JEL classification: C45, C53, E44
Keywords: emerging stock markets, predictability of stock returns, neural networks

How Do Neural Networks Enhance the Predictability of Central European Stock Returns?*

Jozef BARUNÍK – Institute of Economic Studies, FSV, Charles University, and Institute of Information Theory and Automation, Academy of Sciences of the Czech Republic, Prague (barunik@utia.cas.cz)

Abstract
In this paper, we apply neural networks as nonparametric and nonlinear methods to Central European (Czech, Polish, Hungarian, and German) stock market returns modeling. In the first part, we present the intuition of neural networks and we also discuss statistical methods for comparing predictive accuracy, as well as economic significance measures. In the empirical tests, we use data on the daily and weekly returns of the PX-50, BUX, WIG, and DAX stock exchange indices for the 2000–2006 period. We find neural networks to have a significantly lower prediction error than the classical models for the daily DAX series and the weekly PX-50 and BUX series. We also achieve economic significance of the predictions for both the daily and weekly PX-50, BUX, and DAX, with a 60% prediction accuracy.

1. Introduction

Life must be understood looking backwards, but must be lived looking forward. The past is helpful for predicting the future, but we have to know which approximating models to use, in combination with past data, to predict future events. On the basis of the universal approximation theorem, we use neural networks in the hope that they will improve the prediction task, as they are able to approximate any function, as Hornik, Stinchcombe, and White (1989) show. Thus, we will aim to compare the results of econometric modeling and neural network modeling to see whether neural networks bring us a closer insight into the patterns of stock returns or not. The reader will see that the neural network is a very useful nonparametric econometric technique. On the other hand, criticisms arise mainly from the fact that neural networks were inspired by biological phenomena – the physiology of nerve cells – and have become part of a separate literature (see (Hertz, Krogh, Palmer, 1991), (Hutchinson, Lo, Poggio, 1994), (Poggio, Girosi, 1990), and (White, 1988), for an overview). Neural network learning methods provide a robust approach to approximating real-valued, vector-valued, and discrete-valued functions. The study of artificial neural networks (ANNs) was inspired by the observation that biological learning systems are built of very complex webs of interconnected neurons. ANNs are analogously built webs of interconnected sets of simple units, or inputs, which may be outputs of other units and which produce simple outputs which may become inputs in other units (Mitchell, 1997). By referring to "neural networks" we will mainly consider research targeting the development of systems capable of approximating complex functions efficiently and robustly.

* Support from GAUK 46108, Czech Science Foundation, under grant 402/06/1417 and MSM0021620841 is gratefully acknowledged.

As classical econometric models provide us with some insights into the behavior of stock returns, we believe that neural networks will do better. We believe that the neural network learning process will help approximate the learning process of agents or investors more efficiently, resulting in a better understanding of stock prices. Contrary to the Efficient Market Hypothesis, several researchers, such as Hsieh (1991), Barkoulas and Travlos (1998), and Peters (1994), claim that stock markets exhibit chaos. Chaos is a nonlinear deterministic process which appears random, but cannot be easily expressed. With the neural network's ability to learn nonlinear, chaotic systems, it may be possible to outperform traditional analysis. McNelis (2005) shows very good results in predicting artificial data and chaos processes by neural networks and shows how artificial intelligence could shed more light on the time-series processes. He tests the predictive power of the models also on industry data and inflation, but a test on stock markets is missing. In this paper, we will follow his and other work with empirical research on Central European markets. Žikeš (2004) finds that Central European markets also do not follow a random walk. Filáček et al. (1998) find that the daily returns of the main Prague Stock Exchange (PSE) index, the PX-50, are significantly positively autocorrelated. We believe that emerging markets in particular, or markets with a high level of innovation and structural change, represent a great opportunity for the use of neural networks in the prediction task. The reasons are intuitive, as explained below.

The data are often very noisy, either because of thinness of the markets or due to information or discontinuous trading[1] gaps. Thus we have to deal with lots of asymmetries and nonlinearities which cannot be assumed. The other reason is that agents in these markets are themselves in a process of learning, mainly by trial and error. Often they cannot predict the impact of policy news or legal changes to the market simply because they have not seen any real examples in their past. Thus, the information set for the prediction task is very limited. As we will show, parameter estimates of neural networks are themselves a result of "learning by mistake" and the search process and can be compared to the parameters used by agents to forecast and make decisions. In this part, we will present the theoretical framework of neural networks used subsequently in the empirical modeling work.

The paper is organized as follows. In the first part, we introduce the reader to neural networks. We first discuss methodological problems to avoid confusion, and we then present the basic forms of networks and transformation functions which will be tested later on in the paper. We also pay attention to the evaluation of the estimated models and to statistical methods of predictive accuracy and economic significance. In this paper, we apply neural networks to Central European stock market returns – the PX-50, BUX, DAX, and WIG daily and weekly returns. On the in-sample and more important out-of-sample criteria, we test classical autoregressive models – ARIMA (p,I,q) and GARCH – with neural networks. For the comparison, we use the statistical tests described in the theoretical part, and also tests of the economic relevance of the prediction model. The paper concludes with a summary of the empirical results we achieve and suggestions for further research.

[1] Often there are many stocks with no or very low volume trades on these markets.

2. Neural Networks

2.1 Methodological Problems

Much of the early development and work on neural network analysis was within psychology and neuroscience, related to pattern recognition problems. The genetic algorithms used for the empirical implementation of neural networks have followed a similar pattern of development in applied mathematics in the optimization of dynamic nonlinear and discrete systems, moving into data engineering. Thus, these systems have been developed in different surroundings than econometrical and statistical models, resulting in confusion in the literature, mainly as regards the simple technical and naming conventions. A *model* is known as an *architecture*, and we *train* rather than *estimate* the network architecture. A researcher uses a *training set* and *test set* of data instead of *in-sample* and *out-of-sample* data, and the confusion should disappear whenever the reader expects *coefficients* instead of *weights*. If we consider the application of neural networks, or artificial intelligence itself, the gap widens. The broad literature on neural networks is simply not relevant to financial professionals or academics. Also, publications and empirical work using neural networks in finance are not linked to the preceding theoretical financial literature, which is probably why most of this literature is not taken seriously by the broader financial and economic academic community. As McNelis remarks: "The appeal of the neural network approach lies in the assumption of *bounded rationality*: when we forecast in financial markets, we are forecasting the forecasts of others, or approximating the expectations of others." (McNelis, 2005) Thus, market participants are continuously learning and adapting their beliefs from past mistakes.

2.2 What is a Neural Network?

Like linear or nonlinear methods, a neural network relates a set of input variables, say, $\{x_i\}, i = 1,...,k$, to a set of one or more output variables, say, $\{y_j\}, j = = 1,...,k^*$. The only difference between network and other approximation methods is that the approximating function uses one or more so-called hidden layers, in which the input variables are squashed or transformed by a special function. This is known as logistic or logsigmoid transformation. While this approach may seem "*esoteric*" or maybe even "*mystical*" at first glance, the reader will soon see that it may be used as a very efficient way to model nonlinear processes. The reason we turn to neural networks is straightforward. It is the goal of the prediction problem to find an approach or method that best forecasts the data, generated by unknown, nonlinear processes, with as few parameters as possible, which are as simple to achieve and as easy to estimate as can be. Moreover, it has been shown that "neural networks can approximate any function with finitely many discontinuities to arbitrary precision". This is known as *the universal approximation theorem* (Hornik, Stinchcombe, White, 1989).

2.2.1 Feedforward Networks

The most basic and commonly used neural network in finance – with one hidden layer (Hornik, Stinchcombe, White, 1989) containing two neurons, three input variables, and one output – is the feedforward neural network. The general *feedforward* or *multilayered perception* (MLP) network can be described by the following equations:

$$n_{k,t} = \omega_{k,0} + \sum_{i=1}^{i^*} \omega_{k,i}\, x_{i,t} \tag{1}$$

$$N_{k,t} = \Lambda\left(n_{k,t}\right) = \frac{1}{1+e^{-n_{k,t}}} \tag{2}$$

$$y_t = \gamma_0 + \sum_{k=1}^{k^*} \gamma_k N_{k,t} \tag{3}$$

where $\Lambda\left(n_{k,t}\right)$ is the logsigmoid activation function. There are i^* input variables $\{x\}$ and k^* neurons. $\omega_{k,i}$ represents a coefficient vector or *input weights* vector. Variable $n_{t,k}$ is squashed by the logsigmoid function, and becomes a neuron $N_{t,k}$ at time *t*. Then the set of k^* neurons are combined linearly with the vector of coefficients $\{\gamma_k\}, k = 1,...,k^*$ to form the final output, which is the forecast $\hat{y}_t$. This model is the workhorse of the neural network forecasting approach, as almost all researchers start with this network as the first alternative to linear models.

In contrast to classical linear models, there are two additional neurons which process the inputs to improve the predictions. It should be mentioned here that the connections between the input variables and the neurons, also called *input neurons*, and the connections between the neurons and the output, the *output neurons*, are called *synapses*. The reader might note that the simple linear regression model is just a special case of the feedforward neural network, namely a network with one neuron which contains a linear approximation function.

2.2.2 Transformation Functions – Logsigmoid, Tansig, and Gaussian

Maybe the most confusion about neural networks comes from the presence of a hidden layer and the function of neurons. They process inputs by forming linear combinations of them and then squashing these combinations using the transformation function. The previous example of a feedforward network contains a logsigmoid activation function (equation 2.2.). This function reflects the learning behavior of the networks, or, more precisely, "learning by doing". The function is increasingly steep until the inflection point, from which it becomes increasingly flat and its slope moves exponentially to zero. The nonlinear sigmoid function captures the learning process in the formation of expectations characterized by *bounded rationality*. Kuan and White (1994) describe it as the "tendency of certain types of neurons to be quiescent of modest levels of input activity, and to become active only after the input activity passes a certain threshold, while beyond this, increases in input activity have little further effect".

An alternative to the logsigmoid activation function is the *tansig*, or *tanh*, hyperbolic tangent function. Its behavior is very similar to the logsigmoid function, but it squashes the linear combinations within a wider interval of $\langle -1,1 \rangle$ rather than $\langle 0,1 \rangle$. Instead of equation 2.2., we would use a *tansig* squasher function in the network architecture:

$$N_{k,t} = T\left(n_{k,t}\right) = \frac{e^{n_{k,t}} - e^{-n_{k,t}}}{e^{n_{k,t}} + e^{-n_{k,t}}} \tag{4}$$

where $T\left(n_{k,t}\right)$ is the tansig activation function. Another activation function is the *cumulative Gaussian function*, commonly referred to as the normal function. The advantage of using the Gaussian function is that it has thinner tails, so it does not respond to some extreme values. In a network architecture with a Gaussian activation function, we would use $N_{k,t} = \Phi\left(n_{k,t}\right)$, where $\Phi\left(n_{k,t}\right)$ is the standard cumulative Gaussian function, instead of equation 2.2.

We have just described the basic functional forms of neural networks with the most commonly used transformation functions. The reader is now probably asking the questions: "OK but, what transformation function should I use?" or "Are there any other transformation functions?" There are many other possible transformation functions, in fact. The reason we describe these few is that they performed best in our tests and are also used in each of the references used in this paper. The answer to the first question is not as simple as the answer to the second. Each transformation function transforms inputs in a different manner. Some respond to extreme values, some do not, thus they do not serve equally well in approximating the unknown function. Hence, choosing the form of squasher function is often up to the researcher and the data used. The best way is to perform tests with different transformation functions used in the neurons and use the one which performs best. This is one of the main drawbacks of neural networks.

2.3 Learning Algorithms

In order to be able to approximate the target function – in our case stock returns, the neural network has to be able to *"learn"*. The process of learning is defined as the adjustment of weights using a learning algorithm. The backpropagation algorithm and two more specific algorithms – the conjugate gradient algorithm and the Levenberg-Marquardt algorithm – are considered in our empirical tests as the most common methods. The latter two are presented mainly because they provided the most impressive results in comparison to other common methods. The most common way to train a neural network is by learning an algorithm called *"backpropagation"* or *"error-backpropagation."* The main goal of the learning process is to minimize the sum of the prediction errors for all training examples. The training phase is thus an unconstrained nonlinear optimization problem where the goal is to find the optimal set of weights of the parameters by solving the minimization problem:

$$\min\left\{\Psi\left(\omega\right) : \omega \in \Re^n\right\} \tag{5}$$

where $\Psi : \Re^n \to \Re$ is a continuously differentiable error function. There are several ways of minimizing $\Psi\left(\varpi\right)$, but basically we are searching for the gradient $G = \nabla\Psi\left(\varpi\right)$ of function Ψ which is the vector of the first partial derivatives of the error function $\Psi\left(\varpi\right)$ with respect to the weight vector ϖ. Furthermore, the gradient specifies the direction that produces the steepest increase in Ψ. The negative of

this vector thus gives us the direction of steepest decrease. The stochastic gradient descent backpropagation learning algorithm, as well as other methods, will not be discussed in any further detail, in order to keep the length of the paper under control.

Besides the popular steepest descent algorithm, the conjugate gradient algorithm is another search method that can be used to minimize the network error function $\Psi(\varpi)$ in conjugate directions. This method puts into use the orthogonal and linearly independent non-zero vectors and in some cases brings better convergence results than the previous method. While these methods are too simplistic for more complex models, the technique invented by Levenberg (1944) involves blending between the introduced steepest gradient and quadratic approximation. It uses the steepest gradient to approach the minimum, and then switches to quadratic approximation. Marquardt (1963) improved this method with a clever incorporation of estimated local curvature information. Its only drawback is that it requires a matrix inversion step, which makes it much slower than *backpropagation* or the *conjugate gradient* in more complex models. On the other hand, it has much better results.

Finding the coefficient values of nonlinear models is not that easy a job, as a neural network is a highly complex nonlinear system. We can hit several locally optimal solutions, but none of these may be the best solution in terms of minimizing the error between our model prediction $\hat{y}$ and the actual value y. Maybe the reader is asking the question: "But what can we do to avoid this problem?" There are several techniques for minimizing the chance of converting to the "wrong" optimum. A very intuitive way is to re-estimate the whole model. Another way is stochastic evolutionary search. A genetic algorithm reduces the likelihood of landing in a local minimum. We do not need to approximate the Hessian; we start with a "population" of p initial guesses, $\{\omega_{0,1}, \omega_{0,2}, ..., \omega_{0,p}\}$ and update them by genetic selection, breeding, and mutation, for many generations, until the best coefficient vector is found. One of the main drawbacks of genetic algorithms is their extreme slowness.

2.4 Evaluation of the Estimated Models

So far we have presented the complex procedure of estimation with neural networks. Before we proceed to the empirical analysis, we will briefly present a few criteria that will help us interpret the results. We will work with in-sample criteria, or training period results interpretation, which is in fact the evaluation of information on how well the estimated data fits our modeled data. We will see that the model that explains most of the variation of the training data may turn out to be inapplicable for forecasting purposes, or, put better, for out-of-sample data that the model "has not seen before". These are also called testing data, or out-of-sample criteria. They will be the most important for us in the testing part.

The framework of the empirical testing is the following. After preprocessing the data, we divide it into three samples – *training*, *cross-validation*, and *testing sets*. The neural network will be estimated using the training data, and the optimal weights will be found at this stage. Then the weights are put to the cross-validation data and might be slightly adapted to changes if we find that the in-sample criteria have deteriorated. Just after that, the last set of data is put to the test. The coefficients obtained from the training will be used with new data which had no impact on the calculation

of the coefficients. This is the most important part. The reader should also be aware of the ratio of the training to the testing data set. In most of the studies, a 20–25% cut is made for testing purposes, but it can be crucial for our results to do this with patience. Imagine that we want to model AAA stock returns and that on January 15, 2002 there were huge reforms in the company leading to consistently higher-than--expected profits. This would also have an impact on the returns of our AAA company, and if we train the network on the data until January 15, 2002 and try to test them further on, we may be extremely disappointed. Our model will only "know" the pattern from the pre-reform period. Hence, according to the changes, the pattern of returns also changed after the date and our model will not be capable of dealing with it.

2.4.1 Statistical Comparison of Predictive Accuracy

Another key question in forecasting is measurement of the accuracy of different forecasts, as we are interested in the model that produces the most accurate forecasts. As we are going to compare the performance of various econometric models and neural network models, we have to consider statistical methods for comparing the results so we are able to identify if neural network models help us to produce more accurate results or not. This needs to be done on an out-of-sample model evaluation.

Let us consider two *h-step* forecasts, $\left\{\hat{p}^{i}_{t+h|t}\right\}_{t=1}^{T}$ and $\left\{\hat{p}^{j}_{t+h|t}\right\}_{t=1}^{T}$, of the time series $\left\{p_{t+h}\right\}_{t=1}^{T}$, with forecast errors of $\left\{\varepsilon^{i}_{t+h|t}\right\}_{t=1}^{T}$ and $\left\{\varepsilon^{j}_{t+h|t}\right\}_{t=1}^{T}$. To choose a model with a significantly lower prediction error, and thus better accuracy, we wish to compare the expected loss associated with both forecasts. Of course, this will depend on the chosen loss function. We will restrict on the general loss function dependent on the forecast error here, $L\left(\varepsilon_{t+h|h}\right)$, and we will try to find the optimal *h-step* prediction: $\hat{P}^{*}_{t+h|t} \equiv \arg\min E\left[L\left(\varepsilon_{t+h|t}\right)|\mathrm{F}_{t}\right]$.

Thus, we will test the null hypothesis of equal forecast accuracy for two forecasts against the alternative hypothesis of unequal forecast accuracy $H_0 : E\left[L\left(\varepsilon^{i}_{t+h|t}\right) - L\left(\varepsilon^{j}_{t+h|t}\right)\right] = 0$, where $L\left(\varepsilon_{t+h|t}\right)$ is a positive loss function and $L\left(\varepsilon^{i}_{t+h|t}\right) - L\left(\varepsilon^{j}_{t+h|t}\right)$ is the loss differential; a quadratic loss function will be considered.

The most important question is how we can determine if the out-of-sample fit of one model is significantly better than the out-of-sample fit of another model. Diebold and Mariano (1995) have proposed a test for the null hypothesis of equal predictive ability, against the alternative of non-equal predictive ability, which we will use in our testing as a core test. If we find a significant predictive edge in the data, we will analyze how the results of the neural network lend themselves to interpretations that make economic sense and give us better information for decision making. The simple asset allocation strategy test we use for assessing this question

was created by Henriksson and Merton (1981) and Lo and MacKinlay (1997). Henriksson and Merton (1981) proposed a non-parametric measure to evaluate the performance of the trading strategy. In financial time series, one is often interested more in the sign of the stock return predictions than in the exact value. If we have a good sign predicting model, we are able to profit on the market "inefficiency". The statistic we use was formalized by Pesaran and Timmerman (1992) and is based on the null hypothesis that a given model has no economic value in forecasting the direction.

3. Application to Central European Stock Market Return Modeling

Finally, we will use the presented theory for modeling the Central European stock markets. We believe that the emerging markets represent the best ground for the use of neural network models. The data is very often much noisier because the markets are very thin and also due to the speed with which news spreads among market agents. Thus, our assumption is that neural networks should be able to help uncover the process. As motivation for good modeling results for emerging markets, the reader may be interested in the following recent research. Almost all the results are very impressive. Nygren (2004) examines the predictability of the Swedish Stock Exchange, Mohan, Jha, Laha, and Dutta (2005) examine the predictive power of neural networks on the Bombay Stock Exchange, and Cambazoglu (2003) finds impressive patterns on the Turkish Stock Exchange. Finally, Yao, Tan, and Poh (1999) study the Kuala Lumpur Stock Exchange with some impressive results. Encouraged by previous research, we intend to test the power of neural networks in Central European markets against linear methods. We will model the returns on Central European stock indices daily and weekly. Specifically, we look at the Prague, Warsaw, and Budapest stock exchanges, which we believe describe the corresponding stock markets well. For comparison and more complex forecasting model development, we will also analyze the Deutsche Börse index, which is believed to be the most liquid in continental Europe.

3.1 European Stock Markets

3.1.1 Data Description

In the prediction task, we focus on a sample of 1,566 daily returns (to achieve stationarity all the data are first differences of the log series $r_t = \ln P_t - \ln P_{t-1}$) from January 2000 until April 2006, and 382 weekly returns from January 1999 until April 2006 of the value-weighted PX-50, WIG, BUX, and DAX[2] indices. All the data were downloaded and regularly uploaded from Bloomberg during the research. Monthly returns were omitted because the sample size is very small even for a neural network. The descriptive statistics of the series are summarized in *Table 1*.

The Jarque-Bera test statistics tell us that all the indices for daily and weekly returns deviate from the normal distribution. The distributions of the Central European stock markets are in line with the developed stock market distributions. They are *leptokurtic* as expected, which means that they are said to have heavy tails. This may be attributed to conditional heteroskedasticity, so it is important to notice this before estimation.

[2] PX-50 – Prague Stock Exchange, WIG – Warsaw Stock Exchange, BUX – Budapest Stock Exchange, DAX – Deutsche Börse.

TABLE 1 The Descriptive Statistics

			Daily (1564 observations)			Weekly (381 observations)		
	BUX	**DAX**	**PX-50**	**WIG**	**BUX**	**DAX**	**PX-50**	**WIG**
Mean	0.00067	-0.00005	0.00075	0.00056	0.00293	0.00337	0.00028	0.00329
Median	0.00049	0.00045	0.00081	0.00047	0.00240	0.00525	0.00332	0.00507
Maxi-mum	0.06004	0.07553	0.04179	0.05593	0.09569	0.08719	0.12887	0.11501
Mini-mum	-0.07433	-0.08875	-0.06000	-0.08468	-0.13579	-0.09876	-0.13919	-0.18100
Std, Dev.	0.01410	0.01690	0.01248	0.01281	0.02967	0.02748	0.03383	0.03402
Skew-ness	-0.14797	-0.01262	-0.27616	-0.12427	-0.20928	-0.23586	-0.17928	-0.40852
Kurtosis	4.88697	5.61569	4.38258	5.54571	4.61303	3.69753	4.27986	5.35253
Jarque-Bera	237.74*	445.90*	144.45*	426.35*	44.09*	11.26*	28.05*	98.46*

Note: * significance on the 1% level

TABLE 2 In-Sample Performance on Daily Returns

	PX50		**BUX**		**WIG**		**DAX**	
	linear	**neural**	**linear**	**neural**	**linear**	**neural**	**linear**	**neural**
Adj *R*-squared	0.01	**0.19**	0.02	**0.11**	0.01	**0.09**	0.02	**0.16**
Schwarz criterion	-6.02	**-9.283**	-5.73	**-8.58**	-6.01	**-8.61**	-5.71	**-6.5**
Ljung-box Q(4)	8.96*		3.8**		7.70		3.31	
Ljung-box Q(8)	13.31**		5.98		10.48		7.18	
Ljung-box Q(12)	16.42**		9.78		13.82		13.48	

Note: *,**,*** significance on 1%, 5% and 10% levels

3.1.2 Empirical Results – Daily Returns

We start by modeling the daily returns of each index by ARIMA estimation. The augmented Dickey-Fuller statistics exceed the critical values at the 1% significance level, so we can reject the null of the presence of a unit root and state that all the tested series are stationary. The PX-50 seems to follow ARIMA(1,0,1) best. The BUX returns seem to be explained well by ARIMA(2,0,2). The WIG and DAX do not contain AR and MA errors, so the random walk hypothesis cannot be rejected for them. The Ljung-Box *Q* statistics show us the presence of conditional heteroskedasticity in the residuals from the ARIMA models. So, we will try to model them using the GARCH(1,1) model, as it turns out that this model rules not only with its parsimony, but also with its performance with these series. We find the ARIMA--GARCH models to be the most appropriately specified – ARIMA(1,0,1)-GARCH(1,1) for the PX-50, ARIMA(2,0,2)-GARCH(1,1) for the BUX, and GARCH(1,1) for the DAX and WIG returns. According to the results in *Table 2*, we can see that the null hypothesis of no serial correlation can be clearly rejected with the PX-50 model and also with the BUX model. Thus, these models do not explain all of the variance and should be used with caution for the forecasting task. We will use them only as representatives of linear modeling against the neural networks, because we did not find any better specified models for the data. This might be explained by the use of daily stock returns, which are autocorrelated due to the effect of nonsyn-

TABLE 3 Out-of Sample Performance On Daily Returns

	PX50		BUX		WIG		DAX	
	linear	**neural**	**Linear**	**neural**	**linear**	**neural**	**linear**	**neural**
RMSE	0,02	**0,01**	0.02	**0.15**	0.01	**0.01**	0.09	0.08
NMSE	1,00	**0,97**	0.99	**0.98**	1.01	**0.99**	1.01	1.01
D-M(0)		-1.1		-0.59		-0.98		**-1.72 ****
D-M(1)		-0.91		-0.82		-1.09		**-1.78 ****
D-M(2)		-0.83		-0.79		-1.2		**-2.02 ****
D-M(3)		-0.71		-0.78		-1.16		**-1.72 ****
H-M	1*	**1.08 ***	1.01 **	**1.02 ***	1 *	1	1 *	1.03
P-T	51 %	**56%*****	53 %	**54 % *****	54 %	54 %	62 %	47 %
TC	0.00 %	**1.2 %**	0.31 %	**1.1 %**	0.00 %	0.2 %	0.03 %	0.3 %

Note: D-M: Diebold-Mariano statistic (p-values), H-M: Henriksson – Merton statistic, P-T: Pessaran-Timmerman (SR with p-value), TC – total costs.
*,**,*** significance on 1%, 5% and 10% levels.

chronous trading.[3] Consequently, in the following sections the use of weekly data should improve the performance of these models.

In contrast to modern econometric tools, we will model stock returns using the presented neural network methodology. A simple feedforward time-delayed structure of the network will be used in the testing, with one hidden layer and the Levenberg-Marquardt algorithm. The inputs used were three lagged variables mapped into three neurons, as we found this provided the best results. From the results obtained (*Table 2*), we can see that there is a very poor pattern to be learned from our data. It seems that although the index returns are predictable to some extent, the predictability is very small. Neural networks perform a little better as regards explaining the in-sample data. R^2 increases from 0.4 % achieved by the linear model to 19% achieved by the neural net with the PX-50 index, and the results with the other indices are similar, as shown in *Table 2*. The Schwarz criterion also favors neural networks.

But the real test of out-of-samples (*Table 3*) does not show a very big difference between the linear and neural network models. We withheld 20 % of the data as a rule of thumb for out-of-sample testing. As for the Diebold Mariano test, we cannot reject the null hypothesis of equal predictive accuracy of the linear and neural network models for all the series tested except the DAX. Thus, the neural network model does not seem to have significantly different errors for the tested daily returns. On the other hand, the economic significance of the predictions differs. For all the linear models, we cannot reject the null hypothesis of no predictability with the Henriksson-Merton statistic[4] or with the Pesaran-Timmerman. Thus, the linear models have no economic value and should not be used for real predictions. Even the implied transaction costs are at a very low level. The situation is a little different with the neural network models. From the *Table 3* we can also see that with the PX-50 and BUX data, we can reject the null of no predictability, while H-M is significant at the 1% level for both data sets. P-T is significant at the 10% level for the PX-50 and

[3] For more details on this issue, see (Campbell, Lo, MacKinlay, 1997).

[4] We use the PRIBOR as the risk-free rate, and it will also be used in the following tests.

the BUX, which means that the null hypothesis of independence of actual signs and forecasted signs can be rejected at the 10% significance level. The implied transaction costs are higher than the real world transaction costs. Thus, even if the neural networks could not beat the linear models with statistically significant lower errors, they seem to have economic value at least for two of the tested series. Although we can gain some predictive edge with the daily European stock returns, the time series do not seem to explain themselves very well. This may have been caused by autocorrelation which we could not remove, but as to the power of the approximation ability of neural networks, we think the tested daily returns may simply be unpredictable, or producing insignificant predictions. We will see if the weekly data bring us better results, allowing us to gain some more predictive edge using neural network models.

3.1.3 Empirical Results – Weekly Returns

Again, we start with a very similar approach with the weekly returns. The ADF test confirms the stationarity of the data, so we can proceed to the Box-Jenkins methodology. The PX-50 follows ARIMA (1,0,0). This result is interesting, because the weekly data no longer contains MA errors. The other weekly returns are best explained with the same models as the daily ones. After observation of the *Q* statistics, we add GARCH(1,1) to model heteroskedasticity in the residuals and we end up with ARIMA(1,0,0)-GARCH(1,1) for the PX-50, ARIMA(2,0,2)-GARCH(1,1) for the BUX, and GARCH(1,1) for the DAX and the WIG returns. Interestingly, the null hypothesis of no serial correlation cannot be rejected at the 1%, 5%, or even 10% significance levels. Thus, the models seem to explain most of the variance in the data and thus can be used for predicting. A feedforward time-delayed neural network architecture with one hidden layer, three inputs (lagged variables), a logsigmoid squasher function and the Levenberg-Marquardt algorithm is again put to test. From the results obtained, we can see that the in-sample improvement with the neural network seems to be really significant as regards explanatory power and the Schwarz criteria. (*Table 4*)

Let us turn to the more interesting out-of-sample forecasts (result in *Table 5*). Diebold-Mariano tells us that the neural networks have a significantly lower error compared to the linear models with the PX-50 and BUX, as the null hypothesis of equal predictive accuracy can be rejected at the 5% significance level for all lags. For the other two tested series, the WIG and the DAX, the null of equal predictive accuracy cannot be rejected, so for this data the models perform statistically similarly. As to the economic significance of the forecasts, we reject the null hypothesis of no predictability using H-M for the PX-50 and WIG series at the 1% significance level, and for DAX at the 10% significance level. According to P-T, the neural networks also have significant sign predictions, as the null hypothesis of independence of signs between the predicted series and the actual ones can be rejected at the 10% significance level. The implied transaction costs are quite low, but slightly higher than those of the real world.[5] The models did not perform well with the BUX series only, where we cannot reject either the null hypothesis of no predictability, or the null of sign independence.

[5] We found the real-world transaction costs for a €10,000 investment in the Czech Republic to be around 0.05 % on average.

TABLE 4 In-Sample Performance on Weekly Returns

	PX50		BUX		WIG		DAX	
	linear	**neural**	**linear**	**neural**	**linear**	**neural**	**linear**	**neural**
Adj *R*-squared	0.02	**0.48**	0.01	**0.15**	-0.00	**0.28**	-0.00	**0.34**
Schwarz criterion	-4.38	-9.46	-3.95	-11.17	-4.25	-7.29	-4.12	6.87
Ljung-box Q(4)	0.10		1.45		7.07		3.24	
Ljung-box Q(8)	5.94		2.25		16.33***		6.28	
Ljung-box Q(12)	8.09		7.61		19.15***		10.81	

Note: *,**,*** significance on 1%, 5% and 10% levels

TABLE 5 Out-of Sample Performance on Weekly Returns

	PX50		BUX		WIG		DAX	
	linear	**neural**	**linear**	**neural**	**linear**	**neural**	**linear**	**neural**
RMSE	0.02	0.02	0.03	0.02	0.03	0.02	0.02	0.02
NMSE	0.99	0.98	0.99	0.99	1.03	0.97	1.03	0.99
D-M(0)		**-2.01****		**-1.78****		-0.65		-0.67
D-M(1)		**-2.03****		**-1.89****		-0.62		-0.54
D-M(2)		**-1.85****		**-1.98****		-0.68		-0.48
D-M(3)		**-1.94****		**-1.8****		-0.8		-0.51
H-M	1.07 **	**1.09***	0.9**		1	**1.1 ***	1*	**1.2 *****
P-T	58 %	**60 %*****	58 %	**60%**	0.55 %	**58 %*****	55 %	**58 % *****
TC	0.4 %	**0.8 %**	-0.6 %	**0.1 %**	0.01 %	**1 %**	0.03 %	**0.7 %**

Note: D-M: Diebold-Mariano statistic (p-values), H-M: Henriksson – Merton statistic, P-T: Pessaran-Timmerman (SR with p-value), TC – total costs.

*,**,*** significance on 1%, 5% and 10% levels.

From the preceding tests, we can conclude that there is a predictive edge in the European stock markets. Neural networks seem to explain the time series a little better than the classical approach. When facing the prediction task, the results are also improved. We can say that with a significant chance of 3:2, next week's return can be predicted with the use of raw price data with a neural network. We use these results as the starting point for the development of a more robust model in the next part. While it is clear that one can gain abnormal returns using the presented methods, we will try to propose a different model which will use not only the lagged variables of the time series itself, but also other variables to gain more explanatory power and robust results even on daily returns.

3.2 PX-50: Gaining a Predictive Edge

We found that the European Stock markets contain predictable components, but the use of models with lagged data does not seem to provide us with strong results[6] on the daily data. On the weekly data, the models performed significantly better in two cases, and we managed to gain economic significance almost for all the series tested. We will continue with a different approach and try to find an em-

[6] The results are not that bad, though. The reader should bear in mind that if we can predict future returns with 55%–60% accuracy, we have a "3/2 : 1" ratio of winning to losing trades. If we manage to predict returns with 70% accuracy, this is actually an excellent result, as we have a "7/3 : 1" ratio of winning to losing trades and we can consistently earn abnormal returns from the market.

TABLE 6 Results of PCA

	PX50		BUX		WIG		DAX	
	classical	neural	classical	neural	classical	neural	classical	neural
Adj *R*-squared	0.28[a]	0.31	0.35[b]	0.4	0.32[c]	0.34	0.18[d]	0.24
Schwarz criterion	-6.36	-9.13	-6.2	-9.13	-6.47	-9.25	-5.42	-8.2
Ljung-box Q(4)	10.98**		20.2*		8.58***		17.23*	
Ljung-box Q(8)	13.24		30.01*		9.91		55.03*	
Ljung-box Q(12)	15.23		33.97*		14.69		59.95*	

Notes: [a] PX50 returns are being explained by BUX, WIG and DAX with coefficients 0.276*, 0.243* and 0.076* resp.

[b] BUX returns are being explained by PX50, WIG and DAX with coefficients 0.322*, 0.376* and 0.117* resp.

[c] WIG returns are being explained by PX50, BUX, and DAX with coefficients 0.2189*, 0.290* and 0.1075* resp.

[d] DAX returns are being explained by PX50, BUX and WIG with coefficients 0.191*, 0.258* and 0.30* resp.

*,**,*** significance levels of 1%, 5% and 10% resp.

pirical relationship between the European stock markets, and if we manage to find one, we will use it to build a model that will give us a deeper understanding of the PX-50 stock market returns. In this part, we will use the same daily data as described in the previous section. We just remind the reader that for all of the tests we divide the tested sample into 70 % in-sample, 10 % cross-section, and 20 % out--of-sample for the neural nets, and 80 % : 20 % for the regressions.

3.2.1 Cointegration of the BUX, WIG, DAX, and PX-50 Markets

Our first hypothesis is that the PX-50, DAX, BUX, and WIG are co-moving and thus the returns of these markets can be used to shed more light on their patterns and to predict each other. Žikeš (2004) provided us with the results of a Johansen multivariate cointegration analysis and found that all markets are influenced by at least one lagged variable of neighboring markets. Instead of conducting the same research and getting the same results, we will try to use his results in our modeling. The Czech, Hungarian, and Polish markets are moving together. The German market was falling much faster during 2002–2003, and in the middle of 2003 it joined the other markets but underperformed them. From this period, we can see that the markets are co-moving. With the rigorous empirical background of Žikeš's analysis, we can use this information for the prediction of the PX-50 stock market returns.

First of all, we conduct a Principal Component Analysis (PCA) to find which vectors influence market returns the most. We will conduct a classical regression PCA analysis and also a nonlinear neural network PCA (the reader is advised to consult (McNelis, 2005) for the methodology) for all four indices. A logsigmoid squasher function and the Levenberg-Marquardt optimization mechanism will be used. The results are in *Table 6.*

Thus, we can see that all the markets really do influence each other and move within a narrow range. Consequently, we can try to use the lags of the PX-50, BUX, and WIG to explain their variance and follow the previous analysis. The reader has

no doubt noticed that the DAX coefficients are the smallest, so the DAX surprisingly does not have such a big influence on the three market indices. They explain themselves best, and this information can also be used for predicting them in the following text. Not so surprisingly, the DAX is not explained well with the PX-50, BUX, and WIG returns. This is caused mainly by the fact that for half of the tested period the DAX was moving faster against the remaining markets. If we divided the sets into two subsets – pre-2003 and post-2003 –, we would find much better results in the second period. So, we will leave this part as an exercise for interested readers, as we will provide a division into the sub-periods in the next out-of-sample forecasting tests. Thus for now the results are clear and we will move on and use them for real forecasting of the market returns.

3.2.2 Cross-Market Predictions

We found that the PX-50, BUX, WIG, and DAX returns are co-moving, so now we are interested if this information can be used for forecasting. The methodology here will be quite different. We will try to forecast the one-day return of the market using the lags of the three remaining markets. For this purpose, we will apply correlation analysis[7] to find which lags influence the returns the most. Then we will use linear OLS estimation and the feedforward neural network again with the best performing logsigmoid transformation function and the Levenberg-Marquardt search algorithm. The following models were developed.[8] (*Table 7*)

$$PX50_{t+1} = \beta_0 + \beta_1 PX50_{t-1} + \beta_2 PX50_{t-5} + \beta_3 BUX_{t-1} + \beta_4 BUX_{t-3} + \beta_5 DAX_t + u_t \quad (0.6)$$

$$BUX_{t+1} = \beta_0 + \beta_1 BUX_{t-3} + \beta_2 BUX_{t-5} + \beta_3 DAX_t + \beta_4 DAX_{t-2} + u_t \quad (0.7)$$

$$WIG_{t+1} = \beta_0 + \beta_1 WIG_t + \beta_2 WIG_{t-5} + \beta_3 PX50_t + \beta_4 PX50_{t-3} + \beta_5 DAX_t + \beta_6 DAX_{t-2} + u_t \quad (0.8)$$

$$DAX_{t+1} = \beta_0 + \beta_1 DAX_{t-3} + \beta_2 DAX_{t-4} + \beta_3 PX50_{t-5} + \beta_4 WIG_{t-4} + \beta_5 BUX_{t-1} + u_t \quad (0.9)$$

As we can see, the PX-50, BUX, WIG, and DAX returns seem to be explained to some extent by their mutual lags. As for the comparison of the autoregressive model with the neural network, the neural network leaves the autoregressive models far behind. As regards explanatory power, the neural network explains 12 %–20 % of the variance of the returns in the individual model, while autoregression explains only 1.5 %–2.34 %. The Schwartz criterion also strongly prefers the networks. So, the implication for modeling is very intuitive – use the linear regression model to identify the significance of the variables and then improve the estimates with neural networks. The reader can observe a very interesting fact – there is no autocorrelation present in the models. The Ljung-box Q statistics were not significant at any level for any $Q(k)$. So the results suggest to us that we could gain some predictive edge from these models. Again, we will be concerned with out-of-sample testing more than in-sample. In *Table 8* we show the results for whole testing period.

[7] We use a sample correlation coefficient – the Pearson product moment correlation coefficient, which is the best estimate of the correlation between the two series – to determine the potential explanatory variables. We pick all variables with a correlation coefficient statistically significant at the 1%, 5%, and 10% levels.

[8] Estimates can be found in *Appendix A* (*Tables 9–12*).

TABLE 7 In-Sample Performance of the Daily Models for Whole Tested Period

in-sample	PX50		BUX		WIG		DAX	
	classical	neural	classical	neural	classical	neural	classical	Neural
Adj *R*-squared	0.02	**0.12**	0.02	**0.20**	0.023	**0.17**	0.019	**0.11**
Schwarz criterion	-6.04	**-8.99**	-5.78	**-8.85**	-6.08	**-9.41**	-5.23	**-7.98**
Ljung-box Q(4)	1.31		5.53		0.96		2.8	
Ljung-box Q(8)	2.99		5.94		1.30		24.69	
Ljung-box Q(12)	3.02		6.53		4.05		30.51*	

Note: *,**,*** significance levels of 1%, 5% and 10% resp.

TABLE 8 Out-of-Sample Performance of the Daily Models for Whole Tested Period

	PX50		BUX		WIG		DAX	
	linear	neural	linear	neural	linear	neural	linear	neural
RMSE	0.09	**0,01**	0,02	**0.01**	0.01	0.01	0.01	0.01
NMSE	0,985	**0.98**	0.99	**0.96**	1.01	0.99	1.01	0.99
D-M(0)		-0.71		-0.26		-0.06		**-1.9****
D-M(1)		-0.71		-0.23		-0.08		**-1.99****
D-M(2)		-0.68		-0.23		-0.08		**-1.91****
D-M(3)		-0.65		-0.24		-0.077		**-1.8****
H-M	1.03***	**1.07***	1.04	**1.06****	0.98*	1**	1.02	**1.07***
P-T	56 %**	**59 %****	52 %	**57 %****	52 %	56 %	53 %	57 %
TC	0.6 %	**1.2 %**	1 %	**1.7 %**	-0.62 %	-0.46 %	-0.45 %	0.2 %

Note: D-M: Diebold-Mariano statistic (*p*-values), H-M: Henriksson – Merton statistic, P-T: Pessaran-Timmerman (SR with p-value), TC – total costs.

*,**,*** significance on 1%, 5% and 10% levels.

In our final tests, Diebold-Mariano tells us that for almost all the series the errors of the linear models and the neural ones are almost identical, while we cannot reject the null hypothesis of equal predictive accuracy for the PX-50, BUX, and WIG. But we can reject the null hypothesis of no predictability for the PX-50, BUX, and DAX at the 1%, 5%, and 1% significance levels, respectively. We also reject the null hypothesis of independence of the directional change of the actual and predicted series for the PX-50 and BUX at the 10% significance level. The implied transaction costs are also in line with the H-M and *P-T* statistics, while they confirm the economic significance. Again, we were not able to gain consistently and significantly better predictive power for all the series tested, even with the use of neural network models. This may imply that the daily European stock market returns are simply unpredictable, as the lags of the surrounding markets helped to explain the variance very little.

4. Conclusion

In this paper, we present a neural network approach and apply it to European stock market return modeling. We show that there is no black box behind the networks, but rather a robust mathematical model, and we view the analysis as a nonparametric econometric method. Thus, we use it for empirical testing on the Czech, Hungarian, and Polish returns from the 1999–2006 period to see if networks will help

us to uncover the possible return-generating process. We also present statistical and economic tests for comparing the models. After the theoretical background is set, we conduct the tests on the daily returns and we find that with the use of neural networks we did not manage to get significantly lower prediction errors according to the Diebold-Mariano test, but we gained some economic significance on the PX-50 and BUX markets, where the direction predictions were significant at the 5% and 10% levels, respectively, and we were able to predict the next day direction with a 56% and 54% probability of correct prediction, respectively.

We left the daily series to conduct the same tests on the weekly ones. The in-sample adjusted R^2 of the neural network was impressive, as it explained 48 % of the PX-50, 15 % of the BUX, 28 % of the WIG, and 34 % of the DAX variance using only lagged explanatory variables. When faced with out-of-sample forecasting, we were able to reject the null hypothesis of equal prediction errors between the linear and neural models with the PX-50 and BUX series at the 5% significance level. Thus, the neural networks had significantly a better forecasting error when we tested the PX-50 and BUX series. We also achieved better economic significance of the models, and we were able to forecast the PX-50, WIG, and DAX with a directional accuracy of 60 %, 58 %, and 58 %, respectively, significant at the 10% level. Also, the implied transaction costs were higher than the real-world transaction costs, which tells us that the predictions are economically significant. In the next part, we used the fact that the tested markets are co-moving. We used Principal Component Analysis to find out whether or not the lagged returns of the surrounding markets have a significant influence on the tested market, i.e., we tested whether the lagged returns of the BUX, WIG, and DAX can be used to explain the PX-50 return. We found that there are significant lags of the surrounding markets for each of the tested markets. We then used these results to model the stock market return using the cross-country lags on the daily data. We got similar results, as we could not reject the hypothesis of equal errors of the linear and neural models for all series except the DAX. Interestingly, neural networks perform significantly better only on the daily DAX returns. We again obtain economic significance on the PX-50, BUX, and DAX daily returns with neural networks. The WIG daily return predictions are again not economically significant.

To sum up the results of the application of neural networks to Central European stock market returns, we would say that the daily returns do not contain significant patterns, as the neural network could not approximate them. It did significantly better in the case of the German DAX, which was basically picked as a benchmark large liquid European stock market. On the other hand, the WIG seems to be completely unpredictable using just the lagged historical returns. On the PX-50, BUX, and DAX markets, the neural network predictions were economically significant. We managed to gain more predictive edge from the weekly returns, while the neural network performed significantly better than linear modeling for PX-50 and BUX prediction, and it provided us with economically significant predictions and an ability to predict the direction with 60% probability. Of course, our findings have strong implications for the markets and traders, but they are still of quite speculative use. Moreover, there are many problems with using these models in real trading. The main drawback is that most of the models tend to predict movements with a lag. This is fine if the markets are steady and the model captures the short-term trends well. But if there are unexpected exogenous moves or crashes of the stock market, the models

very often fail to warn us. In this research, we presented the models using only lagged historical data, and even if we could gain some predictive edge using neural network models, it is clear that further analysis needs to be done. In particular, the use of other variables affecting the price of stocks should be considered, as we see that the data do not explain them well.

APPENDIX

OLS Estimation Results For PX50, BUX, WIG and DAX Models

TABLE 9 PX50 Model

Variable	Coefficient	Std. Error	*t*-Statistic	Prob.
β_0	0.000834	0.000331	2.518340	0.0119
β_1	-0.066742	0.031630	-2.110083	0.0350
β_2	0.063869	0.027743	2.302173	0.0215
β_3	0.064417	0.027921	2.307141	0.0212
β_4	0.077691	0.024692	3.146457	0.0017
β_5	0.058844	0.018724	3.142688	0.0017

TABLE 10 BUX Model

Variable	Coefficient	Std. Error	*t*-Statistic	Prob.
β_0	0.000766	0.000376	2.037470	0.0418
β_1	0.047583	0.028101	1.693296	0.0906
β_2	0.068265	0.027978	2.439997	0.0148
β_3	0.060791	0.021326	2.850566	0.0044
β_4	0.033276	0.021337	1.559500	0.1191
β_5				

TABLE 11 WIG Model

Variable	Coefficient	Std. Error	*t*-Statistic	Prob.
β_0	0.000554	0.000324	1.710250	0.0875
β_1	0.075760	0.032033	2.365101	0.0182
β_2	0.061295	0.027892	2.197569	0.0282
β_3	-0.063423	0.030753	-2.062354	0.0394
β_4	0.053653	0.027227	1.970563	0.0490
β_5	0.044133	0.019863	2.221846	0.0265
β_6	0.044053	0.018288	2.408821	0.0161

TABLE 12 DAX Model

Variable	Coefficient	Std. Error	*t*-Statistic	Prob.
β_0	0.052203	0.028030	1.862370	0.0628
β_1	-0.110060	0.029929	-3.677415	0.0002
β_2	0.067533	0.041382	1.631945	0.1029
β_3	0.085239	0.045611	1.868815	0.0619
β_4	0.077619	0.036835	2.107219	0.0353

REFERENCES

Barkoulas J, Travlos N (1998): Chaos in an emerging capital market? The case of the Athens Stock Exchange. *Applied Financial Economics*, 8:231–243.

Cambazoglu BB (2003): Predicting the IMKB 30 Index. *Dept.of computer Engineering Bilkent University, Ankara, working paper*.

Campbell J, Lo AW, MacKinlay AC (1997): *The Econometrics of Financial Markets*. Princeton University Press, Princeton.

Diebold FX, Mariano R (1995): Comparing Predictive Accuracy. *Journal of Business and Economic Statistic*, 3:253–263.

Filacek J, Kaplička J, Vošvrda M (1998), Testování hypotézy efektivního trhu na BCPP. (in Czech) *Finance a úvěr*, 48(9):554–566.

Henriksson RD, Merton RC (1981): On Market Timing and Investment Performance. II. Statistical Procedures for Evaluating Forecasting Skills. *Journal of Business*, 54:513–533.

Hertz JA, Krogh A, Palmer RG (1991): *Introduction to the Theory of Neural Computation*. Addison-Wesley.

Hornik K, Stinchcombe M, White H (1989): Multifactor feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366.

Hsieh DA (1991): Chaos and Nonlinear Dynamics: Application to Financial Markets. *Journal of Finance*, 46(5):1839–1877.

Hutchinson JM, Lo AW, Poggio T (1994): A Nonparametric Approach to Pricing and Hedging Derivative Securities via Learning Networks. *Journal of Finance*, 49(3):851–889.

Kuan CM, White H (2004): Artificial Neural Networks: An Econometric Perspective. *Econometric Reviews*, 13:1–91.

Levenberg K (1944): A Method for the Solution of Certain Problems in Least Squares. The *Quarterly of Applied Mathematics*, 2:164–168.

Lo AW, MacKinlay AC (1988): Stock Prices Do Not Follow Random Walk: Evidence from a Simple Specification Test. *Review of Financial Studies*, 1:41–66.

Marquardt D (1963): An Algorithm for Least-Squares Estimation of Nonlinear Parameters. *SIAM Journal of Applied Mathematics*, 11(2):431–441.

McNelis PD (2005): *Neural Networks in Finance: Gaining predictive edge in the market*. Elsevier Academic Press advanced finance series.

Mitchell TM (1997): *Machine Learning*, McGraw-Hill.

Mohan N, Jha P, Laha AK, Dutta G (2005): Artificial Neural Network Models for Forecasting Stock Price Index in Bombay Stock Exchange. *IIMA Working Papers*, Indian Institute of Management Ahmedabad.

Nygren K (2004): *Stock Prediction – A Neural Network Approach, Master's Thesis*. Royal Institute of Technology, Sweden.

Peters E (1949): *Fractal Market Analysis: Applying Chaos Theory to Investment and Economics.* John Wiley and Sons.

Pesaran MH, Timmermann A (1992): A Simple Nonparametric Test of Predictive Performance. *Journal of Business and Economic Statistics*, 10:461–465.

Poggio T, Girosi F (1990): Networks for Approximation and Learning. *Proceednigs of The IEEE*, 78(9):1481-1497.

White H (1988): Economic prediction using neural networks: The case of IBM daily stock returns. *IEEE International Conference on Neural Networks*, San Diego, 451–459.

Yao JT, Tan CL, Poh HL (1999): Neural Networks for Technical Analysis: A Study on KLCI. *International Journal of Theoretical and Applied Finance*, 2(2):221–241.

Zikes F (2004): Cross-Country Predictability and Cointegration: The Case of Central-European Stock Markets. In: Blaha, ZS (eds): *Risk Management and Financial Engineering*. Management Press, Praha 2004: 137–149.