Expert Systems
The Journal of Knowledge Engineering

Article

DOI: 10.1111/j.1468-0394.2011.00581.x

Lag-ℓ forecasting and machine-learning algorithms

Jae Joon Ahn,[1] Il Suh Son,[1] Kyong Joo Oh,[1]
Tae Yoon Kim[2] and Gyu Moon Song[2]

(1) Department of Information and Industrial Engineering, Yonsei University, South Korea
Email: johanoh@yonsei.ac.kr
(2) Department of Statistics, Keimyung University, South Korea

Abstract: *In this study, we discuss the problem of lag-ℓ forecasting, which has been solved using a machine-learning algorithm. The main aim of this study is to define the lag-ℓ forecaster based on a precise classification approach and to discuss the technical issues involved in the lag-ℓ forecasting problem, including a comparison of various machine-learning algorithms for proper implementation of the technique. This study focuses on an application that uses the lag-ℓ forecaster in an early-warning system.*

Keywords: lag-ℓ forecasting classifier, machine-learning algorithms, early warning

1. Introduction

Machine learning has been extensively used for forecasting the behaviour of systems in various areas. The basic idea of this approach is to map the relationship between predictors and response variables through the use of training datasets and to extend the relationship to forecasting. In applications where the response variable is discrete or categorical, forecasting must be performed through the identification of a suitable classifier from a training dataset (Berry & Linoff, 1997; Shmueli *et al*., 2006). In this study our primary concern is the problem of forecasting performed via early labelling of the future condition using the lag-ℓ forecaster, and a machine-learning algorithm is used as the lag-ℓ forecasting classifier. A simple example of this problem is the early forecasting of the variation in the stock market over a given number of days ℓ using an artificial neural network (ANN) (Poddig & Rehkugler, 1996; Tsaih *et al*., 1998).

The lag-ℓ forecasting classifier is popular in several important problems, particularly in building early-warning systems such as those used to issue financial crisis warnings ℓ months ahead of the event through the labelling of future financial market conditions (Oh *et al*., 2006; Kim *et al*., 2009; Son *et al*., 2009). Another example is the ℓ-week ahead warning system used to predict the possibility of heavy rainfall in the tropical regions that suffer from seasonal heavy rain. In this case, the heavy rainfall is labelled as a potentially monitoring weather condition to use the lag-ℓ forecasting classifier (Hong, 2008). Other applications of the lag-ℓ forecasting classifier include early warning against ambient air pollution, wind wave changes, electric load changes, risk management and variations in the price of crude oil (Deo & Naidu, 1998; Osowski & Garanty, 2007; Yu *et al*., 2008; Hong, 2009; Wu & Olson, 2009; Chen *et al*., 2010; Wu *et al*., 2010).

Although the lag-ℓ forecasting classifier is popular in numerous important applications, it has, until now, been considered a simple and typical classification technique (Deo & Naidu, 1998; Osowski & Garanty, 2007; Hong, 2008, 2009; Yu *et al.*, 2008). However, the lag-ℓ forecasting classifier must be specially considered because its model can be uniquely defined using the additional key parameter ℓ and the oracle lag-zero predictor. The primary purpose of this study is twofold. First, we provide a precise description of the lag-ℓ forecasting classification problem. Second, we consider various machine-learning algorithms (or classifiers) to understand the technical issues involved in lag-ℓ forecasting. The primary application of interest in this manuscript is the early-warning problem applied to the Korean stock market. In particular, our primary results have been addressed through empirical experiments in which the lag-ℓ forecasting classifier is used to develop an early-warning system to predict the possibility of a massive pullout by *global institutional investors* (GII) off the Korean stock market.

The remaining sections of the manuscript are organized as follows. Section 2 introduces a precise description of the lag-ℓ forecasting classifier, and Section 3 discusses the various technical issues involved, including a description of the selection of a proper machine-learning algorithm for the lag-ℓ forecasting classifier. Section 4 provides empirical experiments for Section 3. Finally, concluding remarks are provided in Section 5.

2. The lag-ℓ forecasting classifier

Let us assume that there exists the oracle lag-zero rule

$$f_0 : \mathbf{Z} \to \mathrm{Y} \tag{1}$$

which maps the oracle predictor $Z=(Z_1,\ldots,Z_q)\in\mathbf{Z}$ onto its classification label $Y\in\mathrm{Y}$. Consider the problem of forecasting y_{t+l} $(l>0)$ at time t by the predictor variable $\mathbf{x}_t$, which is different from $\mathbf{z}_{t+1}$. For this problem, we introduce the lag-ℓ forecasting or classification model, which can be defined as

$$Y_{t+\ell}=f_\ell(X_{1t},\ldots,X_{pt}) \tag{2}$$

where f_ℓ is a lag-ℓ classifier with a set of predictor variables $X=(X_{1t},\ldots,X_{pt})$ and a discrete (or categorical) response variable $Y_{t+\ell}$. For the model described in equation (2), the training dataset of size n may be expressed as

$$\Xi_n=\{(x_{11},\ \ldots\ ,\ x_{p1},\ y_{1+\ell}),\ \ldots\ ,\ (x_{1n},\ \ldots\ ,\ x_{pn},\ y_{n+\ell})\} \tag{3}$$

where $y_t=(z_{1t},\ldots,z_{pt})$ from equation (1). The training dataset Ξ_n then produces the lag-ℓ forecasting classifier

$$\hat{f}_\ell : \mathbf{X} \to \mathrm{Y} \tag{4}$$

which maps $\{\mathrm{x}_t=(x_{1t},\ldots,x_{pt}):t=1,\ldots,n\}$ onto its classification label $\{(y_{1+\ell},\ldots,y_{n+\ell})\}$.

Remark 1 The oracle model in equation (1) assumes that the unknown rule f_0 defines $Y\in\mathrm{Y}$ in terms of the predictor $Z\in\mathbf{Z}$ 'without lag'. In other words, to determine the value of y_t for a given value of t, it is essential to wait until time t and first determine the value of $\mathbf{z}_t=(z_{1t},\ldots,z_{qt})$. The model in equation (1) has been introduced to emphasize that the predictor Z of the oracle rule, which is different from the predictor for the model in equation (2), plays a key role in the working of the lag-ℓ forecasting classifier. Sometimes it is easy to determine Z. For instance, in the example of rise and fall forecasting of the stock market, $Z(=Y)$ equals one or zero depending on whether the price of stocks has risen or fallen, whereas X is the stock price and the trading volume of the previous day. However, it is usually difficult to determine Z, especially for early-warning forecasters. For instance, determining the value of Z for a financial crisis situation (or obtaining a set of variables that define financial crisis accurately) is very difficult (Oh *et al.*, 2006; Oh & Kim, 2007; Kim *et al.*, 2009).

Remark 2 The model in equation (2) assumes that forecasting $y_{t+\ell}$ $(\ell>0)$ at time t is possible $\mathbf{x}_t=(x_{1t},\ldots,x_{pt})$. The basic assumption of the

model in equation (2) is of course stationary (or time-invariant) relation of f_ℓ. Empirically some technical tips for achieving the stationary relation with training data may be considered. First, because $\hat{f}_\ell$ is trained using Ξ_n so that it can accurately forecast the label of Y defined by f_0 after lag-ℓ, it is necessary to include these variables, which mimic the oracle predictor Z as a subset of X. Second, the normalization or transformation of X (e.g. the linear transformation of the predictor variables from the training data to $[0, 1]$) may be effective in preventing extremely large or small values of the predictor variables from affecting $\hat{f}_\ell$ significantly. It is a known fact that that transformation within a specified range helps to reduce forecasting error by inducing a stationary relationship between $(X_{1t}, \ldots, X_{pt})$ and $Y_{t+\ell}$ (Lapedes & Farber, 1998). Although we normally assume that $Z \neq X$, this may not be true if a stationary or time-invariant relationship exists between Z_t and $Z_{t+\ell}$. Such a case, however, is not of concern in this study, and it may be handled by establishing a time series stationary model for the relationship between Z_t and $Z_{t+\ell}$.

3. Machine-learning algorithms

For efficient training of $\hat{f}_\ell$, we consider the following four popularly used classification methods: ANN, multi-nomial logistic regression (MLR), decision tree (DT) and case-based reasoning (CBR). Our experiments with these machine-learning algorithms address various technical issues including the selection of a proper machine-learning algorithm for $\hat{f}_\ell$. Among the four algorithms mentioned above, MLR is parametric whereas the remaining three are non-parametric. Out of the three non-parametric algorithms, DT is the closest to a parametric algorithm.

3.1. ANN

ANN has gained popularity in a variety of applications (Burrell & Folarin, 1997; Zhang *et al.*, 1998; Oh & Kim, 2002; Frutos *et al.*, 2003; Oh *et al.*, 2006; Kim *et al.*, 2009). In particular, ANN is a universal-function approximator that can map a non-linear function better than most other statistical machine-learning methods so that it is less sensitive to error-term assumptions and can tolerate noise, chaotic characteristics and heavy tails (White, 1989; Kaastra & Boyd, 1996). In our empirical study, we have used a three-layer fully connected back-propagation neural network (BPN), where the three layers correspond to an input layer, a hidden layer and an output layer. As the algorithm's name implies, the errors (and therefore the learning process) propagate backwards from the output nodes to the input nodes. Technically, back-propagation is used to calculate the gradient of error of the network with respect to the network's modifiable weights. This gradient is then almost always used in a simple *stochastic gradient descent algorithm* to determine weights that minimize the error. For the hidden layer to serve any useful function, the BPN must also have *non-linear activation functions* for each of its layers. Successful implementation of BPN requires a specific configuration of both the non-linear activation function and the stochastic gradient descent algorithm (i.e. learning rate, momentum and initial weight)

3.2. MLR

Logistic regression is one of the statistical methods used to estimate the probability of a binary outcome with an upward or downward status (Hosmer & Lemeshow, 1989). The MLR model generalizes logistic regression by allowing more than two discrete outcomes for an experiment. As a result, it is well suited in forecasting problems with a categorical output variable. The MLR model assumes the following *parametric function* model:

$$P(Y_{t+\ell} = 0) = 1/[1 + \sum_{j=1}^{J} \exp(X_t \beta_j)] \quad (5)$$

$$P(Y_{t+\ell} = j) = \exp(X_t \beta_j) / \left[1 + \sum_{j=1}^{J} \exp(X_t \beta_j)\right] \quad (6)$$

where X_t is a vector of predictor variables at time t and $J+1$ is the number of outcomes of the response variable. The unknown parameters β_j are typically estimated using maximum likelihood methods.

3.3. DT

DT creates a model that predicts the value of a response variable based on several input variables. Each interior node corresponds to one input variable, and there are branches to children for each possible value of that input variable. Each leaf represents a value of the response variable based on the values of the input variables that are represented by the path from the root to the leaf. A tree can be 'learned' by splitting the source-set into subsets based on an attribute value test. This process is repeated for each derived subset in a recursive manner known as *recursive partitioning*. The recursion is considered complete when splitting does not add value to the predictions any more. In our empirical study, we use the standard CART algorithms described by Breiman *et al.* (1984), which are designed for both real-valued and integer-valued responses. Our CART algorithms use the Gini diversity index for the splitting of the tree nodes and the cross-validation technique for the pruning of the trees. Note that the Gini diversity index plays a key role in choosing a variable at each step and determines the next best variable for the splitting of the set of items.

3.4. CBR

CBR is a problem-solving method in which *cases and experiences* are reused to find an appropriate solution to a given new case (Lee, 2007). The nearest neighbour approach is normally used for retrieving cases that are most similar to the case at hand. The following three factors are crucial in the implementation of CBR: distance function, combination function and number of neighbours (Shin & Han, 1999). Successful implementation of CBR requires a specific configuration of these factors (Kolodner, 1991). Our empirical study uses the Euclidean distance as the distance function and the equally weighted voting method as the combination function. In addition, experiments have been conducted to determine the optimum number of neighbours. An exemplified reference is provided in Table 10.

Remark 3 MLR and DT exhibit certain limitations for a (relatively) large value of ℓ. MLR assumes the parametric model given in equations (5) and (6). The equation reduces to $P(Y_{t+\ell}=j)=1/(1+J)$ for $j=0,1,\ldots,J$ if $\beta_j=0$. Note that $\beta_j=0$ implies independence between the response and predictor variables. Such independence is expected with large values of ℓ. Therefore, as the value of ℓ increases, the distribution of Y in the MLR model approaches a uniform distribution ℓ. Thus, one may expect the MLR model to fail for large values of ℓ unless the real values of Y follow an exactly same uniform distribution. DT appears to exhibit a similar limitation because it produces a DT with a structure that determines a specific distribution of Y when independence is assumed between the response and predictor variables.

Remark 4 Performance of $\hat{f}_\ell$ depends on a combination of the oracle rule f_0 and the training dataset Ξ_n. The performance of $\hat{f}_\ell$ may be considerably better when an easily (not easily) classifiable or separable dataset Ξ_n meets a conservative (sensitive) rule f_0, where a conservative (sensitive) rule f_0 implies a stricter (looser) rule for the definition of the response labels. However, this basic intuitive tendency disappears for extremely large values of ℓ. This mechanism will be illustrated in Section 4 (refer to Table 1).

4. Empirical experiments

In recent work, Son *et al.* (2009) have developed an *early-warning system for monitoring the behaviour of global institutional investors* (EWSGII) based on the lag-ℓ classification approach, which consists of constructing two classifiers f_0

Table 1: *Lag-zero classification rules for market condition*

	Classification rule
$f_0(1)$	**If** Quarterly net sale more than 2.4* (or) monthly net sale more than 1.6* (or) weekly net sale more than 0.8* (or) daily net sale more than 0.4*, **Then** $y = 3$ **(CP)** **Else If** Quarterly net sale more than 1.2* (or) monthly net sale more than 0.8* (or) weekly net sale more than 0.4* (or) daily net sale more than 0.15*, **Then** $y = 2$ **(TP)** **Else** $y = 1$ **(SP)**
$f_0(2)$	**If** Quarterly net sale more than 3.0* (or) monthly net sale more than 2.0* (or) weekly net sale more than 1.0* (or) daily net sale more than 0.5*, **Then** $y = 3$ **(CP)** **Else If** Quarterly net sale more than 1.5* (or) monthly net sale more than 1.0* (or) weekly net sale more than 0.5* (or) daily net sale more than 0.2*, **Then** $y = 2$ **(TP)** **Else** $y = 1$ **(SP)**
$f_0(3)$	**If** Quarterly net sale more than 4.0* (or) monthly net sale more than 3.0* (or) weekly net sale more than 1.5* (or) daily net sale more than 0.7*, **Then** $y = 3$ **(CP)** **Else If** Quarterly net sale more than 2.0* (or) monthly net sale more than 1.5* (or) weekly net sale more than 0.8* (or) daily net sale more than 0.4*, **Then** $y = 2$ **(TP)** **Else** $y = 1$ **(SP)**

*Unit: 1 billion won.

and $\hat{f}_\ell$. This approach is required for EWSGII because $\hat{f}_\ell$ is designed to predict future market conditions that are predetermined by f_0.

To build a lag-ℓ forecasting classifier for this application, we first obtain the oracle lag-zero rule f_0 that classifies the current market condition into one of the following three categories: the stable period (SP), the transition period (TP or the grey zone) and the crisis period (CP). In the development process of f_0, we assume that CP is the period during which the GII sell enormous quantities of stocks with a contingency plan, and TP is the period in which the GII change from a net long position (i.e. buying trend) to a net short position (i.e. selling trend) to initiate the contingency plan. With these assumptions, SP, TP and CP can be defined in terms of the quarterly, monthly, weekly and daily net sales by the GII. These four quantities are known to accurately reflect the selling trend with a contingency plan. In fact, the four quantities are constantly monitored by the Financial Supervisory Service in Korea, which uses its own rule of thumb in determining the quantities corresponding to abnormal massive selling by the GII. Let the four variables $Z_1, \ldots, Z_4$ correspond to the oracle predictor variables for f_0 and Y be the response variable that assumes values 1, 2 and 3 corresponding to SP, TP and CP, respectively. Under these assumptions f_0 is considered well defined. We use three different types of f_0, that is, $f_0(1)$, $f_0(2)$ and $f_0(3)$ (see Table 1). It should be noted that compared with $f_0(2)$, $f_0(1)$ tends to produce more CPs (i.e. it is a sensitive classifier) whereas $f_0(3)$ tends to produce few CPs (i.e. it is a conservative classifier). It should also be noted that the oracle rule $f_0(2)$ is used by the Financial Supervisory Service in Korea, and $f_0(1)$ and $f_0(3)$ have been considered for comparison purpose.

Next we build the lag-ℓ forecasting classifier $\hat{f}_\ell$ as follows. We choose 18 variables $X_1, \ldots, X_{18}$ as the input variables for $\hat{f}_\ell$. These variable include the *daily net buying position of the GII* (NPG), the *Korean stock price index* (SPI), the *Korean won/US dollar exchange rate* (FER), the *Dow-Jones index* (DJI), and the *daily net long position of index futures by the GII* (NPI), each with its own derivative variables (see Tables 2 and 3). Here NPG (NPI) represents the difference of the daily long position of the GII (index futures by the GII) and the daily short position of GII (index futures by the GII). A detailed discussion of the selection of input variables may be found in Son *et al.* (2009).

Training sets are constructed for each lag-ℓ forecasting classifier $\hat{f}_\ell$ $(\ell = 1, 5, 20, 60)$. The $\hat{f}_{60}$ classifier (i.e. the quarterly classifier) is included here to determine the behaviour of the lag-ℓ classifiers for large values of ℓ where the assumption of

Table 2: *List of input and output variables for EWSGII*

Variable name	*Input variables*	*Output variable*
NPG	IND, MA(10), MA(20), MA(60), MV(20)	y (condition of market)
SPI	IND, MA(10), MA(20), MV(20)	
FER	IND, MA(20), MV(20)	
DJI	IND, MA(20), MV(20)	
NPI	IND, MA(10), MA(20)	

Selected input variables consist of a total of the 18 variables shown. The output variable y denotes the condition of the market (1 = SP, 2 = TP, 3 = CP).

Table 3: *Legend for Table 2*

Variable name	*Numerical formula*	*Description*
IND	x_t	Index or rate
MA(m)	$\bar{p}_{m,t} = \sum_{i=t-(m-1)}^{t} x_t/m$	m-day moving average
MV(m)	$s^2_{m,t} = \sum_{i=t-(m-1)} (x_i - \bar{p}_{m,t})^2/m$	m-day moving variance

a nearly independent stationary relationship holds true. For the training of $\hat{f}_\ell$, two different types of enormous selling periods (ESP) have been taken into account, that is, July 1999–September 1999 and February 2002–April 2002. The ESP in 1999 (ESP_{99}) corresponds to a period when the market was on the fast track to overheat, whereas the ESP in 2002 (ESP_{02}) corresponds to an extended period of continuous downslide in the market. Thus, ESP_{99} is against the normal market trend and ESP_{02} is in agreement with the normal market trend. In this regard, ESP_{99} provides a more easily classifiable training period (and therefore, training datasets) than ESP_{02}. For each period ESP_{99} and ESP_{02}, we have built three training datasets by applying the three different lag-zero rules, that is, $f_0(1)$, $f_0(2)$ and $f_0(3)$. This provides a total of six possible training datasets. Please note that although six datasets are possible in theory, only *five* separate training datasets are available in this study because of technical reasons due to which $f_0(3)$ failed to yield a training dataset for 2002. In short, we have built *five* separate training datasets for each lag-ℓ forecasting classifier $\hat{f}_\ell$ ($\ell = 1, 5, 20, 60$) (see Tables 4–7). In addition, it should be noted that the three lag-zero rules $f_0(1)$, $f_0(2)$ and $f_0(3)$ are applied to the entire period (i.e. January 1999–April 2004), which produces $y \in Y$ for the entire training and test dataset (Table 8).

Five training datasets (or five equivalent classifiers $\hat{f}_\ell$) are established for each value of ℓ, that is, $\ell = 1$, 5, 20, 60, to provide a total of 20 classifiers. Each of the 20 resulting classifiers is then trained and tested using the MLR, DT, CBR and ANN techniques. As explained in Section 3, implementation of each machine-learning technique requires a specific configuration of the corresponding algorithm. Hit rates, which are given by the number of correct forecasts divided by the number of total forecasts, are often used to find such configurations. Specific configurations and analysis tools used for each learning algorithm are listed in Table 9. An illustration of the use of hit rates for obtaining a desirable configuration is provided here using ANN and CBR. For ANN, some trial-and-error experiments using hit rates have been conducted to build an appropriate configuration of BPN. As a result, a three-layer connected BPN has been used with the following parameters: logistic activation function, learning rate = 0.1, momentum = 0.1, initial weight = 0.1 and six hidden neurons. To prevent over-fitting, the validation set is randomly assigned to 25% of the total training set. Training of the ANN is stopped when the hit rate of the validation set ceases to show improvement. In this case, the number of training epochs is limited to 100000

Table 4: *Training period for f_1 when the lag-zero rules are applied separately for the ESP in 1999 and 2002 (yy/mm/dd)*

Year	*Lag-zero rule*	*SP*	*TP*	*CP*
1999	$f_0(1)$	99/06/17–99/07/08	99/07/09–99/07/30	99/08/02–99/08/23
	$f_0(2)$	99/06/18–99/07/09	99/07/12–99/08/02	99/08/03–99/08/24
	$f_0(3)$	99/06/23–99/07/23	99/07/26–99/09/02	99/09/03–99/10/07
2002	$f_0(1)$	02/01/28–02/02/20	02/02/21–02/02/26	02/03/15–02/03/15
		02/02/27–02/03/14	02/03/18–02/04/03	02/04/04–02/04/15
			02/04/16–02/04/26	02/04/29–02/05/23
	$f_0(2)$	02/01/25–02/02/25	02/02/26–02/02/27	02/04/10–02/04/12
		02/02/28–02/03/13	02/03/14–02/04/09	02/05/07–02/06/11
			02/04/15–02/05/06	

SP, stable period; TP, transition period; CP crisis period.

Table 5: *Training period for f_5 when the lag-zero rules are applied separately for the ESP in 1999 and 2002 (yy/mm/dd)*

Year	*Lag-zero rule*	*SP*	*TP*	*CP*
1999	$f_0(1)$	99/06/11–99/07/02	99/07/05–99/07/26	99/07/27–99/08/17
	$f_0(2)$	99/06/14–99/07/05	99/07/06–99/07/27	99/07/28–99/08/18
	$f_0(3)$	99/06/17–99/07/19	99/07/20–99/08/26	99/08/30–99/10/01
2002	$f_0(1)$	02/01/22–02/02/14	02/02/15–02/02/20	02/03/11–02/03/11
		02/02/21 ~ 02/03/08	02/03/12–02/03/28	02/03/29 ~ 02/04/09
			02/04/10–02/04/22	02/04/23–02/05/17
	$f_0(2)$	02/01/21–02/02/18	02/02/19–02/02/20	02/04/03–02/04/08
		02/02/21–02/03/07	02/03/08–02/04/02	02/04/30–02/06/04
			02/04/09–02/04/29	

SP, stable period; TP, transition period; CP crisis period.

in this case. The use of hit rates for selecting the number of neighbours for CBR is illustrated in Table 10. It is well known that the movements of average hit rate are influenced by different k values (Li & Wu, 2010).

After obtaining specific configurations for MLR, DT, CBR and ANN, hit rates are used to identify the most suitable machine-learning algorithm for lag-ℓ forecasting among MLR, DT, CBR and ANN. As shown in Tables 11–14, hit rates are recorded for the 20 classifiers using one of the above-mentioned methods, that is, MLR, DT, CBR and ANN. Examination of Tables 11–14 along with Figure 1 indicates that the performance of MLR and DT become noticeably poor at $\ell = 20$, whereas ANN and CBR work reasonably well at $\ell = 20$ (see the testing summaries shown in Tables 11–14, that is, the last row of the tables). It may also be noticed that DT outperforms the remaining methods for $\hat{f}_1$, and $\ell = 60$ prefers non-parametric training to parametric training (refer to Figure 1). This observation is consistent with Remark 3, which states that DT and MLR perform poorly for large values of ℓ unless the real distribution of Y with values of 1, 2 or 3 is not equal to the specific distribution of Y that is induced by MLR and DT with large ℓ values. For instance, the specific distribution of Y that is induced by MLR is uniform for $Y = 1, 2, 3$ but the real distribution of Y is far from uniform because $Y = 2$ or 3 are relatively rare in reality.

From Figure 1, one may also notice that on average CBR appears to perform better across

Table 6: *Training period for f_{20} when the lag-zero rules are applied separately for the ESP in 1999 and 2002 (yy/mm/dd)*

Year	*Lag-zero rule*	*SP*	*TP*	*CP*
1999	$f_0(1)$	99/05/21–99/06/11	99/06/14–99/07/05	99/07/06–99/07/27
	$f_0(2)$	99/05/24–99/06/14	99/06/15–99/07/06	99/07/07–99/07/28
	$f_0(3)$	99/05/27–99/06/28	99/06/29–99/08/06	99/08/09–99/09/08
2002	$f_0(1)$	01/12/28–02/01/21	02/01/22–02/01/25	02/02/15–02/02/15
		02/01/28–02/02/14	02/02/18–02/03/07	02/03/08–02/03/18
			02/03/19–02/03/29	02/04/01–02/04/25
	$f_0(2)$	01/12/27–02/01/23	02/01/24–02/01/25	02/03/13–02/03/15
		01/01/28–02/02/08	02/02/14–02/03/08	02/04/09–02/05/14
			02/03/18–02/04/08	

SP, stable period; TP, transition period; CP crisis period.

Table 7: *Training period for f_{60} when the lag-zero rules are applied separately for the ESP in 1999 and 2002 (yy/mm/dd)*

Year	*Lag-zero rule*	*SP*	*TP*	*CP*
1999	$f_0(1)$	99/03/24–99/04/15	99/04/16–99/05/10	99/05/11–99/06/01
	$f_0(2)$	99/03/25–99/04/16	99/04/19–99/05/11	99/05/12–99/06/02
	$f_0(3)$	99/03/30–99/04/30	99/05/03–99/06/03	99/06/04–99/07/06
2002	$f_0(1)$	01/11/01–01/11/21	01/11/22–01/11/27	01/12/13–01/12/13
		01/11/28–01/12/12	01/12/14–02/01/04	02/01/07–02/01/15
			02/01/16–02/01/28	02/01/29–02/02/26
	$f_0(2)$	01/10/31–01/11/23	01/11/26–01/11/27	02/01/04–02/01/08
		01/11/28–01/12/11	01/12/12–02/01/03	02/01/25–02/03/08
			02/01/09–02/01/24	

SP, stable period; TP, transition period; CP crisis period.

the range of ℓ values. It should be noted that the performances decrease in the order of CBR, ANN, DT, MLR and the order is true even for $\ell = 60$ although the performances at $\ell = 60$ are relatively moderate (i.e. the hit rates for $\ell = 60$ are approximately 55% across the four machine-learning algorithms).

The two observations presented above may also be verified using statistical hypothesis testing. A one-way ANOVA test has been performed for the following null hypothesis, that is, H_0: There is no difference between the three types of classifiers (daily, weekly and monthly) for each of the machine-learning algorithms under consideration (ANN, MLR, DT and CBR). This yields four testing results for H_0. We do not consider the quarterly classifier (i.e. lag-ℓ classifier with $\ell = 60$) for testing of H_0, because the quarterly classifier has been introduced mainly to explain the independent stationary relationship in our empirical experiments, not for practical use. The results of the hypothesis testing (i.e. *P*-values) are provided in

Table 8: *The number of data for training and testing of the lag-ℓ classifier*

Dataset	*99(1)*[a]	*99(2)*	*99(3)*	*02(1)*	*02(2)*
Training	48	48	69	78	84
Testing	1254	1254	1227	1224	1211
Total	1302	1302	1296	1302	1295

[a]99(*j*) indicates that training and testing data are obtained from the application of the lag-zero rule $f_0(j)$ to the EPS of 1999 and to the EPS of 1999.1–2004.4 minus the EPS of 1999. 02(*j*) indicates a similar process performed for ESP of 2002.

Table 9: *Configuration and computational tools used for each learning algorithm*

Learning algorithm	*Configuration*	*Computational tool*
MLR	Type: multi-nomial Link function: logit	SAS enterprise miner v9.1
DT	Algorithm: standard CART algorithm Split criterion: Gini diversity index	SAS enterprise miner v9.1
CBR	Algorithm: Rd-tree algorithm Number of neighbors: 5	SAS enterprise miner v9.1
ANN	Algorithm: three-layered BPN Activation function: logistic function Learning rate: 0.1 Momentum: 0.1 Initial weight: 0.3 Number of hidden nodes: 6	NeuroShell2 v4.0

MLR, multi-nomial logistic regression; DT, decision tree; CBR, case-based reasoning; ANN, artificial neural network.

Table 10: *Selecting the number of neighbors (k) for case-based reasoning (CBR) based on hit rates*

	$\hat{f}_1$ *(daily)*	$\hat{f}_5$ *(weekly)*	$\hat{f}_{20}$ *(monthly)*	$\hat{f}_{60}$ *(quarterly)*	*Mean*
$k = 5$[a]	95.8	89.6	91.7	93.8	92.725
$k = 6$	95.8	89.6	91.7	91.7	92.2
$k = 7$	91.7	85.4	91.7	93.8	90.65
$k = 8$	91.7	89.6	89.6	95.8	91.675
$k = 9$	91.7	85.4	83.3	95.8	89.05
$k = 10$	87.5	83.3	87.5	93.8	88.025

[a]$k = 5$ with highest hit rate is selected.

Table 11: *Training and testing hit rates (%) by multi-nomial logistic regression (MLR)*

	Datasets	$\hat{f}_1$ *(daily)*	$\hat{f}_5$ *(weekly)*	$\hat{f}_{20}$ *(monthly)*	$\hat{f}_{60}$ *(quarterly)*
99(1)	Training	100	97.9	100	100
	Testing	87.7	81.5	64.5	58.2
99(2)	Training	100	95.9	100	100
	Testing	91.1	80.5	68.0	62.5
99(3)	Training	76.8	94.2	100	100
	Testing	69.9	92.4	72.7	56.5
02(1)	Training	94.9	90.0	97.4	100
	Testing	64.3	79.1	71.9	40.9
02(2)	Training	95.2	89.3	97.6	100
	Testing	84.3	55.7	54.4	40.6
Summary of testing	Mean	79.5	77.8	66.3	51.7
	Standard deviation	11.7	13.5	7.4	10.2

Table 15. One may observe that small *P*-values are generated for MLR and DT whereas relatively large *P*-values are generated for ANN and CBR. Given that small *P*-values indicate a tendency against the null hypothesis (i.e. rejection of the null hypothesis), our test results indicate that parametric tools are more sensitive to lag size ℓ than non-parametric tools (as expected). Furthermore, taking Figure 1 into account, it appears that parametric methods

Table 12: *Training and testing hit rates (%) by decision tree (DT)*

	Datasets	$\hat{f}_1$ *(daily)*	$\hat{f}_5$ *(weekly)*	$\hat{f}_{20}$ *(monthly)*	$\hat{f}_{60}$ *(quarterly)*
99(1)	Training	100	100	100	100
	Testing	87.6	74.2	67.6	59.4
99(2)	Training	100	100	100	100
	Testing	91.2	73.4	67.5	57.0
99(3)	Training	95.6	95.6	98.5	100
	Testing	98.3	88.4	72.5	53.3
02(1)	Training	94.9	91.5	92.3	91.0
	Testing	87.0	52.5	61.0	44.9
02(2)	Training	91.6	94	92.9	95.2
	Testing	75.6	77.0	34.9	45.6
Summary of testing	Mean	87.9	73.1	60.7	52.0
	Standard deviation	8.2	13.0	15.0	6.5

Table 13: *Training and testing hit rates (%) by case-based reasoning (CBR)*

	Datasets	$\hat{f}_1$ *(daily)*	$\hat{f}_5$ *(weekly)*	$\hat{f}_{20}$ *(monthly)*	$\hat{f}_{60}$ *(quarterly)*
99(1)	Training	96.0	91.7	91.7	95.8
	Testing	85.0	85.0	71.0	63.5
99(2)	Training	91.7	93.7	91.7	100
	Testing	88.0	88.0	75.0	61.1
99(3)	Training	91.3	94.2	92.8	95.7
	Testing	91.0	91.0	85.0	58.6
02(1)	Training	89.7	84.6	89.9	92.3
	Testing	81.0	82.0	80.6	54.7
02(2)	Training	92.9	91.7	82.1	94.1
	Testing	60.0	74.0	82.4	49.8
Summary of testing set	Mean	81.0	84.0	78.8	57.5
	Standard deviation	12.3	6.5	5.7	5.4

work better for smaller values of ℓ whereas non-parametric machine-learning tools work well for ℓ overall (i.e. they work more efficiently for larger values of ℓ). The largest P-value is observed for CBR with a high average hit rate, which indicates that CBR is a robust tool with reasonably good accuracy.

Regarding Remark 4, it can be noticed from Tables 11–14 that a better performance of $\hat{f}_\ell$ is observed for each given value of $\ell = 1, 5, 20$, when an easily (not easily) classifiable or separable dataset Ξ meets a conservative (sensitive) rule f_0. However, this tendency is valid only for $\ell = 1, 5, 20$. For $\ell = 60$, the tendency is no longer visible, which suggests that the (nearly) independent stationary relationship may hinder this basic intuitive property. Recall that 99(1), 99(2) and 99(3) represent the cases where an easily separable training dataset called ESP_{99} meets the three types of f_0 rules $f_0(1)$, $f_0(2)$, $f_0(3)$, whereas 02(1) and 02(2) represent the cases where a training dataset that is not easily separable, called ESP_{02}, meets two types of the f_0 rules $f_0(1)$ and $f_0(2)$. It may be recalled that $f_0(j)$ becomes more conservative in the order of $j = 1, 2, 3$.

5. Concluding remarks

In this study we define the lag-ℓ forecasting classification problem formally with the aid of the oracle rule and the parameter lag-ℓ. Our major findings are as follows: (i) a parametric machine-learning algorithm such as MLR and

Table 14: *Training and testing hit rates (%) by artificial neural network (ANN)*

	Datasets	$\hat{f}_1$ *(daily)*	$\hat{f}_5$ *(weekly)*	$\hat{f}_{20}$ *(monthly)*	$\hat{f}_{60}$ *(quarterly)*
99(1)	Training	97.9	97.9	100	97.9
	Testing	83.8	78.5	75.6	60.6
99(2)	Training	95.8	83.3	97.9	97.9
	Testing	83.4	86.0	83.1	61.1
99(3)	Training	97.1	95.6	100	100
	Testing	62.6	89.7	83.5	56.3
02(1)	Training	85.9	83.3	92.3	92.3
	Testing	81.9	72.8	72.2	51.8
02(2)	Training	88.0	86.9	90.5	90.3
	Testing	50.6	74.2	60.5	52.9
Summary of testing	Mean	72.5	80.2	75.0	56.5
	Standard deviation	15.1	7.4	9.4	4.2

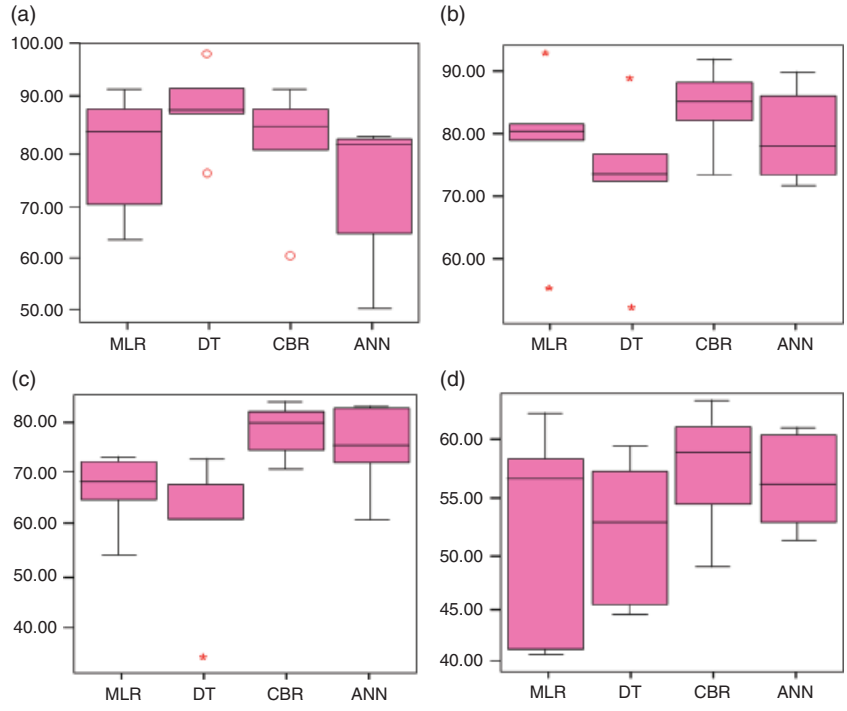

Figure 1: *Box plot of hit rates (%) for each classifier in the case of (a) daily forecasting* $\hat{f}_1$*, (b) weekly forecasting* $\hat{f}_5$*, (c) monthly forecasting* $\hat{f}_{20}$ *and (d) quarterly forecasting* $\hat{f}_{60}$*.*

DT may be more effective for short-term forecasting (or small values of ℓ), whereas a non-parametric machine-learning algorithm such as ANN or CBR may be more effective for long-term forecasting (or large values of ℓ); (ii) the oracle rule may influence the performance of the lag-ℓ forecasting classifier. Our empirical experiment, which establishes an early-warning system for financial market crisis, confirms these findings.

The finding that CBR performs better than other machine-learning algorithms across

Table 15: *One-way ANOVA testing results for H_0: there is no difference between the three types of classifiers ($\ell = 1, 5, 20$) for each algorithm*

Machine-learning algorithm	*P-value*
MLR	0.169
DT	0.015
CBR	0.647
ANN	0.550

MLR, multi-nomial logistic regression; DT, decision tree; CBR, case-based reasoning; ANN, artificial neural network.

different values of ℓ in our experiment is interesting. This may be because CBR is useful when knowledge of the sample conditions is incomplete or when the sample is sparse (Kolodner, 1991). Sparse sampling is intrinsic to crisis-related data because crisis itself is an uncommon occurrence. In this sense, CBR appears to be a desirable choice for building early-warning systems for prediction of a rare crisis. Recently, the sample-sparseness aspect of crisis has been resolved by minutely examining the rare occurrence of crisis from several different aspects, which inevitably increases the number of available inputs or predictor variables. In fact, this type of problem is known as a high-dimension low sample-size (HDLSS) classification problem (Hall *et al.*, 2005). Therefore, a future study to find a proper machine-learning tool for efficient HDLSS lag-ℓ forecasting may be of interest. An additional possible future study that may be of interest is to use receiver operating characteristic (ROC) curves as a major error metric in place of the hit rate. This may of particular interest because ROC curves are independent of the classification costs of assigning false positives or negatives.

Acknowledgements

T. Y. Kim's work was supported by Basic Science Research Programme through the National Research Foundation of Korea funded by the Ministry of Education, Science and Technology (2009-0073872). G. M. Song's work was supported by a grant (no. RT104-01-01) from the Regional Technology Innovation Programme of the Ministry of Knowledge Economy of Korea.

References

BERRY, M. and G. LINOFF (1997) *Data Mining Techniques*, New York: Wiley.

BREIMAN, L., J.H. FRIEDMAN, R.A. OLSHEN and J.S. CHARLES (1984) *Classification and Regression Tree*, New York: Wadsworth.

BURRELL, P.R. and B.O. FOLARIN (1997) The impact of neural networks in finance, *Neural Computing and Applications*, **6**, 193–200.

CHEN, X., X. WANG and D.D. WU (2010) Credit risk measurement and early warning of SMEs: an empirical study of listed SMEs in China, *Decision Support Systems*, **49**, 301–310.

DEO, M.C. and C.S. NAIDU (1998) Real time wave forecasting using neural networks, *Ocean Engineering*, **26**, 191–203.

FRUTOS, S., E. MENASALVAS, C. MONTES and J. SEGOVIA (2003) Calculating economic indexes per household and censal section from official Spanish databases, *Intelligent Data Analysis*, **7**, 603–613.

HALL, P., J.S. MARRON and A. NEEMAN (2005) Geometric representation of high dimension, low sample size data, *Journal of the Royal Statistical Society, B*, **67**, 427–444.

HONG, W. (2008) Rainfall forecasting by technological machine learning models, *Applied Mathematics and Computation*, **200**, 41–57.

HONG, W. (2009) Chaotic particle swarm optimization algorithm in a support vector regression electric load forecasting model, *Energy Conversion and Management*, **50**, 105–117.

HOSMER, D.W. and S. LEMESHOW (1989) *Applied Logistic Regression*, New York: Wiley.

KAASTRA, I. and M. BOYD (1996) Designing a neural network for forecasting financial and economic time series, *Neurocomputing*, **10**, 215–236.

KIM, D.H., S.J. LEE, K.J. OH, T.Y. KIM and C. KIM (2009) An early warning system for financial crisis using a stock market instability index, *Expert Systems*, **26**, 260–273.

KOLODNER, J. (1991) Improving human decision making through case-based decision aiding, *AI Magazine*, **12**, 52–68.

LAPEDES, A. and R. FARBER (1998) How neural nets work, in *Neural Information Processing Systems*, D.Z. Anderson (ed.), New York: American Institute of Physics.

LEE, Y.C. (2007) Application of support vector machines to corporate credit rating prediction, *Expert Systems with Applications*, **33**, 67–74.

LI, N. and D.D. WU (2010) Using text mining and sentiment analysis for online forums hotspot detection and forecast, *Decision Support Systems*, **48**, 354–368.
OH, K.J. and K. KIM (2002) Piecewise nonlinear model for financial time series forecasting with artificial neural networks, *Intelligent Data Analysis*, **6**, 175–185.
OH, K.J. and T.Y. KIM (2007) Financial market monitoring by case-base reasoning, *Expert Systems with Applications*, **32**, 789–800.
OH, K.J., T.Y. KIM and C. KIM (2006) An early warning systems for detection of financial crisis using financial market volatility, *Expert Systems*, **23**, 83–98.
OSOWSKI, S. and K. GARANTY (2007) Forecasting of the daily meteorological pollution using wavelets and support vector machine, *Engineering Applications of Artificial Intelligence*, **20**, 745–755.
PODDIG, T. and H. REHKUGLER (1996) A 'world' model of integrated financial markets using artificial neural networks, *Neurocomputing*, **10**, 251–273.
SHIN, K.S. and I. HAN (1999) Case-based reasoning supported by genetic algorithms for corporate bond rating, *Expert Systems with Applications*, **16**, 85–95.
SHMUELI, G., N.R. PATEL and P.C. BRUCE (2006) *Data Ming for Business Intelligence*, New York: Wiley.
SON, I.S., K.J. OH, T.Y. KIM and D.H. KIM (2009) An early warning system for global institutional investors at emerging stock markets based on machine learning forecasting, *Expert Systems with Application*, **36**, 4951–4957.
TSAIH, R., Y. HSU and C.C. LAI (1998) Forecasting S&P 500 stock index futures with a hybrid AI system, *Decision Support Systems*, **23**, 161–174.
WHITE, H. (1989) Learning in artificial neural networks: a statistical perspective, *Neural Computation*, **1**, 425–464.
WU, D.D., X. KEFAN, L. HUA, Z. SHI and D.L. OLSON (2010) Modeling technological innovation risks of an entrepreneurial team using system dynamics: an agent-based perspective, *Technology Forecasting and Social Change*, **77**, 857–869.
WU, D.D. and D.L. OLSON (2009) Introduction to the special section on "optimizing risk management: methods and tools", *Human and Ecological Risk Assessment*, **15**, 220–226.
YU, L., S. WANG and K.K. LAI (2008) Forecasting crude oil price with an EMD-based neural network ensemble learning paradigm, *Energy Economics*, **30**, 2623–2635.
ZHANG, G., B.E. PATUWO and M.Y. HU (1998) Forecasting with artificial neural networks: the state of the art, *International Journal of Forecasting*, **14**, 35–62.

The authors

Jae Joon Ahn

Jae Joon Ahn is a PhD candidate in the Department of Information and Industrial Engineering at Yonsei University, South Korea. He received his BA (2004) and ME (2008) degree in Industrial Engineering at Yonsei University. He has published some technical articles to international journals. His research interest includes financial engineering and financial time series analysis.

Il Suh Son

Il Suh Son is an analyst at research centre of Daewoo Securities Co. He received his BA (2006) and ME (2008) degree in the Department of Information and Industrial Engineering at Yonsei University, South Korea. He has published some technical articles to international journals and conferences. His research fields are financial information systems and system trading.

Kyong Joo Oh

Kyong Joo Oh is an associate professor in the Department of Information and Industrial Engineering at Yonsei University, South Korea. He received his BA (1991) and MA (1993) degree in Applied Statistics at Yonsei University and PhD degree (2000) in Management Information Engineering at Korea Advanced Institute of Science and Technology (KAIST). Dr Oh served as a researcher at Marketing Strategic Institute of Diamond Advertising Co. (1995), KAIST Techno-Management Institute (2000) and Research Center of Hyundai Securities Co. (2001). He has published over 20 technical articles in some international journals, such as *Expert Systems*, *Applied Intelligence*, *Technological Forecasting and Social Change*, *Expert Systems with Applications*, *Neurocomputing*, *Intelligent Data Analysis*, *Asia-Pacific Journal of Operational Research*, etc. His research fields are financial information systems, financial and

investment engineering, artificial intelligence in finance, system trading, etc.

Tae Yoon Kim

Tae Yoon Kim is a professor in the Department of Statistics at Keimyung University, South Korea. He got his PhD in statistics at the University of Illinois at Urbana-Champaign (1990). His research interest includes non-parametric function estimation, bootstrap, neural network and time series analysis of financial markets. He served as visiting research scholar at the Bank of Korea (2001), Department of Statistics at Rice University (1998) and Department of Statistics at Seoul National University (2006).

Gyu Moon Song

Gyu Moon Song is a professor in the Department of Statistics at Keimyung University, South Korea. He received PhD degree in statistics at Sungkyunkwan University, South Korea (1991). His research interest includes statistical linear models and time series analysis of financial markets.