Machine learning approaches for financial time series forecasting

Vasily Derbentsev[1[0000-0002-8988-2526]], Andriy Matviychuk[1[0000-0002-8911-5677]], Nataliia Datsenko[1[0000-0002-8239-5303]], Vitalii Bezkorovainyi[1[0000-0002-4998-8385]] and Albert Azaryan[2[0000-0003-0892-8332]]

[1] Kyiv National Economic University named after Vadym Hetman, 54/1 Peremohy Ave., Kyiv, 03057, Ukraine
derbv@kneu.edu.ua, editor@nfmte.com, d_tashakneu@ukr.net, retal.vs@gmail.com

[2] Kryvyi Rih National University, 11 Vitalii Matusevych Str., Kryvyi Rih, 50027, Ukraine
azaryan325@gmail.com

Abstract. This paper is discusses the problems of the short-term forecasting of financial time series using supervised machine learning (ML) approach. For this goal, we applied several the most powerful methods including Support Vector Machine (SVM), Multilayer Perceptron (MLP), Random Forests (RF) and Stochastic Gradient Boosting Machine (SGBM). As dataset were selected the daily close prices of two stock index: SP 500 and NASDAQ, two the most capitalized cryptocurrencies: Bitcoin (BTC), Ethereum (ETH), and exchange rate of EUR-USD. As features we used only the past price information. To check the efficiency of these models we made out-of-sample forecast for selected time series by using one step ahead technique. The accuracy rates of the forecasted prices by using ML models were calculated. The results verify the applicability of the ML approach for the forecasting of financial time series. The best out of sample accuracy of short-term prediction daily close prices for selected time series obtained by SGBM and MLP in terms of Mean Absolute Percentage Error (MAPE) was within 0.46-3.71 %. Our results are comparable with accuracy obtained by Deep learning approaches.

Keywords: financial time series, short-term forecasting, machine learning, support vector machine, random forest, gradient boosting, multilayer perceptron.

1 Introduction

Forecasting financial tine series have been in focus of researchers for a long time. This topic continues to be relevant from both theoretical and applied points of view. Brokers, financial analysts and traders make daily decisions about buying and selling various financial assets, including currency, stocks, bonds and others. To reduce the risk of such transactions and to obtain the expected return on their investments, each of them must analyze a number of factors that affect market conditions and generate upward or downward trends.

In this regard, the problem of developing adequate forecasting approaches is relevant to the scientific community as well as to financial analysts, investors and traders.

There are two main approaches to solving the problem of forecasting financial assets. The first one is to construct a casual model that describes the relationship between the asset's value and other macroeconomic factors. This approach was implemented within the framework of fundamental analysis and based on different mathematical tools, such as econometric modeling and systems of differential equations [13; 17; 28].

Another approach is based on the analysis of past observations selected asset and used variety of technical indicators and oscillators that help predict market trends. This approach has been realized in technical analysis which is actively used now in addition to time series analyses [7; 13; 28]. Within time series framework has been developed manifold class of linear and nonlinear approaches, such as ARIMA-GARCH models [6; 27].

Recent time the methods and algorithms of Machine Learning (ML) which have developed within Data Science paradigm [14; 36] ML have been also applied to forecasting financial and economic time series [2; 12], and various automated trading systems (bots) built on these algorithms began to be used for trading. Results of numerous empirical studies have shown that ML approaches outperform time series models in forecasting different financial assets [10; 18; 22; 26; 31; 38].

The main advantage of ML is that the algorithms themselves interpret the data, so we don't need to perform their initial decomposition. Depending on the purpose of the analysis, these algorithms themselves build the logic of modeling on the basis of available data.

This avoids the complex and lengthy pre-model stage of statistical testing of various hypotheses about studied process. The main hypothesis, in particular, in terms of the purpose of our study, is only the thesis of the ability of ML methods to effectively analyze the financial time series, to identify hidden patterns and time correlations, which are the basis for making qualitative short-term forecasts.

The main goal of our paper is to compare the predictive properties of the most efficient ML algorithms: Artificial Neural Network (ANN), Support Vector Machine (SVM), Random Forest (RF) and Gradient Boosting Machine (GBM) for short-term forecast financial time series (stock indices, currencies and cryptocurrencies). At the same time, as predictors (features) we used only the past values of the studied time series. Our main assumption is that ML methods be able to extract latent patterns from the data, which allows us to make more efficient predictions.

This paper is organized as follows: in Section 2 is represented brief literature review devoted using ML approaches in the field of financial time series forecasting. Section 3 is described the main concept of applied methods. Data description and empirical results are given in Section 4. Concluding remarks and future perspectives are given in Section 5.

2 Brief review recent studies

It should be noted that financial time series forecasting have been studied for a long time. Since, ML approaches proved their efficiency in many areas and became popular; they have been widely used for research financial time series. Numerous articles in scientific journals, reviews, conferences and internet resources are devoted to this topic.

Last five years researchers basically have been focused attention on novel network ML approaches Deep Learning (DL), which includes asset of powerful methods, such as Recurrent Neural Network (RNN), Long-Short Time Memory (LSTM), and Convolutional Neural Network (CNN) and so on [22; 24; 32; 34; 38]. Recently was published detailed overview devoted to using DL approaches in the field of financial forecasting [24]. The main finding this survey is that generally DL framework outperforms time series models and often shows higher accuracy than traditional ML algorithms.

The key advantage of DL models is very powerful in feature learning and selection of input data using a general-purpose learning procedure. But DL models have such disadvantage that it takes much more time to train them, besides, it is a nontrivial problem is tuning hyperparameters. At the same time, traditional ML models often show comparable accuracy in time series forecasting.

As for using ML algorithms in financial forecasting, the most common are Neural Networks (ANNs) of various architecture [1; 8; 10; 20; 21; 33; 38], Support Vector Machines (SVM) [23; 25; 29; 30; 35], and Fuzzy Logic (FL) [25; 39].

The application of these approaches for forecasting task has shown their efficiency for both traditional financial assets [1; 8; 18; 20; 21; 23; 24; 29; 30; 35] and cryptocurrencies [8; 26; 31; 38].

Several studies [1; 8; 20; 21] presented the results that ANNs have better predictive properties then other ML approaches for forecasting financial time series. At the same time, there are a number of research papers (see, for example, Okasha, [29]; Sapankevych and Sankar, [30]; Hitam and Ismail, [19]), which presented results that SVMs have also been proven to outperform other non-linear techniques including neural-network based non-linear prediction techniques such as multi-layer perceptron (MLP).

It should be noted, that much less attention has been paid to another powerful class of ML approaches of designing ensembles Classification and Regression Trees (C&RT): Random Forest (RF) [4; 5] and Gradient Boosting Machine (GBM) [15; 16], which used bagging (RF) and boosting (GBM) technique. Both RF and GBM are powerful methods that can efficient capture complex nonlinear patterns in data.

Thus, Varghade and Patel [35] tested RF and SVM to forecasting stock market index S&P CNX NIFTY. They noted that the Decision Trees model outperforms the SVR, although RF at times is found to overfit the data.

Kumar and Thenmozh [23] explored set of classification models for predicting direction of index S&P CNX NIFTY. Their empirical results suggest that both the SVM and RF outperforms the other classification methods (NN, Linear Discriminant Analysis, Logit), in terms of predicting the direction of the stock market movement, but at the same time SVM it turned out to be more accurate.

Recently there have been appeared several papers devoted to applying ensembles approaches for forecasting cryptocurrency prices [3; 9; 11]. Borges and Neves [3] tested four ML algorithms for prediction price trend: LR, RF, SVM and GBM. All learning algorithms outperform the Buy and Hold investment strategy in cryptomarket. The best result was obtained by ensembles voting (accuracy 59.3%).

Chen et al. [9] applied a set of learning models including RF, XGBoost, Quadratic Discriminant Analysis, SVM and LSTM for Bitcoin 5-minute interval and daily prices. Authors used wide dataset including as features technological, market and trading, socio-media and fundamental factors. Somewhat unexpected was that for daily prices better results were obtained by using statistical methods (average accuracy 65%) unlike ML methods (average accuracy 55.3%). Among the best ML the SVM was the best, with an accuracy of 65.3%.

3 Methodology

In this paper we have been applied supervised ML technique for forecasting financial time series. Consider a sample of pairs of features $\boldsymbol{x} = (x_1, x_2, \dots, x_p, \dots, x_n)$ and the labels *y*: $(\boldsymbol{x}_i, y_i)_{i=1,2,\dots,n}$ length *n*. In our case labels (or target) are values of selected financial assets, and features are only lagged daily values these assets $y_{i-1}, y_{i-2}, \dots, y_{i-p}, i > p$.

Our main goal is to predict future value of target variable on the next time period (next day since we used daily quotes) by using several ML approaches (SVM, ANN, RF and GBM) and compare their forecasting performance.

Thus, our task is to construct some functional (regression) or rule-based (decision tree based) dependence of the form

$$\hat{y}_{n+1} = f(\boldsymbol{x}_j, \boldsymbol{w}_j), \tag{1}$$

where $\boldsymbol{x}_j = (x_i)_{i=1}^{n}, j = 1,2,\dots,k$ are vectors of features; $\boldsymbol{w}_j$ – weights of the features, *n* – total number of samples in dataset; *k* – number of features.

3.1 Support Vector Machine (SVM)

The support vector machine (SVM) is an extension of the support vector classifier that results from enlarging the feature space in a specific way by using kernels function. The main idea of the SVM method is to map the original vectors into a space of a higher dimension and search for a separating hyperplane with a maximum margin in this space. Two parallel hyperplanes are constructed on both sides of the hyperplane separating the classes. The separating hyperplane will be the hyperplane that maximizes the distance to two parallel hyperplanes. The algorithm works under the assumption that the lager difference or distance between these parallel hyperplanes (margine) provides the smaller average error of the classifier.

Support Vector Regression (SVR) is the regression process performed by SVM which tries to identify the hyperplane that maximizes the margin between two classes and minimize the total error. In order for an efficient SVM to be constructed, a penalty of complexity is also introduced, balancing forecasting accuracy and computational performance.

Unlike classic regression problem SVR seeks coefficients that minimize a different type of loss, where only residuals larger in absolute value than some positive constant contribute to the loss function. This is an extension of the margin used in support vector classifiers to the regression setting.

The mathematical formalization of SVR is reduced to the following. Let's regression equation is written in the form

$$a(\boldsymbol{x}) = \langle \boldsymbol{x}, \boldsymbol{w} \rangle - w_0, \tag{2}$$

where $\langle \cdot,\cdot \rangle$ – is operator of inner product; w_0 is a constant.

Then the problem is reduced to minimizing functional:

$$\langle \boldsymbol{w}^T, \boldsymbol{w} \rangle + \frac{1}{2C} \sum_{i=1}^{l} (|\langle \boldsymbol{w}, x_i \rangle - w_0 - y_i| - \varepsilon) \rightarrow \min_{w,w_0}, i = 1,2,\ldots,l, \tag{3}$$

where C is the regularization parameter or penalty coefficient for incorrectly estimating the output associated with input vectors, which also controls the relationship between a smooth boundary; l is the number of samples in training set ($l < n$, as a rule $l \approx 0.7 \div 0.8n$); ε is the margin value.

After changing variables and some algebraic transformations loss function for SVR can be presented in such form:

$$\frac{1}{2} \langle \boldsymbol{w}^T, \boldsymbol{w} \rangle + C \sum_{i=1}^{l} (\xi_i^+ + \xi_i^-) \rightarrow \min_{w,w_0,,\xi_i^+,,\xi_i^-}, i = 1,2,\ldots,l, \tag{4}$$

where $\xi_i^- = (-a(x_i) + y_i - \varepsilon)$, $\xi_i^+ = (a(x_i) - y_i - \varepsilon)$ are slack variables, that allow individual observations to be on the wrong side of the margin or the hyperplane; $K(x_i, x_j)$ is the kernel function. The most commonly used kernel functions are Linear, Polynomial, Gaussian, Radial Based Function (RBF) and so on.

Loss function (4) is minimized under condition

$$\begin{cases} y_i - \varepsilon - \xi_i^- \le \boldsymbol{w}^T K(x_i, x_j) - w_0 \le y_i + \varepsilon + \xi_i^+ \\ \xi_i^+, \xi_i^- \ge 0, i = 1,2,\ldots,l \end{cases}. \tag{5}$$

The Lagrangian of this problem can be expressed in terms of the dual variables λ_i^+, λ_i^-, thus the regression equation on support vectors can be written in the such form:

$$a(\boldsymbol{x}) = \sum_{i=1}^{l} (\lambda_i^+ - \lambda_i^-) K(x_i, x_j) - w_0. \tag{6}$$

3.2 Artificial Neural Network (ANN)

ANNs are the most popular methods of ML. Numerous empirical studies show the efficiency of ANNs in the different fields both for classification and regression problem: pattern recognition, image and voice analysis, machine translation and so on. Several last decides they are widely used for analysis and forecasting financial time series. In [12; 22; 31] it was shown that ANNs have better predictive properties than time series models and other ML algorithms for financial time series forecasting problem.

In this paper we have been used network model of the most common architecture: Multi-Layer Perceptron (MLP) with three layers: input layer, one hidden layer and output layer with one neuron that represent target variable (predicted value). It should be noted that despite simple structure MLP be able to take into account complex patterns in data due using different nonlinear activation functions.

The network output depends on its configuration, weights and activation functions of neurons on the hidden and output layers:

$$\hat{y}_{n+1} = g\left(\sum_{i=1}^{k} w_i f\left(\sum_{j=1}^{p} \omega_{ji} y_{n-j+1} + b_i\right) + b_0\right), \tag{7}$$

where $f(\cdot), g(\cdot)$ – activation functions of neurons of the hidden and input layer, respectively; w_i – the weight of the connections between the i-th neuron of the hidden layer and the output of the network; ω_{ji} – the weight of the connections between the j-th neuron of the input and the i-th neuron of the hidden layers; b_0, b_i – bias neurons of the output and hidden layers.

Network learning consists in finding and setting the neurons weights (synaptic weights) which minimized difference between the target variable and the network output. The search of minimum of the loss function was performed by the gradient descent method, embodied in the back-propagation algorithm.

3.3 Gradient Boosting Machine (GBM)

Boosting is a procedure for sequentially building a composition of machine learning algorithms, when each of them seeks to compensate for the shortcomings of the composition of all previous algorithms. In contrast to bagging, boosting does not use simple voting but a weighted one. The major attractions of boosting are that it is easy to design computationally efficient weak classifiers (as a rule used shallow decision trees). Boosting over decision trees is considered one of the most efficient methods in terms of classification quality.

Gradient Boosting Machine method (GBM) was proposed Friedman [15; 16]. Commonly the basic steps of GBM are the next.

The final classifier $a_N(\boldsymbol{x})$ is constructed as a weighted sum of *N* basic algorithms $h_i(\boldsymbol{x}, \theta)$ (Decision Trees):

$$a_N(\boldsymbol{x}) = \frac{1}{N} \gamma_N \sum_{i=0}^{N} h_i(\boldsymbol{x}, \theta), \tag{8}$$

where θ is vector of adjusted parameters, γ_N is the weight coefficient.

Let's we choose the initial classifier $h_0(\boldsymbol{x}, \theta)$, for example, it may be the median or mean of the time series (target variable).

If we on the *N*–1 step have already built new classifier $a_{N-1}(\boldsymbol{x}_i, \theta)$, then we select the next basic algorithm $h_N(\boldsymbol{x})$ that reduced the error given by previous classifier as much, as possible:

$$\sum_{i=1}^{l}\left[L\left(y_i, a_{N-1}(\mathbf{x}_i,\theta) + \gamma_N h_N(\mathbf{x}_i,\theta)\right)\right] \rightarrow \min_{\gamma_N, h_N}, \tag{9}$$

where $L(\cdot)$ – is loss function.

We can select $h_N(\boldsymbol{x}_i.\theta)$ that minimized sum of squares deviations for all samples in training set

$$h_N(\mathbf{x},\theta) = \underset{h(\mathbf{x},\theta)}{\operatorname{argmin}} \sum_{i=1}^{l}(h(x_i,\theta) - s_i)^2, \tag{10}$$

where s_i is deviations that equal to the anti-gradient of the loss function $L(\cdot)$.

In this way we perform predictions for samples in the training set by using gradient descent in the *l*-dimensional space.

If a new basic algorithm has been found, it is possible to select its coefficient γ_N by analogy with the gradient descent:

$$\gamma_N = \underset{\gamma}{\operatorname{argmin}} \sum_{i=1}^{l} L\left(y_i, a_{N-1}(\mathbf{x}_i,\theta) + \gamma_N h_N(\mathbf{x}_i,\theta)\right), \tag{11}$$

It should be note, that boosting usually does not result the overfitting problem because shallow decision trees are used. These trees have a large bias, but are not inclined to overfitting.

The effective ways to solve this problem is to reduce the step: instead of moving to the optimal direction of the anti-gradient, a shortened step can be taken by

$$a_N(\boldsymbol{x},\theta) = a_{N-1}(\boldsymbol{x},\theta) + \lambda\left(\gamma_N h_N(\boldsymbol{x})\right), \tag{12}$$

where $\lambda \in [0,1]$ is the learning rate.

3.4 Random Forest (RF)

The main concept of the RF is that a composition of weak classifiers can give good results for both classification and regression problems. Proposed by Breiman [4; 5] in 1996 the RF is based on bagging technique (bootstrap aggregation) over decision trees. Bagging reduces the variance of the base algorithms if they are weakly correlated. In RF the correlation between trees is reduced by randomization in two directions.

Firstly, each tree is trained on a bootstrapped subset. Secondly, the feature by which splitting is performed in each node is not selected from all possible features, but only from their random subset of size m. The main distinction between bagging and RF is the choice of these features subset. RF works well when all of the features are at least marginally relevant, since the number of features selected for any given tree is small. Using a small value of m will typically be helpful when we have a large number of correlated predictors.

The RF algorithm generates each of the N trees independently, which makes it very easy to parallelize. For each tree, it constructs a full binary tree of maximum depth. The main concept is that classifiers (trees) do not correct each other's mistakes, but compensate for them when voting. Basic classifiers should be independent and they can be based on different groups of methods or trained on independent datasets. Bagging allows us to reduce prediction error in the case when the variance of the error base method is high.

Thereby efficiency of RF performance is achieved even though some trees will query on useless features and make random predictions. But some of the trees will happen to query on good features and will make good predictions (because the leaves are estimated based on the training data).

If we have enough trees, the random ones will wash out as noise, and only the "good" trees will have an effect on the final result (classification or prediction).

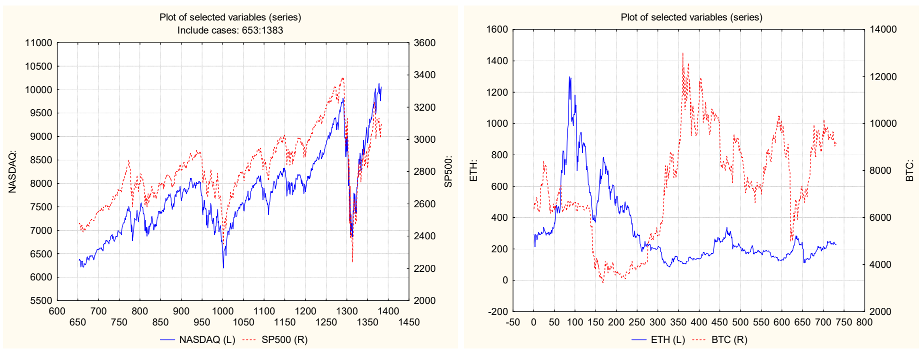
4 Empirical results

4.1 Dataset

To reduce our analysis to the most popular financial assets, we were used daily close prices two stock index: Nasdaq and SP&500, two most capitalized cryptocurrencies: Bitcoin (BTC), Ethereum (ETH), and exchange rate EUR USD. Our initial dataset covers the period from 01/01/2015 to 30/06/2020 for all series (for ETH from 06/08/2020) according to the Yahoo Finance [37].

So, our dataset includes 1384 observations for Nasdaq, 1383 for SP&500, 1434 for exchange rate EUR USD, 2008 for BTC and 1278 for ETH.

It should be noted that selected time series during this period had different type of dynamics due to we can better estimate forecasting performance for ML approaches (see fig. 1).



(a) (b)

Fig. 1. Dynamics of traditional assets (a) and cryptocurrencies (b) from 30/06/2018 to 30/06/2020.

On purpose of training models, fitting and tuning their parameters dataset was divided into the training and test subsets in the ratio of 80% and 20%. Moreover, the last 100 observations (from 22/03/2020 to 30/06/2020) were reserved for validation which was performed by out-of-sample one-step ahead forecast.

Since we focus on ML approach of forecasting financial time series data, the main purpose of our paper is to get the most accurate one-step ahead forecast of daily prices, based on only their past value.

According to some empirical studies devoted forecasting financial time series, there is a seasonal lag which is a multiple of 5 if we use daily observations and a multiple of 7 for cryptocurrencies because the fact that cryptocurrencies are traded 24/7.

For stabilization variance all features were taken in to natural logarithm. This is special case of Box-Cox transform.

4.2 Hyper-parameters tuning

It should be noted that hyper-parameters tuning is an important and sophisticated step of the model design. First of all, it is necessary to choose the functional form of the loss function. In the point of main purposes of our study the quadratic loss, which generally used for solving the regression problem, was selected.

According to our hypothesis regarding lag length as MLP models, we tested the following architectures:

- 7 inputs and from 5 to 14 hidden layer neurons for cryptocurrencies;
- 5 inputs and from 5 to 10 hidden layer neurons.

The most common functions such as logistic, hyperbolic tan, exponential and ReLu were tested as activation functions. Training MLP for each time series and different lag values (number of input neurons) was conducted over 100 epochs, of which the best 5 architectures were selected for each case (in terms of minimum PE error on the test sample and matching the model residuals to normal distribution).

The final prediction for each asset was obtained as the prediction of the ensemble of networks, that is, average of the best 5 corresponding MLP models.

For SVM models we have chosen RBF as a kernel which is the best for regression problem. Regularization parameter was estimated by the greed search in the range from 1 to 15 and it was selected C=10.

Both of tree-based methods (RF, and GBM) based on partitions the data into training and testing sets by randomly selecting cases. We applied in this study stochastic modification of GBM (SGBM) which based on such partition. The training sample is used to fitting models by adding simple trees to ensembles. Testing set is used to validate their performance. For regression tasks validation is usually measured as the average error. We select 30% of the dataset as test cases for both approaches.

Since the RF is not inclined to overfitting, one can choose a large number of trees for the ensemble. We designed RF model with 500 trees. At the same time, in order for the model to be able to describe complex nonlinear patterns in data, it is necessary to use complex trees. So, we have been chosen 15 the maximum number of levels.

Other important parameter for RF is the number of features to consider at each split. As noted in the Section 3.2 it is recommended to choose this value as $m \approx \frac{M}{3}$ (where M is the total number of features) for regression task. We tested different RF models with value m within 8 to 12.

As stop condition for number of trees in SGBM (boosting steps) we took the number of trees at which the error on the test stops decreasing. This is necessary in order to avoid the overfitting. For boosting, unlike the RF, the simple trees are usually used. That's why we fitted maximum number of levels in trees and number of terminal nodes by the criteria of lowest average squared error on both training and test samples.

For GBM an important parameter is a learning rate (shrinkage). Regularization by shrinkage consists in modifying the update rule (12) by tuning λ. We selected this value on the grid search according to minimum prediction error on the test set. The final values of hyper-parameters setting are reported in table 1.

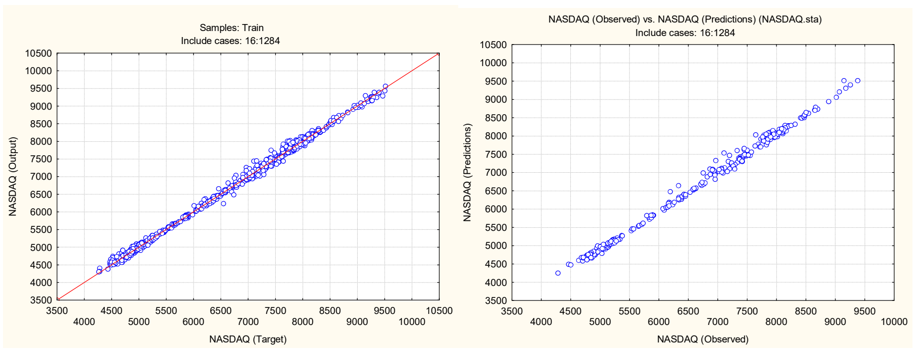
Table 1. Final hyper-parameters setting for RF and SGBM**.**

Parameters	**RF**	**GBM**
Loss-function	quadratic	quadratic
Training / test subsamples proportion, %	70/30	70/30
Random subsample rate	0.7	0.7
Maximum number of trees in ensemble	500	400
Maximum number of levels in trees	10	5
Maximum number of features to consider at each split	12	-
Maximum number of terminal nodes in trees	150	15
Minimum samples in child nodes	5	-
Learning rate (shrinkage)		0.1

4.3 Forecasting performance

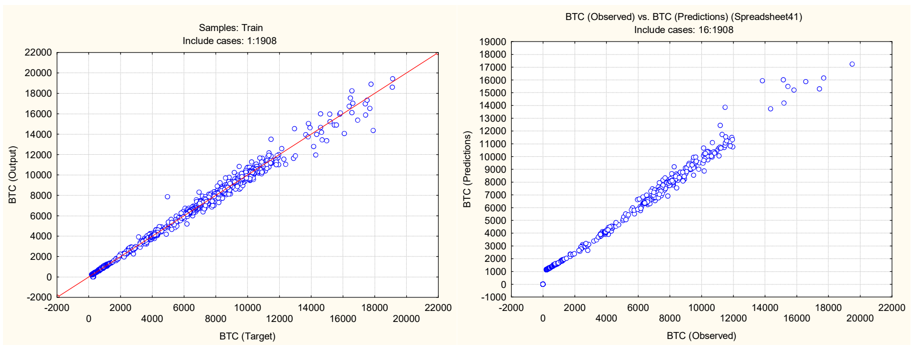
The short-term forecasts for selected time series were made for absolute values of prices (log prices). The target variable is the prediction the value of close prices for each series in the next time period (day) although we used daily observation. All models were trained with the same set of features.

On figures 2-3 were shown quality models fitting for BTC, and NASDAQ obtained by MLP and SVM. figures 4-5 presented results for RF, and SGBM respectively.



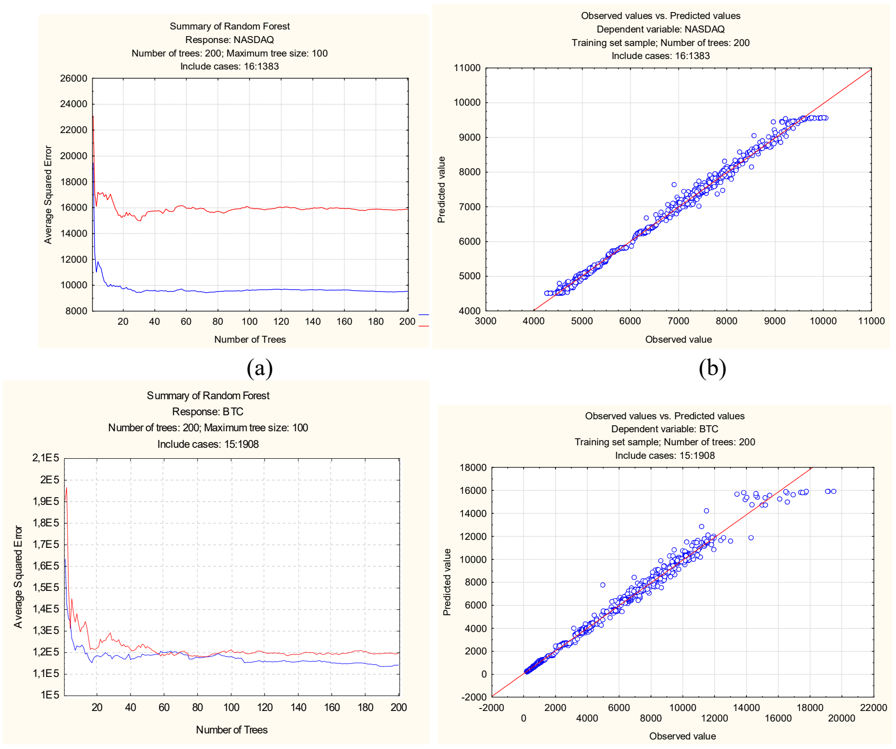
(a) (b)

Fig. 2. Fitting accuracy on the training and test subsets for NASDAQ: (a) MLP, (b) SVM.



(a) (b)

Fig. 3. Fitting accuracy on the training and test subsets for BTC: (a) MLP, (b) SVM.



(c) (d)

Fig. 4. Fitting accuracy on the training and test subsets for RF: (a, b) NASDAQ, (c, d) BTC.

These graphs characterize the dependence of the predicted values (vertical axis) on the actual data (horizontal axis) on the test set and allow us to visually determine the quality of the fitting.

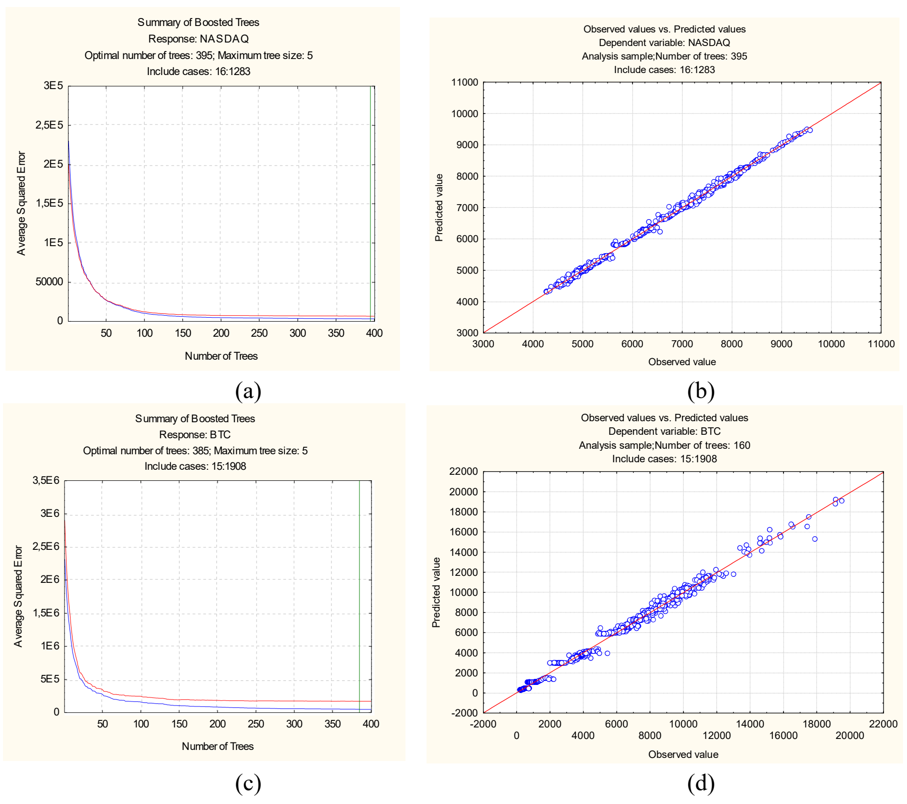


Fig. 5. Fitting accuracy on the training and test subsets for SGBM: (a, b) NASDAQ, (c, d) BTC.

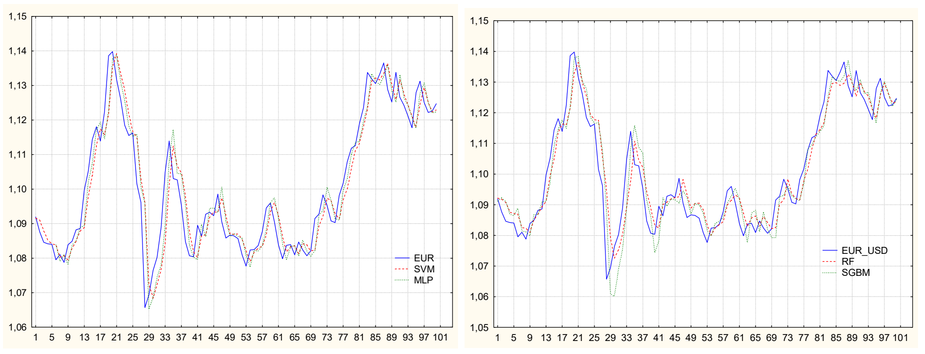
Figures 4-5 shows both the dependence of the predicted values (vertical axis) on the actual data (horizontal axis) (graphs (b, d)) and dependence of models fitting quality on number of trees in ensemble for RF (fig. 4) and SGBM (fig. 5) on the training and test subsets (graphs (a, c)) for selected assets (NASDAQ, BTC).

Forecasting results for last 100 observations (hold-out dataset) for all models and selected time series are shown on fig. 6-9. On fig. 6-10 (a) represented results obtained by SVM and MLP, on fig. 6-10 (b) results for RF and SGBM.

Analysis of the graphs allows us to conclude that SGBM and MLP well approximate time series dynamics, but one can see a certain delay in the model graphs in comparison to real data. RF and SVM showed good approximation not for all time series.

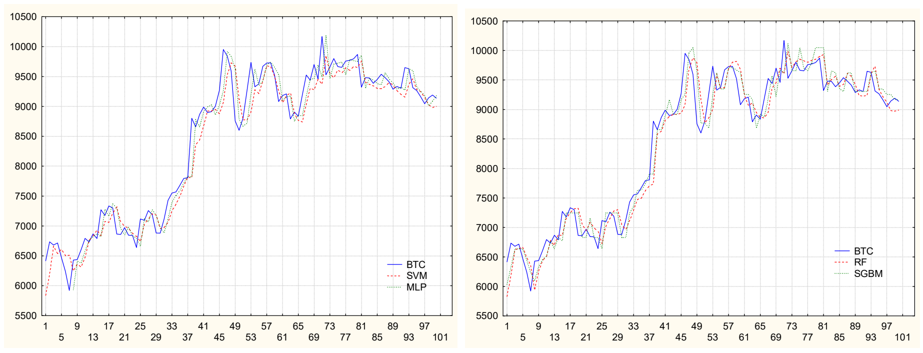
Summary accuracy results in terms of MAPE and RMSE metrics are shown in table 2.

Thus, we can conclude MLP, SVM and SGBM methods have the same order of accuracy for the out-of-sample dataset prediction, although boosting also was somewhat more accurate. The best prediction performance is produced by SGBM for EUR-USD – 0.46 % (MAPE), and the best result for NASDAQ also provided SGBM – 2.38%. For BTC better performance shown SVM but MLP outperformed other models for SP&500.



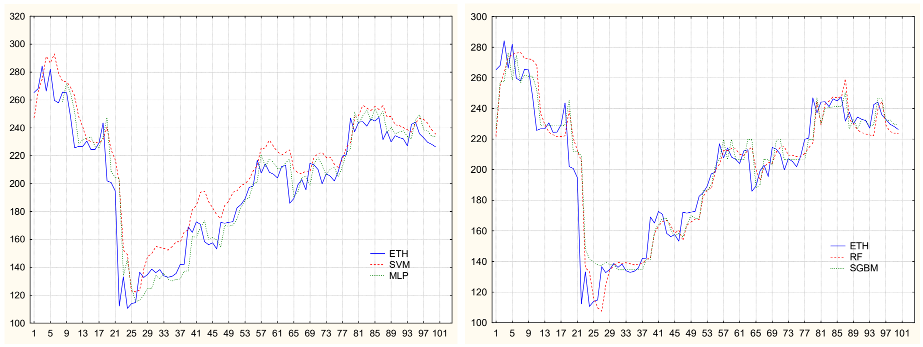
(a) (b)

Fig. 6. Out of sample prediction EUR/USD.



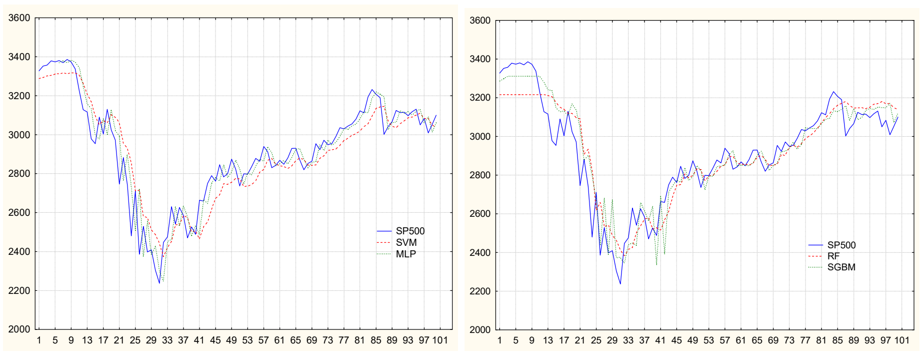
(a) (b)

Fig. 7. Out of sample prediction BTC /USD.



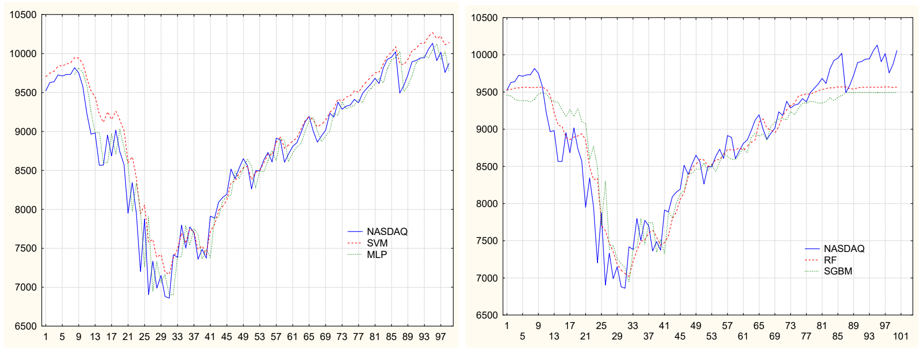
(a) (b)

Fig. 8. Out of sample prediction ETH/USD.



(a) (b)

Fig. 9. Out of sample prediction S&P 500.



(a) (b)

Fig. 10. Out of sample prediction NASDAQ.

Table 2. Out-of-sample accuracy forecasting performance results.

	SGBM		RF		SVM		MLP	
	MAPE, %	RMSE	MAPE, %	RMSE	MAPE, %	RMSE	MAPE, %	RMSE
EUR/USD	0.46	0.0656	0.47	0.0067	**0.40**	0.0065	0.45	0.0067
BTC/USD	2.44	283.6	2.65	321.8	**1.03**	106.5	2.25	274.5
ETH/USD	**5.09**	15.6	5.17	15.89	8.36	260.4	5.25	14.83
S&P 500	2.54	97.99	2.85	108.9	2.91	106.5	**2.35**	91.2
NASDAQ	**2.38**	289.3	2.51	289.1	2.77	340.3	2.23	257.6

It should be noted that our results are comparable with accuracy obtained by Deep learning approaches [24]. Therefore, using both tree-based ensembles, ANNs and SVM are powerful enough forecasting tools for financial time series.

5 Conclusion and discussion

Our research has shown efficiency of using ML approaches to predicting financial time series. According to our results, the out of sample accuracy of short-term forecasting daily quotes obtained by SGBM has the same rate as MLP and SVM. In terms of MAPE for selected time series it was within 0.46-5.9 %. Moreover, for NASDAQ and ETH SGBM outperformed other approaches. The worse results were obtained by RF which was used as a baseline.

At the same time all models showed the worst results for ETH, accuracy rate (MAPE) turned out in the range from 5.9 (SGBM) to 8.38 % (RF).

By designing models, we explored different sets of features: from 5 to 15 lags of target variable (from 7 to 14 for cryptocoins). Our final dataset contained only past values of target variable with 14 and 15 lag depth. In this case larger dataset provided better training for all models and given more efficient results.

It should be noted that we used a minimal dataset - only lag values of the studied series (closing prices). In our opinion, forecasting accuracy can be improved by including additional features, for example, open, max, min and average prices, fundamental variables, different indicators and oscillators, such as, Price rate-of-change, Relative strength index, and so on.

Future research should extend by investigating of the prediction power of described ML approaches by using additional features. In the conclusion, we note that the proposed methodology by the development of combined ensemble of C&RT with other powerful ML models, such as NN and SVM is a promising approach to forecasting financial time series. Moreover, it seems to us promising to use DL approaches for features selection and making prediction.

References

1. Bahrammirzaee, A.: A comparative survey of artificial intelligence applications in finance: artificial neural networks, expert system and hybrid intelligent systems. Neural Computing and Applications **19**(8), 1165–1195 (2010). doi:10.1007/s00521-010-0362-z
2. Bontempi, G., Taieb S., Borgne Y.: Machine Learning Strategies for Time Series Forecasting. Business Intelligence. Lecture Notes in Business Information Processing **138**, 62–77 (2013). doi:10.1007/978-3-642-36318-4_3
3. Borges, T.A., Neves, R.N.: Ensemble of Machine Learning Algorithms for Cryptocurrency Investment with Different Data Resampling Methods. Applied Soft Computing **90**, 106187 (2020). doi:10.1016/j.asoc.2020.106187
4. Breiman, L., Friedman, H., Olshen, R.A., Stone, C.J.: Classification and Regression Trees. Chapman & Hall/CRC, Boca Raton (1984)
5. Breiman, L.: Random Forests. Machine Learning **45**, 5–32 (2001). doi:10.1023/A:1010933404324
6. Brockwell, P.J., Davis, R.A.: Introduction to Time Series and Forecasting, 3rd edn. Springer International Publishing, New York (2016). doi:10.1007/978-3-319-29854-2
7. Brooks, A.: Trading Price Action Trends: Technical Analysis of Price Charts Bar by Bar for the Serious Trader. John Wiley & Sons, New Jersey (2012)

8. Caporale, G.M., Plastun, A., Oliinyk, V.: Bitcoin fluctuations and the frequency of price overreactions. Financial Markets and Portfolio Management **33**, 109–131 (2019). doi:10.1007/s11408-019-00332-5
9. Chen, Z., Li, C., Sun, W.: Bitcoin price prediction using machine learning: An approach to sample dimension engineering. Journal of Computational and Applied Mathematics **365**, 112395 (2020). doi:10.1016/j.cam.2019.112395
10. Derbentsev, V., Datsenko, N., Stepanenko, O., Bezkorovainyi, V.: Forecasting Cryptocurrency Prices Time Series Using Machine Learning. CEUR Workshop Proceedings **2422**, 320–334 (2019)
11. Derbentsev, V., Matviychuk, A., Soloviev, V.N.: Forecasting of Cryptocurrency Prices Using Machine Learning. In: Pichl, L., Eom, C., Scalas, E., Kaizoji, T. (eds.) Advanced Studies of Financial Technologies and Cryptocurrency Markets, pp. 211–231. Springer, Singapore (2020). doi:10.1007/978-981-15-4498-9_12
12. Di Persio, L., Honchar O.: Multitask Machine Learning for Financial Forecasting. International Journal of Circuits, Systems and Signal Processing **12**, 444–451 (2018)
13. Eiamkanitchat, N., Moontuy, T., Ramingwong, S.: Fundamental analysis and technical analysis integrated system for stock filtration. Cluster Computing **20**, 883–894 (2017). doi:10.1007/s10586-016-0694-2
14. Flach, P.: Machine Learning: The Art and Science of Algorithms that Make Sense of Data. Cambridge University Press, Cambridge (2012)
15. Friedman, J.H.: Greedy Function Approximation: A Gradient Boosting Machine. The Annals of Statistics **29**(5), 1189–1232 (2001)
16. Friedman, J.H.: Stochastic Gradient Boosting. Computational Statistics & Data Analysis **38**(4), 367–378 (2002). doi:10.1016/S0167-9473(01)00065-2
17. Glabadanidis, P.: Market Timing and Moving Averages: An Empirical Analysis of Performance in Asset Allocation. Palgrave Macmillan, New York (2015). doi:10.1057/9781137359834
18. Hamid, S.A., Habib, A.: Financial Forecasting with Neural Networks. Academy of Accounting and Financial Studies Journal **18**(4), 37–55 (2014)
19. Hitam, N.A., Ismail, A.R.: Comparative Performance of Machine Learning Algorithms for Cryptocurrency Forecasting. Indonesian journal of electrical engineering and computer science **11**(3), 1121–1128 (2018). doi:10.11591/ijeecs.v11.i3.pp1121-1128
20. Kara, Y., Boyacioglu, M., Baykan, Ö.K.: Predicting direction of stock price index movement using artificial neural networks and support vector machines: The sample of the Istanbul Stock Exchange. Expert Systems with Applications **38**(5), 5311–5319 (2011). doi:10.1016/j.eswa.2010.10.027
21. Kourentzes, N., Barrow, D.K., Crone, S.F.: Neural network ensemble operators for time series forecasting. Expert Systems with Applications **41**(9), 4235-4244 (2014). doi:10.1016/j.eswa.2013.12.011
22. Kumar, D., Rath, S.K.: Predicting the Trends of Price for Ethereum Using Deep Learning Technique. In: Dash, S., Lakshmi, C., Das, S., Panigrahi, B. (eds.) Artificial Intelligence and Evolutionary Computations in Engineering Systems. Advances in Intelligent Systems and Computing, vol. 1056, pp. 103–114. Springer, Singapore (2020). doi:10.1007/978-981-15-0199-9_9
23. Kumar, M., Thenmozhi, M.: Forecasting Stock Index Movement: A Comparison of Support Vector Machines and Random Forest. Paper presented at Indian Institute of Capital Markets 9th Capital Markets Conference (2006). doi:10.2139/ssrn.876544

24. Liu, W., Wang, Z., Liu, X., Zeng, N., Liu, Y., Alsaadi, F.E.: A survey of deep neural network architectures and their applications. Neurocomputing **234**, 11-26 (2017). doi:10.1016/j.neucom.2016.12.038
25. Matviychuk, A.: Fuzzy logic approach to identification and forecasting of financial time series using Elliott wave theory. Fuzzy Economic Review XI(2), 51–68 (2006). doi:10.25102/fer.2006.02.04
26. McNally, S., Roche, J., Caton, S.: Predicting the price of Bitcoin using Machine Learning. In: 2018 26th Euromicro International Conference on Parallel, Distributed and Network-based Processing (PDP), 21-23 March 2018, Cambridge, UK(2018). doi:10.1109/PDP2018.2018.00060
27. Mills, T.C.: Forecasting Financial Time Series. In: Clements, M.P., Hendry, D.F. (eds.) The Oxford Handbook of Economic Forecasting. Oxford University Press, Oxford (2011). doi:10.1093/oxfordhb/9780195398649.013.0019
28. Nti, I.K., Adekoya, A.F., Weyori, B.A.: A systematic review of fundamental and technical analysis of stock market predictions. Artificial Intelligence Review **53**, 3007–3057 (2020). doi:10.1007/s10462-019-09754-z
29. Okasha, M.K.: Using Support Vector Machines in Financial Time Series Forecasting. International Journal of Statistics and Applications **4**(1), 28–39 (2014). doi:10.5923/j.statistics.20140401.03
30. Sapankevych, N.I., Sankar, R.: Time Series Prediction Using Support Vector Machines: A Survey. IEEE Computational Intelligence Magazine **4**(2), 24–38 (2009). doi:10.1109/MCI.2009.932254
31. Saxena, A., Sukumar, T.R.: Predicting bitcoin price using lstm And Compare its predictability with ARIMA model. International Journal of Pure and Applied Mathematics **119**(17), 2591–2600 (2018)
32. Semerikov, S.O., Teplytskyi, I.O., Yechkalo, Yu.V., Kiv, A.E.: Computer Simulation of Neural Networks Using Spreadsheets: The Dawn of the Age of Camelot. CEUR Workshop Proceedings **2257**, 122–147 (2018)
33. Sezer, O.B., Gudelek, M.U., Ozbayoglu, A.M.: Financial time series forecasting with deep learning : A systematic literature review: 2005–2019. Applied Soft Computing **90**, 106181 (2020). doi:10.1016/j.asoc.2020.106181
34. Tarasenko, A.O., Yakimov, Y.V., Soloviev, V.N.: Convolutional neural networks for image classification. CEUR Workshop Proceedings **2546**, 101–114 (2019)
35. Varghade, P., Patel, R.: Comparison of SVR and Decision Trees for Financial Series Prediction. IJACTE **1**(1), 101–105 (2012)
36. Volkova, N.P., Rizun, N.O., Nehrey, M.V.: Data science: opportunities to transform education. CEUR Workshop Proceedings **2433**, 48–73 (2019)
37. Yahoo Finance: Stock Market Live, Quotes, Business & Finance News. https://finance.yahoo.com (2020). Accessed 30 Jun 2020
38. Yao, Y., Yi, J., Zhai, S., Lin, Y., Kim, T., Zhang, G., Lee, L.Y.: Predictive Analysis of Cryptocurrency Price Using Deep Learning. International Journal of Engineering & Technology **7**(3), 258–264 (2018). doi:10.14419/ijet.v7i3.27.17889
39. Zarandi, M.H.F., Rezaee, B., Turksen, I.B., Neshat, E.: A type-2 fuzzy rule-based experts system model for stock price analysis. Expert Systems with Applications **36**(1), 139–154 (2009). doi:10.1016/j.eswa.2007.09.034