), \tag{18}$$

where $k = k_1 + k_2$, k_1 is the number of instances belongs to class Y in the k nearest neighbors, and k_2 is the number of instances belonging to class N in k nearest neighbors.

The third combination method compares the sum of similarity of each class in the k nearest neighbors. Assume there are two decision classes, Y and N. The instance p will be classified as belonging to class Y if:

$$\sum_{i\in Y(p,k)} d(i,p) < \sum_{i\in N(p,k)} d(i,p). \tag{19}$$

When the distribution of class Y and N are extremely asymmetric in the training data set, it may lead to the classification decision that favors the class with majority instances. Instead of selecting k nearest neighbors for an unclassified instance p, k_1 nearest neighbors that belong to class Y and k_2 nearest neighbors belonging to N are selected, where k_1 and k_2 are user-defined parameters.

Fig. 3 shows an example of KNN algorithm. Here the displayed training points represented by hollow circle and solid circle are known to the algorithm. When a new point, X_q, is queried, the nearest three points are found by using the City-block distance measure. Two of the points closest to the query point are hollow circle and only one is solid circle so X_q is classified as hollow circle.

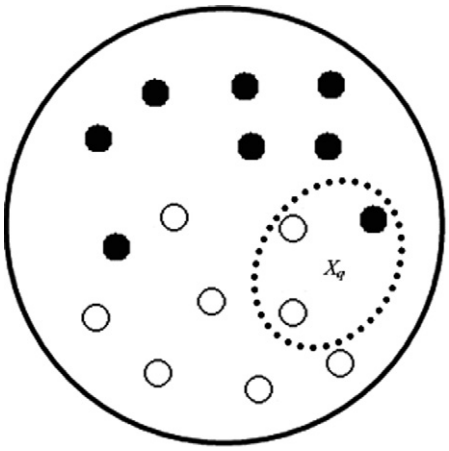

Fig. 3. An example of KNN algorithm.

3.2.3. Back-propagation neural network

A neural network is able to recognize patterns and generalize from them. An essential feature of this technology is that it improves its performance on a particular task by gradually learning a mapping between inputs and outputs. Generalization is used to predict the possible outcome for a particular task. This process involves two phases known as the training phase and the testing phase. Back-propagation (BP) neural network (Medsker & Liebowitz, 1994; Russel & Norvig, 1995) is one of neural networks most in common use. Fig. 4 shows an example of back-propagation neural network. In back-propagation, the learning procedure basically follows that of a traditional feed-forward neural network. However, there are two main differences. The first difference is the use of the activation function of the hidden unit y_j, and the second is that the gradient of the activation function is contained.

A back-propagation neural network consists of several layers of nodes including an input layer, one or more hidden layers and an output layer. Each node in a layer receives its input from the output of the previous layer nodes. The connections between nodes are associated to synaptic weights that are iteratively adjusted during the training process. Each hidden and output node is associated to an activation function. Several functions can be used as activation functions, but the most common choice is the sigmoid function:

$$f(a) = \frac{1}{1 + e^{-a}}. \tag{20}$$

Provided that the activation function of the hidden layer nodes is non-linear, a back-propagation neural network with an adequate number of hidden nodes is able to approximate every non-linear function. The adjustment of the synaptic weights in an error back-propagation algorithm consists of four steps:

(1) The network is initialized by assigning random values to synaptic weights.
(2) A training pattern is fed and propagated forward through the network to compute an output value for each output node.
(3) Actual outputs are compared with the expected outputs.
(4) A backward pass through the network is performed, changing the synaptic weights on the basis of the observed output errors.

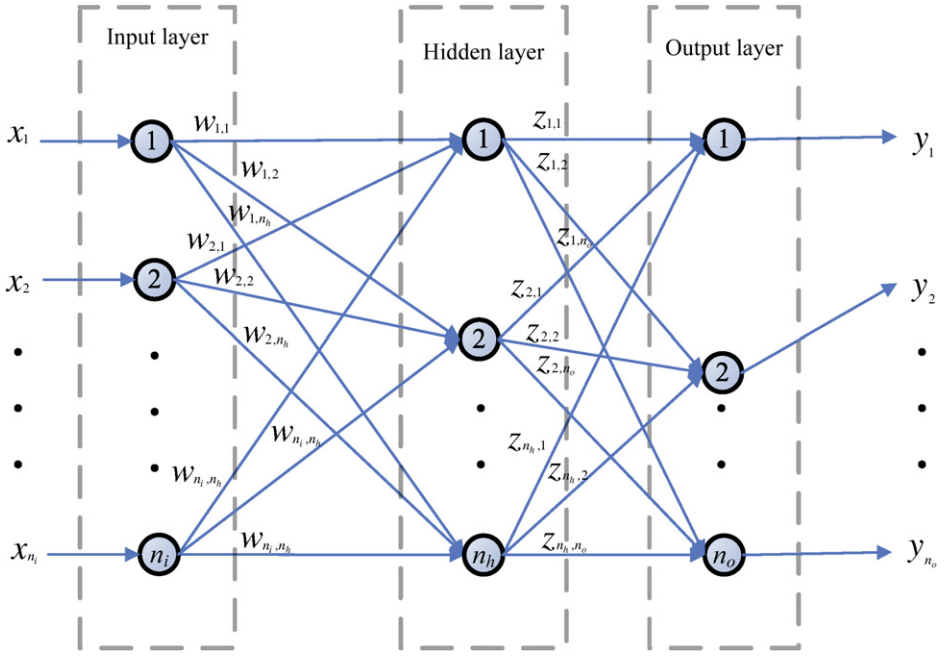

Fig. 4. A back-propagation neural network.

Steps (2)–(4) are iterated for each pattern in a training set until convergence.

In the case of neural networks with a single hidden layer as shown in Fig. 4, the forward propagation step is carried out as follows:

$$h_k = \sum_{j=1}^{n_h+1} x_j w_{jk} \quad (k = 1, 2, \ldots, n_h), \tag{21}$$

where $x_j$is the jth input from input layer, and w_{jk} is the weight of the connections between input x_j and the kth node at hidden layer. To compute the outputs of the hidden layer, these weighted sums are passed to the activation function,

$$g_k = f(h_k), \tag{22}$$

$$g_{n_h+1} = 1, \tag{23}$$

where g_{n_h+1} denotes the output of the bias node at hidden layer. Then, the network outputs are computed by:

$$o_l = \sum_{k=1}^{n_h+1} h_k z_{kl} \quad (l = 1, 2, \ldots, n_o). \tag{24}$$

After the forward propagation, estimated output o_l of the lth node at output layer is compared with expected output $y_l$and a mean quadratic error for the current pattern is derived by:

$$E = \frac{1}{n_o} \sum_{l=1}^{n_o} (y_l - o_l)^2 \tag{25}$$

In the back-propagation step, all the synaptic weights are adjusted in order to follow a gradient descent on the error surface. For the connection weight between the kth node at hidden layer and the lth node at output layer, z_{kl}, is adjusted by:

$$z_{kl} = z_{kl} + \eta \delta_l^o h_k \quad (k = 1, 2, \ldots, n_h + 1; l = 1, 2, \ldots, n_o), \tag{26}$$

where η denotes the learning rate and

$$\delta_l^o = (y_l - o_l) \cdot f'(o_l) = (y_l - o_l) \cdot o_l(1 - o_l). \tag{27}$$

The weight w_{jk} of the connection between the kth node at hidden layer and the jth input is adjusted by:

$$w_{jk} = w_{jk} + \eta \delta_k^h x_j \quad (k = 1, 2, \ldots, n_h; j = 1, 2, \ldots, n_i), \tag{28}$$

where δ_k^h is computed by,

$$\delta_k^h = f'(h_k) \cdot \sum_{l=1}^{n_o} \delta_l^o z_{kl} = h_k(1 - h_k) \cdot \sum_{l=1}^{n_o} \delta_l^o z_{kl}. \tag{29}$$

The network training is iterated until a given condition is met.

3.2.4. Decision tree

Decision tree (Breiman, Friedman, Olshen, & Stone, 1984; Hughes, 1968; Safavian & Landgrebe, 1991) is a predictive mode, a mapping of observations about an item to conclude about the item's target value. Each interior node corresponds to a variable; an arc to a child represents a possible value of that variable. A leaf represents the predicted value of target variable given the values of the variables represented by the path from the root. There are several advantages for decision tree. For instance, it is simple to understand and interpret, and it is able to handle nominal and categorical data and perform well with large data set in a short time. In this work, we use C4.5 decision tree to predict the direction change of stock price because C4.5 decision tree performs well in prediction application as report in Peterson et al. (2005).

A decision tree is a hierarchy of yes/no questions in which the specific questions asked depend on the answers given to the previous questions, with the branches spreading out from the original question until an appropriate response is given. Decision trees can be used to encapsulate the knowledge of an expert about a specific system. Various methods exist for the development of decision trees from datasets, with the goal of each method being to produce a structure that gives the highest degree of accuracy for the smallest tree design (Endou & Zhao, 2002; Llorà & Garrell, 2001; Papagelis & Kalles, 2001).

Fig. 5 shows an example of decision tree algorithm. The algorithm starts at the topmost point in the tree and ask the question, ⌜What is the outlook for the day ?⌟ The answer to the question determines the path we take through the tree. For instance, if the response to the question is "Overcast", we move down the middle path to a position which provides a class value for the observation in question. The square given in Fig. 5 represents an example of final destinations, leaf node, which has no other paths leading away from it.

Decision tree learning is one of the most widely used and practical methods for inductive inference. It is a method for approximating discrete-valued functions that is robust to noisy data and capable of learning disjunctive expressions. C4.5, one of the most widely used decision tree learning algorithm, is adopted in our work.

In the process of constructing the decision tree, the root node is first selected by evaluating each attribute using a statistical test to determine how well it alone classifies the training examples. The best attribute is selected and used

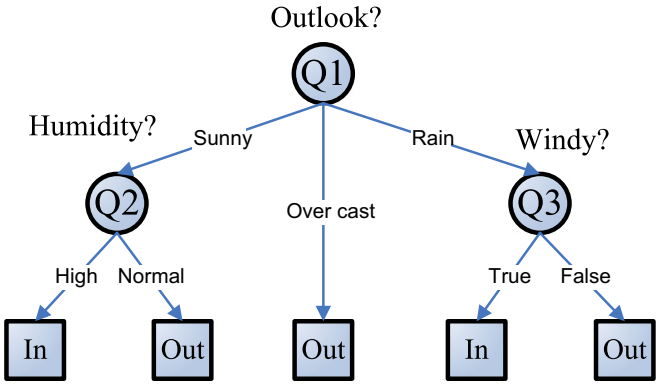

Fig. 5. An example of decision tree algorithm.

to test at the root node of the tree. A descendant of the root node is created for each possible value of this selected attribute, and the training examples are sorted to the appropriate descendant node. The entire process is then repeated using the training examples associated with each descendant node to select the best attribute to test at that point in the tree.

There are two frequently used metrics for attributes selection. One is the information gain, $\mathrm{Gain}(S,A)$ of an attribute A, relative to a collection of examples S,

$$\mathrm{Gain}(S,A) = \mathrm{Entropy}(S) - \sum_{v\in \mathrm{Value}(A)} \frac{|S_v|}{|S|}\mathrm{Entropy}(S_v), \tag{30}$$

where $\mathrm{Value}(A)$ is the set of all possible values for attribute A, and S_v is the subset of S for which attribute A has value v. $\mathrm{Entropy}(S)$ is the entropy of S. The entropy is defined as:

$$\mathrm{Entropy}(S) = -\sum_{i=1}^{c} p_i \log_2 p_i, \tag{31}$$

where c denotes the number of all possible values for attribute A. Notably, the first term on the right hand side of Eq. (30) denotes the entropy of the original collection S, and the second term is the expected value of the entropy after S is partitioned using attribute A.

The second metric commonly used for attributes selection is called Gain Ratio,

$$\mathrm{GainRatio}(S,A) = \frac{\mathrm{Gain}(S,A)}{\mathrm{SplitInformation}(S,A)}, \tag{32}$$

where Split Information is defined as:

$$\mathrm{SplitInformation}(S,A) = \sum_{i=1}^{c} \frac{|S_i|}{|S|}\log_2\frac{|S_i|}{|S|}, \tag{33}$$

where S_1 through S_c are the c subsets of examples resulting from partitioning S by the c-values attribute A. Note that *Split Information* is actually the entropy of S with respect to the values of attribute A. Notably, the metric for attribute selection as given in Eq. (32) is employed in C4.5 algorithm for better performance achievement (Quinlan, 1993).

3.2.5. Logistic regression

Logistic regression (Kokuer, Naguib, Janclovic, Younghusband, & Green, 2006) is a statistical regression model for binary dependent variables. It can be considered as a generalized linear model that utilizes the logit as its link function, and has binomially distributed errors. The model uses the form:

$$\mathrm{logit}(p) = \ln\left(\frac{p}{1-p}\right) = \alpha + \beta_1 x_{1,i} + \cdots + \beta_k x_{k,i}, \tag{34}$$

where $i = 1,\ldots,n$, and

$$p = \Pr(Y_i = 1|X) = \frac{e^{\alpha+\beta_1 x_{1,i}+\cdots+\beta_k x_{k,i}}}{1+e^{\alpha+\beta_1 x_{1,i}+\cdots+\beta_k x_{k,i}}}. \tag{35}$$

Here p is the probability of belonging to one class, $p/(1-p)$ is the odds ratio, and $\alpha,\beta_1,\ldots,\beta_n$ are regression coefficients which are most widely estimated by maximum likelihood.

Table 1
Technical indices and their formulae

Feature name	Description	Formula
OP	Open price	O_t is the open price at time t
HP	High price	H_t is the high price at time t
LP	Low price	L_t is the low price at time t
CP	Closing price	C_t is the closing price at time t
V	Volume	V_t is the volume at time t
MA6	Day moving average	$\mathrm{MA}_t = \frac{1}{6}\sum_{i=1}^{6} C_{t-i}$
MA12	Moving average	$\mathrm{MA}_t = \frac{1}{6}\sum_{i=1}^{n} C_{t-i}$
BIAS6	Bias	$\mathrm{BIAS}_t = \frac{C_t - \mathrm{MA}_t}{t}$
BIAS12	Bias	$\mathrm{BIAS}_t = \frac{C_t - \mathrm{MA}_t}{t}$
EMA12, EMA26	Exponential moving average	$\mathrm{EMA}_t = \alpha(C_t - \mathrm{EMA}_{t-1}) + \mathrm{EMA}_{t-1}$, where $\alpha = \frac{2}{t+1}$
MACD	Moving average convergence and divergence	$\mathrm{MACD}_t = \frac{2}{10}(\mathrm{DIF}_t - \mathrm{MACD}_{t-1}) + \mathrm{EMA}_{t-1}$
DIF	Difference	$\mathrm{DIF}_t = \mathrm{EMA}_{12} - \mathrm{EMA}_{26}$
%K	Stochastic %K	$\%K = \frac{C-L_t}{H_t-L_t}\times 100$
%D	%D is a 3-day moving average of %K	$\%D = \frac{H_3}{L_3}\times 100$, where $H_3 = \sum_{j=t-3}^{t}(C-L_j)$, $H_3 = \sum_{j=t-3}^{t}(H_j - L_j)$
TR	True range (TR) of price movements	$\mathrm{TR} = \max\{X,Y,Z\}$, where$X = H_t - L_t$, $Y = L_t - C_{t-1}$, $X = H_t - C_{t-1}$
MTM6, MTM12	Momentum	$\mathrm{MTM}_6 = C_t - C_{t-6}$, $\mathrm{MTM}_{12} = C_t - C_{t-12}$
OSC6	Oscillator	$\mathrm{OSC}_n = \frac{C_t}{C_{t-n}}\times 100$
OSC12	Oscillator	$\mathrm{OSC}_n = \frac{C_t}{C_{t-n}}\times 100$
%R5, %R10	Larry William's %R	$\%R_t = 100 - \frac{C-L_t}{H_t-L_t}\times 100$
OBV	On balance volume	$\mathrm{OBV}_t = \mathrm{OBV}_{t-1} + \begin{cases} V_t & \text{if } C_t > C_{t-1} \\ 0 & \text{if } C_t = C_{t-1} \\ -V_t & \text{if } C_t < C_{t-1} \end{cases}$

Remarks: C is the closing price, C_t is the closing price at time t, L_t is the low price at time t, H_t is the high price at time t, V_t is the volume at time t, D_t is the upward-day at time t, U_{v_t} is the upward-index-value at time t, and D_{V_t} is the download-index-value at time t.

Logistic regression analysis is much like linear regression in that we are interested in the relationship of a group of independent variables with a response or dependent variable. One significant difference between the logistic and linear models is that the linear model has a continuous response variable and the logistic model uses a binary or dichotomous response. As a result, the method of estimation uses maximum likelihood as opposed to least squares (Hosmer & Lemeshow, 2000).

3.3. Feature selection and classification

Around the beginning of the twentieth century, Charles Dow was believed to be the first one to attempt to place concepts such as price trends, the relationship between volume and price and even the ideas of support and resistance, at the heart of the process of analyzing the likely behavior of the price of a security. Technical analysis seeks to capture some or all of these factors in varying ways in different indicators with the aim of analyzing the historical behavior of the price of a security to determine its most likely future price. Many of the techniques are capable of being expressed in a precise mathematical formula.

In this paper, we use 23 technical indices as the whole features set and the direction of change in the daily Korea and Taiwan stock price index as the prediction target. Table 1 lists these technical indices and their definitions. 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. The total number of samples is 365 trading days, spanned from June 1990 to May 1991. The number of the training data is 294 and that of holdout data is 71. Thus near 20% of the data is used for holdout and 80% for training. The holdout data is used to test results with the data that is not utilized to build the model. We first use wrapper approach to find out the impact features for each individual classifier and use voting scheme to build prediction model, and then examine the model with the collected data set from the Taiwan and Korea stock exchange corporations.

4. Experiment result

The experiments were performed with the assistance of the Weka machine learning package (Wittena & Frank, 2005). We first compare the wrapper approach with other feature selection algorithms, including χ^2-Statistic, Information gain, ReliefF, Symmetrical uncertainty and CFS to evaluate the feature selection algorithm. The adopted prediction method is SVM. Next, we compared the voting scheme with each single classification algorithm, including SVM, KNN, BP, C4.5 DT and logistic regression to evaluate the proposed voting scheme. The wrapper approach was used to determine the feature set for each individual classifier. Tables 2–4 show the experiment results.

Table 2
Accuracy for different feature selection methods plus SVM for Korea stock trend prediction

Feature selection method	Prediction accuracy
Wrapper	67.61% (46/71)
χ^2-Statistic	40.8451% (29/71)
Information gain	49.2958% (35/71)
ReliefF	38.0282% (27/71)
Symmetrical uncertainty	49.2958% (35/71)
CFS	40.8451% (29/71)

Table 3
Comparison of prediction accuracy for different classification algorithms for Korea stock trend prediction

Classification algorithm	Prediction accuracy
Wrapper + voting	76.06% (54/71)
Wrapper + SVM	67.61% (48/71)
Wrapper + KNN	64.79% (46/71)
Wrapper + BP	69.01% (49/71)
Wrapper + C4.5 DT	64.79% (46/71)
Wrapper + logistic regression	64.79% (46/71)

Table 4
Comparison of prediction accuracy for different classification algorithms for Taiwan stock trend prediction

Classification algorithm	Prediction accuracy
Wrapper + voting	80.28% (57/71)
Wrapper + SVM	70.42% (50/71)
Wrapper + KNN	64.79% (46/71)
Wrapper + BP	66.2% (47/71)
Wrapper + C4.5 DT	71.83% (51/71)
Wrapper + logistic regression	67.61% (48/71)

In Table 2, the comparison of Korea stock trend prediction accuracy using SVM classifier along with different feature selection methods is given in Table 2. It can be seen that wrapper method indeed selects the key features for the corresponding classifier.

The comparison of Korea and Taiwan stock trend prediction by using different classifiers along with wrapper feature selection method are given in Tables 3 and 4, respectively. As expected, the proposed approach achieved the best performance. This verifies that the wrapper approach indeed can find the best feature subset with the assistance of the prediction algorithm since it examines all kinds of subset combinations from the original feature set. Meanwhile, the voting machine takes advantage of combing each classifier into consensus and thus outperforms each individual classifier.

5. Conclusion

In this paper, we show that among many feature selection algorithms, such as wrapper, χ^2-Statistic, Information gain, ReliefF, Symmetrical uncertainty and CFS, wrapper approach can find the most relevant feature from the feature set as expected. Experiment result shows that the accuracy for voting plus wrapper approach achieves the

accurate prediction rate up to 80.28%. Meanwhile, the experiment result also shows that it performs better when different classifiers are combined into the voting scheme. In the future work, we will try different combination of classifiers such as weighted voting and find other useful features besides the ordinarily used technical indices to achieve better performance in stock market trend prediction application.

Acknowledgements

The authors would like to thank the National Science Council of the Republic of China, Taiwan for financially supporting this research under Contract No. NSC 94-2213-E-026-001.

References

Atkeson, C. G., Moore, A. W., & Schaal, S. (1997). Locally weighted learning. *Artificial Intelligence Review, 11*(1–5), 11–73.

Breiman, L., Friedman, J. H., Olshen, R. A., & Stone, C. J. (1984). Classification and regression trees. Chapman and Hall/CRC.

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

Chang, Chih-Chung, & Lin, Chih-Jen (2001). *LIBSVM: A library for support vector machines*. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

Cristianini, N., & Shawe-Taylor, J. (2000). An introduction to support vector machines and other Kernel-based learning methods. Cambridge University Press.

Endou, T., & Zhao, Q. F. (2002). Generation of comprehensible decision trees through evolution of training data. In *IEEE congress on evolutionary computation*.

Haykin, S. (1994). Neural networks: A comprehensive foundation. New York: Macmillan College Publishing Company.

Hosmer, David W., & Lemeshow, Stanley (2000). Applied logistic regression (2nd ed.). Chichester, New York: Wiley.

Hughes, G. F. (1968). On the mean accuracy of statistical pattern recognizers. *IEEE Transactions on Information Theory, IT-14*(1), 55–63.

Kelly, J. D., & Davis, L. (1991). Hybridizing the genetic algorithm and the k nearest neighbors classification algorithm. In *Proceedings of the fourth international conference on genetic algorithms & applications* (pp. 377–383).

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

Kim, K., & Han, I. (2000). Genetic algorithms approach to feature discretization in artificial neural networks for the prediction of stock price index. *Expert Systems with Applications, 19*, 125–132.

Kohavi, Ron, & John, George H. (1995). Wrappers for feature subset. *Artificial Intelligence Journal, 97*(1–2), 273–324.

Kokuer, M., Naguib, R. N. G., Janclovic, P., Younghusband, H. B., & Green, R. C. (2006). Cancer risk analysis in families with hereditary nonpolyposis colorectal cancer. *IEEE Transactions on Information Technology in Biomedicine, 10*(3), 581–587.

Llorà, X., & Garrell, J. M. (2001). Evolution of decision trees. In *Proceedings of the conference on artificial intelligence*.

Medsker, L., & Liebowitz, J. (1994). Design and development of expert systems and neural networks. New York: Macmillan.

Papagelis, A., & Kalles, D. (2001). Breeding decision trees using evolutionary techniques. In *Proceedings of the international conference on machine learning*.

Peterson, M. R., Doom, T. E., & Raymer, M. L. (2005). GA-facilitated KNN classifier optimization with varying similarity measures. In *IEEE congress on evolutionary computation, Vol. 3* (pp. 2514–2521).

Quinlan, R. J. (1993). C4.5: Programs for machine learning. San Mateo, CA: Morgan Kaufman.

Russel, S., & Norvig, P. (1995). Artificial intelligence: A modern approach. Englewood Cliffs, NJ: Prentice-Hall.

Safavian, S. P., & Landgrebe, D. (1991). A survey of decision tree classifier methodology. *IEEE Transactions on Systems, Man, and Cybernetics, 21*, 660–674.

Vapnik, V. N. (1995). The nature of statistical learning theoary. Springer.

Vapnik, V. N. (1998). Statistical learning theory. New York: Wiley.

Witten, Ian H., & Frank, Eibe (2005). Data mining: Practical machine learning tools and techniques. *2nd ed.*. San Francisco: Morgan Kaufman.