Chaos, Solitons and Fractals 133 (2020) 109641

Contents lists available at ScienceDirect

Chaos, Solitons and Fractals

Nonlinear Science, and Nonequilibrium and Complex Phenomena

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

Intelligent forecasting with machine learning trading systems in chaotic intraday Bitcoin market

Salim Lahmiri[a], Stelios Bekiros[b,c,*]

[a] John Molson School of Business, Concordia University, Canada
[b] European University Institute, Florence, Italy
[c] Athens University of Economics & Business, Athens, Greece

ARTICLE INFO

Article history:
Received 27 December 2019
Accepted 19 January 2020

Jel classification:
Machine learning
Intraday trading
Bitcoin
Forecasting
90.01 Social Phenomena

ABSTRACT

Due to the remarkable boost in cryptocurrency trading on digital blockchain platforms, the utilization of advanced machine learning systems for robust prediction of highly nonlinear and noisy data, gains further popularity by individual and institutional market agents. The purpose of our study is to comparatively evaluate a plethora of Artificial Intelligence systems in forecasting high frequency Bitcoin price series. We employ three different sets of models, i.e., statistical machine learning approaches including support vector regressions (SVR) and Gaussian Poisson regressions (GRP), algorithmic models such as regression trees (RT) and the *k*-nearest neighbours (kNN) and finally artificial neural network topologies such as feedforward (FFNN), Bayesian regularization (BRNN) and radial basis function networks (RBFNN). To the best of our knowledge, this is the first time an extensive empirical investigation of the comparative predictability of various machine learning models is implemented in high-frequency trading of Bitcoin. The entropy analysis of training and testing samples reveals long memory traits, high levels of stochasticity, and topological complexity. The presence of inherent nonlinear dynamics of Bitcoin time series fully rationalizes the use of advanced machines learning techniques. The optimal parameter values for SVR, GRP and kNN are found via Bayesian optimization. Based on diverse performance metrics, our results show that the BRNN renders an outstanding accuracy in forecasting, while its convergence is unhindered and remarkably fast. The overall superiority of artificial neural networks is due to parallel processing features that efficiently emulate human decision-making in the presence of underlying nonlinear input-output relationships in noisy signal environments.

1. Introduction

About ten years ago, the cryptocurrency was introduced as an alternative investment within the framework of a peer-to-peer digital money system. Interestingly though, in recent years, the volume of daily operations keeps dramatically increasing and the cryptocurrency total market capitalization is approximately 202 billion USD [1]. In this regard, various studies have been conducted to investigate the dynamics of digital markets, including mapping cryptos to complexity-entropy causality planes [2], agglomerative hierarchical clustering [3], volatility modeling [4,5], multifractal analysis [6-9], tail-risk vulnerability [10], and volatility forecasting [11]. Recently, the accurate prediction of cryptocurrencies is a point of special interest for scholars and quantitative traders who focus on modeling time series in order to generate profits. In the literature, various predictive models have been implemented to predict daily cryptocurrency prices including, deep learning [12,13], α-Sutte indicators and ARIMA [14], dynamic model selection and averaging [15], graph chainlets [16], neuro-fuzzy systems [17], vector autoregressive models [15], and light gradient boosting machine techniques [18]. However, there is still opportunity to implement other sophisticated methods to achieve improvements in predictions, such as advanced hybrid machine learning approaches.

Our research implements and compares the performance of seven machine learning systems in forecasting ultra high frequency Bitcoin data. We employ two statistical models, two algorithmical models, and three artificial intelligence approaches. The statistical machine learning models include support vector regression (SVR) and Gaussian regression Poisson (GRP) models, whilst the algorithmic models include regression trees (RT) and the *k*-nearest neighbors (*k*NN) algorithm. Lastly, artificial intelligence methods involve feedforward neural networks (FFNN), Bayesian regularization network (BRNN), and radial basis function networks (RBFNN). Overall, seven separate machine learning predictive systems are

* Corresponding author.
E-mail address: stelios.bekiros@eui.eu (S. Bekiros).

https://doi.org/10.1016/j.chaos.2020.109641
0960-0779/

employed toward efficient forecasting of intra-day Bitcoin prices. In particular, the support vector regression (SVR) [19] maps the input-output data onto a nonlinear plane. The main advantage of the SVR is the application of the structural risk minimization principle, namely the minimization of an upper bound of the generalization error rather than implementing an empirical risk minimization of the training error [19], hence theoretically achieves a global optimum. In addition, the SVR can provide good generalization results even when the data sample is small [19] while at the same time it is resistant to over-fitting. On the other hand, the Gaussian process regression [20] simply implements flexible, and fully probabilistic prediction. The regression trees [21] are used to mimic human level thinking thus allow seeing the logic decision making. They are easy to understand and interpretations can be made whenever needed. In addition, they are based on tree-building algorithms, which are a set of if-then rules to split data and perform nonlinear mapping upon the input-output data plane. Furthermore, they are scalable to large problems and can deal with small data sets. Moreover, the *k*NN algorithm [22] is an instance-based *lazy* learning algorithm used to learn complex target functions quickly without losing information. Finally, the FFNN [23], BRNN [24], and RBFNN [25] are three different topologies of artificial neural networks used to emulate information processing as performed inside human cortex. They comprise parallel information processing systems robust to noisy data and capable of approximating nonlinear and nonstationary signals. The FFNN is a complex multi-layer perceptron trained by computing the gradient of the objective function with respect to parameters/weights. The information in the input layer is processed via an activation function within each neuron. Then, the processed information is propagated through layers to obtain the final output. The BRNN is a special case of FFNN whereby a regularization task is utilized to set the output less sensitive to outliers, completely smooth and generic, hence less likely to overfit. Finally, the RBFNN is a flexible system to model complex dynamic signals with an ability to learn local features, thus enhancing system stability.

The main contributions of our study are as follows: firstly, we attempt to forecast Bitcoin intra-day price time series by using a plethora of different predictive systems that belong to three distinct classes in machine learning literature. In particular, we make use of statistical machine learning models (SVR, GRP), algorithmic techniques (RT, *k*NN) and AI-based topologies (MLNN, BRNN, RBFNN). Although these machine learning systems are well known to be effective in forecasting, they have not yet been evaluated in forecasting Bitcoin ultra-high frequency datasets. Thirdly, Bayesian optimization [26] is employed in order to find optimal values for the SVR, GRP, and *k*NN algorithms. Fourthly, we employ nonlinear statistics calculated from training and testing datasets including Hurst's exponents [27], sample entropy [28], Lempel-Ziv complexity [29] and Kolmogorov complexity metrics [30]. In this regard, we accurately measure the comparative performance of each learning method.

The rest of the paper comprises firstly Section 2 which describes in detail the machine learning approaches and introduces various performance nonlinear statistical metrics. Next, Section 3 presents the data and provides empirical and simulation results. Finally, Section 4 concludes with further inference.

2. Machine learning approaches

2.1. Support vector regression (SVR) modeling

Let $\{(\boldsymbol{x}_k, \boldsymbol{y}_k)\}_{k=1}^{N}$ denote the *k*th input vector $\mathbf{x}$ of the *k*th training pattern and $\mathbf{y}_k$ represent its corresponding desired output. Then, the regression function *f* which is depicted by a linear SVR [19] is expressed as follows:

$$f(\boldsymbol{x}) = \boldsymbol{\omega}\boldsymbol{x}^T + b \tag{1}$$

where $\boldsymbol{x}$, $\boldsymbol{\omega}=(\omega_1, \omega_2,\ldots, \omega_n)\in\Re^n$, $b\in\Re$ and *T* are respectively the input vector, the weight vector, the intercept, and transpose operator. The optimization problem for training the linear SVR is given by:

$$Minimize\ \frac{1}{2}\omega^2 + C\sum_{k=1}^{N}\left(\xi_k + \xi_k^*\right) \tag{2}$$

Subject to,

$$y_k - \boldsymbol{\omega}\boldsymbol{x}_k^T - b \le \varepsilon + \xi_k \ and\ \boldsymbol{\omega}\boldsymbol{x}_k^T + b - y_k \le \varepsilon + \xi_k^* \tag{3}$$

where *C* is the penalty coefficient for incorrectly estimating the output associated with input vectors, $\varepsilon > 0$ is the regularization factor that weights the trade-off between the **y** estimated value and the target value, and ξ and ξ^* are slack variables, and $k = 1,\ldots, N$. Briefly speaking, the nonlinear support vector regression (SVR) [19] seeks to solve the following nonlinear regression problem:

$$f(\boldsymbol{x}) = \boldsymbol{\omega}\ \varphi(\boldsymbol{x})^T + b = \sum_{k=1}^{N}\left(\alpha_k - \alpha_k^*\right)\ \varphi(\boldsymbol{x}_k)\varphi(\boldsymbol{x}_k)^T + b \tag{4}$$

where $\varphi(\mathbf{x})$ denotes a mapping function that maps the input vector $\mathbf{x}$ into a higher dimensional feature space, and where α and α^* are the Lagrange multipliers. The inner product of functions $\varphi(\mathbf{x})$ and $\varphi(\mathbf{x})^T$ can be replaced by a kernel function $K(\bullet)$. Thus, the general form of the SVR is given by:

$$f(\boldsymbol{x}) = \sum_{k=1}^{N}\left(\alpha_k - \alpha_k^*\right)\ K(\boldsymbol{x}, \boldsymbol{x}_k) + b \tag{5}$$

In this study, the polynomial kernel is $K(x,x_i) = (xx_i+1)^d$ where *d* is set to three.

2.2. Gaussian process regression

Gaussian process regression [20] is a prevailing prediction method for accurate function approximation, simple to implement, flexible, and fully probabilistic. In this framework, the goal is to approximate a mapping function $f(\mathbf{x}_i)$ used to transform the input vector x_i into the target value y_i given by $y_i = f(\mathbf{x}_i)+\varepsilon_i$, where ε_I is a Gaussian noise with zero mean and variance σ_ε^2. Then, the observed targets can be expressed by a Gaussian distribution $y \sim \mathbf{N}(\mathbf{0},\mathbf{K}(\mathbf{X},\mathbf{X})+ \sigma_\varepsilon^2\mathbf{I})$, where $\mathbf{X}$ indicates the set having all input points x_i and $\mathbf{K}(\mathbf{X},\mathbf{X})$ represents the covariance matrix computed using a specified covariance function. Gaussian kernels are utilized as most popular covariance functions [20] and are expressed as follows:

$$k(\boldsymbol{x}_p, \boldsymbol{x}_q) = \sigma_s^2 exp\left(-\frac{1}{2}(\boldsymbol{x}_p - \boldsymbol{x}_q)^T\boldsymbol{W}(\boldsymbol{x}_p - \boldsymbol{x}_q)\right) \tag{6}$$

where σ_s^2refers to the signal variance and $\mathbf{W}$ corresponds to the width of the Gaussian kernel. Then, the predicted mean value $f(\mathbf{x}_*)$ with the corresponding variance $V(\mathbf{x}_*)$ are expressed as follows:

$$f(\boldsymbol{x}_*) = k_*^T\left(\boldsymbol{K} + \sigma_\varepsilon^2\boldsymbol{I}\right)^{-1} y = k_*^T\alpha \tag{7}$$

$$V(\boldsymbol{x}_*) = k(\boldsymbol{x}_*, \boldsymbol{x}_*) - k_*^T\left(\boldsymbol{K} + \sigma_\varepsilon^2\boldsymbol{I}\right)^{-1}k_* \tag{8}$$

where $k_* = k(\mathbf{X},x_*)$, $\mathbf{K} = \mathbf{K}(\mathbf{X},\mathbf{X})$, and α represents the prediction output.

2.3. Regression trees

A regression tree model is a non-parametric method for predicting a continuous dependent variable whereby the data is partitioned into nodes based on its conditional responses to the predictors. For instance, the regression tree model with ***t*** terminal

nodes calculates the conditional distribution of $\boldsymbol{y}$ given $\mathbf{x}$, where $\mathbf{x}$ represents a vector of predictors; $x = (x_1,x_2,x_3,\ldots,x_n)$. A parameter $\Phi = (\varphi_1,\varphi_2,\varphi_3,\ldots,\varphi_t)$ links the factor value $\varphi_i(i = 1,2,3,\ldots,t)$ with the ith terminal node. The partitioning process finds the variable $\mathbf{x}$ that provides best partition into child nodes. The conditional distribution $f(y|\phi i)$ *of* $y|x$ indicates the occurrence that the vector $\mathbf{x}$ falls within the region corresponding to the ith terminal node. After splitting each independent variable on several points, the sum of squared errors (SSE) for each group of nodes is computed. Then, the **SSE** is minimized such that each group of data gets a value used for making the predictions. The entire procedure is reiterated so as all group of nodes obtain a value.

2.4. The kNN algorithm

The kNN regression-based algorithm [22] is nonparametric and does not make any assumption regarding the data generating process. The main merit is that it can learn a complex function quickly without losing information. For a given input x of training data, K observations with x_i in the proximity, are taken into account and the average of the response of those K independent variables produces $\hat{y}$ as follows:

$$\hat{y}(x) = \frac{1}{k}\sum_{x_i \in N_{k(x)}} y_i \tag{9}$$

where $N_k(x)$ denotes K closest points in the neighborhood of x. In practice, various distance metrics can be used to evaluate the closeness between points. In our current work, the Euclidean distance is adopted. In general, it is efficient to assign different weights to the variables around the neighborhood. These weights are determined by using a density function. Accordingly, the Gaussian density function is implemented.

2.5. Feedforward neural network

The feedforward neural network belongs to the type of supervised learning systems trained by the gradient descent algorithm to reduce a given error function. Its main merit is the capability of approximating nonlinear functions even if the data is noisy. The topology of the feedforward neural network consists of an input layer used to receive signals, a hidden layer for processing them, and an output layer for computing the predicted temporal signal. The final output has the following form:

$$O_k = f\left(\sum_{j=1}^{S} f\left(\sum_{i=1}^{n} x_i w_{ij}\right) w_{jk}\right) \tag{10}$$

where n is the number of input units, s the number of neurons in the hidden layer, m the number of output neurons, w_{ij} denotes the connection strength between units i and j, and k denotes the number of output units. In this work, the objective function minimized is simply the sum of squared errors between the true and predicted values. In addition, the network is trained with the Levenberg-Marquardt (L-M) numerical algorithm with the weights adjusted as follows:

$$\Delta w = w_{k+1} - w_k = -\left[J^T J + \mu I\right]^{-1} J^T e \tag{11}$$

where J is the Jacobian matrix, e is a vector of network errors (for instance, SSE), and μ is an adaptive parameter. The L-M algorithm approximates the Hessian matrix by computing the Jacobian matrix which is less complex, than directly computing the Hessian. As a result, the L-M numerical algorithm is fast. The parameter μ is set to 0.001.

2.6. Bayesian regularization neural network

The Bayesian regularization network [24] integrates the properties of fast convergence similarly to conventional feedforward neural networks, and prior information of Bayesian statistics. The main role of the Bayesian regularization neural network is to reduce the possible effects of overfitting. In this regard, the Bayesian regularization adds an additional term to the sum of squared error of the neural network model, and eventually the goal is to minimize such quantity. The Bayesian regularization adds an additional term to this equation as follows:

$$F = \beta E_D + \alpha E_w \tag{12}$$

where F is the objective function, E_D is the sum of squared errors, E_w is sum of the square of the network weights, and α and β the regularization parameters. In the Bayesian neural network framework, the weights are considered to be random variables and the density function of the network weights is calculated following the rule of Bayes:

$$P(w|D,\alpha,B,M) = \frac{P(D|w,\beta,M)P(w|\alpha,M)}{P(D|\alpha,\beta,M)} \tag{13}$$

where w is the vector of network weights, D is the data vector, and M is the neural network model being trained. In our work, we adopt the Levenberg–Marquardt algorithm to train the Bayesian regularization neural network and we employ the Gauss–Newton algorithm to approximate the Hessian matrix $F(w)$ in order to find the optimal values of the parameters α and β.

2.7. Radial basis function neural network

As with the FFNNs and BRFFs, the RBFNN system consists of three layers, namely the input, hidden and output layer. The input layer distributes the input data among the hidden nodes that are fully connected to the previous layer. Each one of the input variables are assigned to a node in the input layer and pass directly to the hidden layer without weights. The hidden nodes contain the radial basis functions (RBFs) represented by Gaussian kernels and used as transfer functions to process information contained in the input layer nodes. Each neuron in the hidden layer estimates the local response to its input. Finally, the neuron in the output layer sums up the outputs of hidden neurons. Mathematically, the output of the jth unit $H_j(x)$, in the hidden layer for an input x_i is computed as follows:

$$H_j(x) = H_j\left[x_i - c_j\right] = exp\left(\frac{-\left(x_i - c_j\right)^2}{2\delta_j^2}\right) \tag{14}$$

whereby x_i is the input, c_j represents the position of the center of the jth Gaussian function, and δ the width parameter controlling the smoothness of the Gaussian function. The value of the parameter δ is set to unity in the current study. Finally, the output y of the system is calculated by the linear combination of the K radial basis functions plus the bias w_0 as follows:

$$y\left(x_j\right) = \sum_{j=1}^{K} w_j H_j\left[x_i - c_j\right] + w_0 \tag{15}$$

2.8. Bayesian optimization, performance and complexity measurement

We employ Bayesian optimization (BO) [26] to estimate the optimal values of the key parameters of the SVR (penalty coefficient C, regularization factor ε and slack variable ξ), the width of the kernel of the Gaussian regression process, and the optimal k and distance metric associated with the kNN algorithm. The BO has

Table 1
Nonlinear characteristics

Metric	Training sample	Testing sample
Hurst's exponent	1.5672	1.4746
Sample entropy	2.6325×10^{-04}	0.0015
L-Z complexity	5.9143	5.7021
Kolmogorov complexity	9.0181	8.6239

two major advantages. Firstly, it makes use of an acquisition function to consider both regions where the model calculates the objective function being low and regions where uncertainty is high. Secondly, it is fast to execute and provides good results in general avoiding overfitting.

Further, we assess the performance of each machine learning predictive system based on the root mean squared errors (RMSE), which is a standard metric adopted in signal analysis and prediction. Accordingly, the lower the RMSE performance metric is, the better the forecasting accuracy. Finally, four nonlinear statistics are used to characterize training and testing datasets with a goal to improve our understanding of the inherent performance of each machine learning model.

In order to account for the nonlinear dynamics of Bitcoin intraday price data, we attempt to quantify such variations by performing a general complexity analysis based on nonlinear measures, namely, the Hurst's exponent [27], sample entropy [28], Lempel–Ziv [29] and Kolmogorov complexity [30]. The advantage is that those metrics are appropriate to describe the behaviour of nonlinear and non-stationary temporal series with no prior assumptions about distributions and generating process. In this regard, the Hurst's exponent is used to evaluate any long memory traits so that hidden temporal structure can be better described. The method of detrended fluctuation analysis (DFA) is also adopted for estimation [27]. Entropy, estimates randomness and redundancy of information in the signal. To analyze randomness in Bitcoin intraday price data, sample entropy [28] is chosen as it provides high consistent estimates, exhibits strong robustness and produces good representation. The Lempel–Ziv complexity represents the rate of new, non-repetitive, pattern generation inside the series. Finally, the Kolmogorov complexity is based on the compressibility of a sequence to assess regularity in the signal. In short, we believe that by employing such various complexity measures we will provide complementary information useful to optimally divide the training and testing datasets. Overall, the application of nonlinear metrics would offer additional insight into the structure of Bitcoin complex data-generated topological manifolds.

3. Empirical and simulation results

We investigate the Bitcoin intraday price data at 5 minutes high-frequency sampling for the period from 1 January 2016 to 16 March 2018. Hence, the total sample size is 65,535 observations. For each machine learning model, the first 80% of the dataset obs is employed for training while the remaining 20% for testing. The training and testing samples are illustrated in Fig. 1. The measured nonlinear metrics of the training and testing samples are provided respectively in Table 1. Compared to training set, the testing sample shows lower level of persistence, higher score of sample entropy, lower level in the rate of pattern generation, thus overall a lower level of complexity. The training and testing samples exhibit distinctive nonlinear dynamical traits.

We forecast the next 5-minute step price level of Bitcoin, whilst the previous five observations are used as inputs, for each predictive system. Fig. 2 exhibits the graphical output from Bayesian optimization applied to attain the optimal width of the Gaussian kernel. The Bayesian optimization algorithm involved thirty iterations.

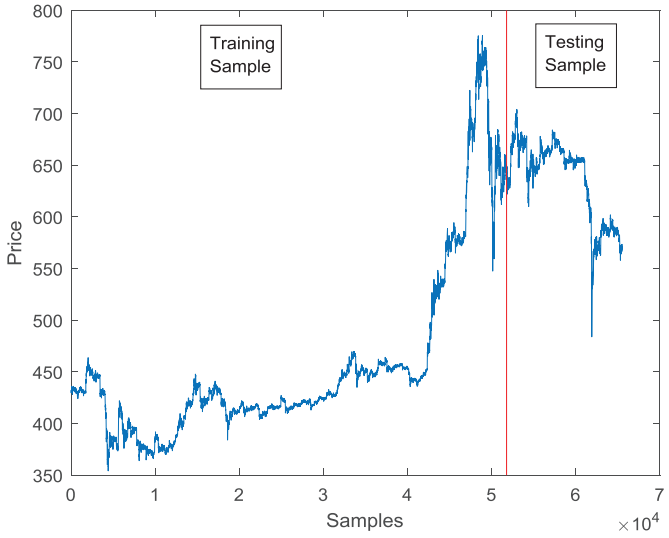


Fig. 1. The disjoint samples of the full one, i.e., the training sample (left of red vertical line) and the testing sample (right of vertical line). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

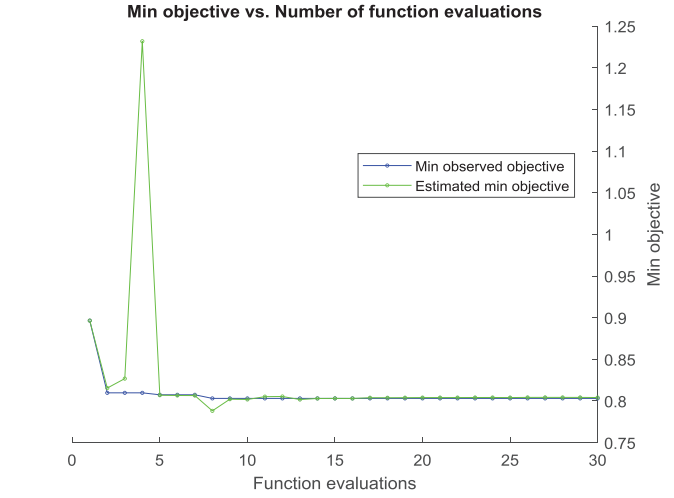
The minimum observed value of the objective function and the estimated minimum simultaneously converged at the ninth iteration as shown in the top panel of Fig. 2. Hence, the best feasible width of the Gaussian kernel is 1.6514 according to the bottom panel of Fig. 2.

In Figs. 3–9 we comparative plot the true versus the forecasted values of intraday Bitcoin prices. A visual inspection suggests that all machine learning systems achieved relatively accurate forecasts, considering that the curves of the true and forecasted values are very close to each other. Moreover, the calculated RMSE statistics and the associated values are shown in Fig. 10 for each predictive model. Accordingly, in terms of performance measurement the RMSE scores for the SVR, GRP, RT, *k*NN, BPNN, BRNN, and RBFNN yielded 1.1139, 0.8674, 1.2231, 2.1111, 0.8475, 0.8443, and 1.3165 respectively. It appears that the best predictive model for Bitcoin high-frequency price data is the BRNN followed by BPNN, GRP, SVR, RT, RBFNN, and *k*NN. Aside from RBFNN, we observe that artificial neural networks outperform all other predictive topologies. In this regard, we also confirm that the performance of BRNN and FFNN is very similar, as they achieved 0.8475 and 0.8443 RMSE scores respectively.

4. Discussion and conclusions

Intelligent machine learning systems present several advantages in modeling and forecasting big data sets such as Bitcoin high frequency price time series. Indeed, they are able to accommodate multiple interactions among predictive inputs, run under no assumptions about the form of the relationship between inputs and outputs, while they are able to identify nonlinear chaotic patterns in temporal signal oscillations.

Ou novel contributions are three-fold; first and foremost we comparatively explored various AI systems to address the problem of accurate predictability in Bitcoin markets. This empirical investigation has not been conducted in the literature to the best of our knowledge. The set of models included support vector and Gaussian Poisson regressions, regression trees and the *k*-nearest neighbors algorithms, and neural network topologies such as feedforward, Bayesian regularization and radial basis function NNs. Secondly, we implemented the seven different systems on ultra high frequency data in cryptocurrency markets. Lastly, we offered a



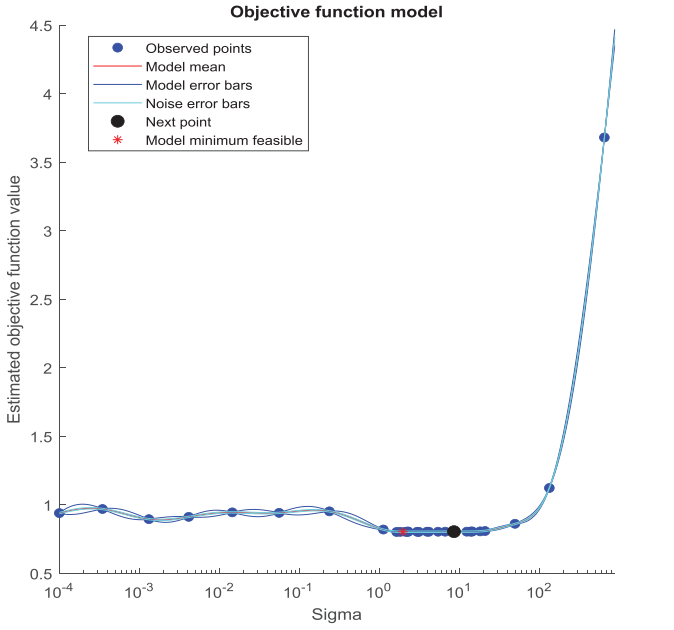


Fig. 2. Graphical results from Bayesian optimization when applied to the Gaussian regression process. We obtain the optimal width of the Gaussian kernel after 30 iterations. The best observed feasible point for Gaussian kernel width is 1.6514, according to the bottom panel (red *). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

broad and in-depth examination of the inherent complexity that drives the underlying Bitcoin high frequency price time series and we revealed their chaotic structure.

We found that both training and testing samples exhibit strong nonlinear dynamical patterns. According to our results, all three types of machine learning predictive systems provided accurate forecasts. They were very effective in modeling noisy high frequency, nonlinear and complex time series. Given the small feature space i.e., comprising five inputs to predict the one-step-ahead price of Bitcoin, and given the brute force performance of the predictive systems under study, it is clear that the inclusion of additional features would not provide a significant predictive advantage. Indeed, only five lags were found effective in learning and forecasting Bitcoin intraday prices.

Model ranking rendered the Bayesian regularization neural network as the best followed by the feedforward multilayer net-

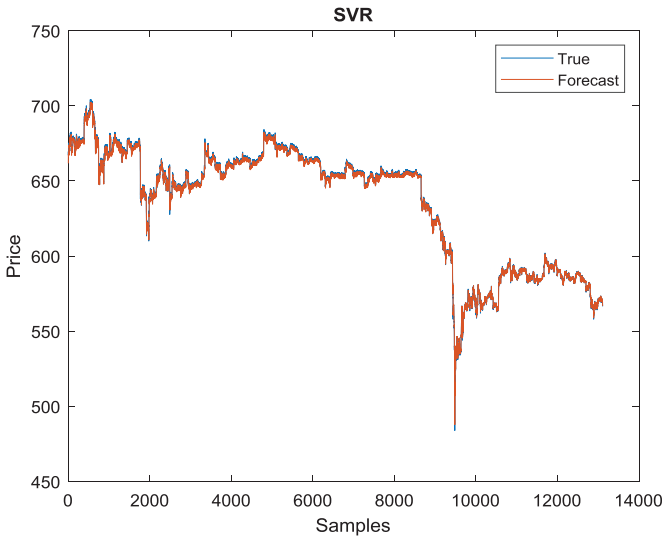


Fig. 3. Plot of true versus forecasted values generated by support vector regression. Bayesian optimization is used to fine-tune key parameters.

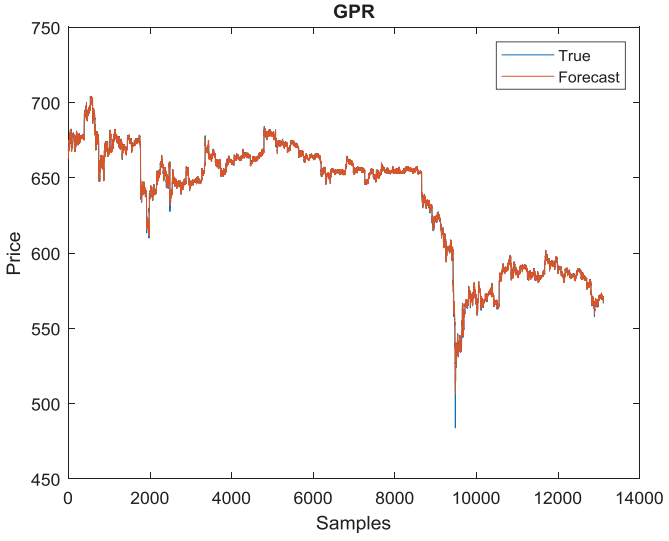


Fig. 4. Plot of true versus forecasted values generated under Gaussian process regression. Bayesian optimization is utilized fine-tune key parameters.

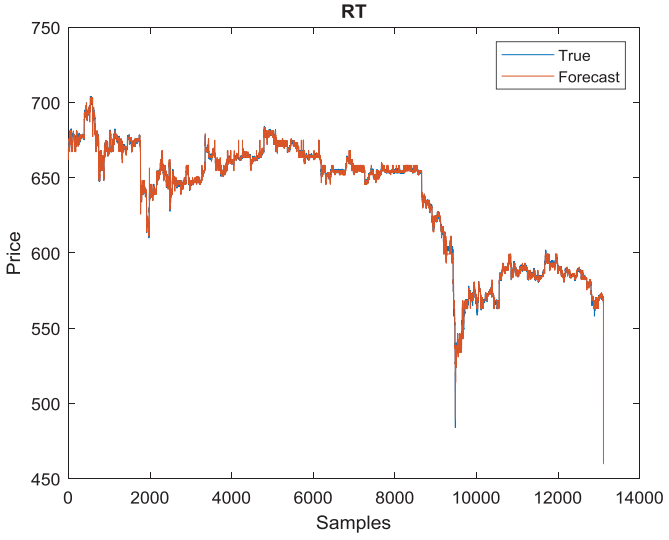


Fig. 5. Plot of true versus forecasted values produced by regression trees.

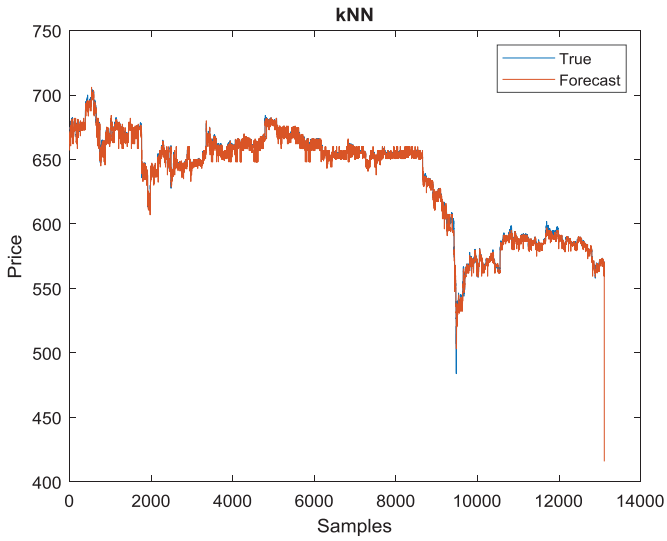


Fig. 6. Plot of true versus forecasted values with the use of *k*NN algorithm. Bayesian optimization is used to fine-tune its key parameters.

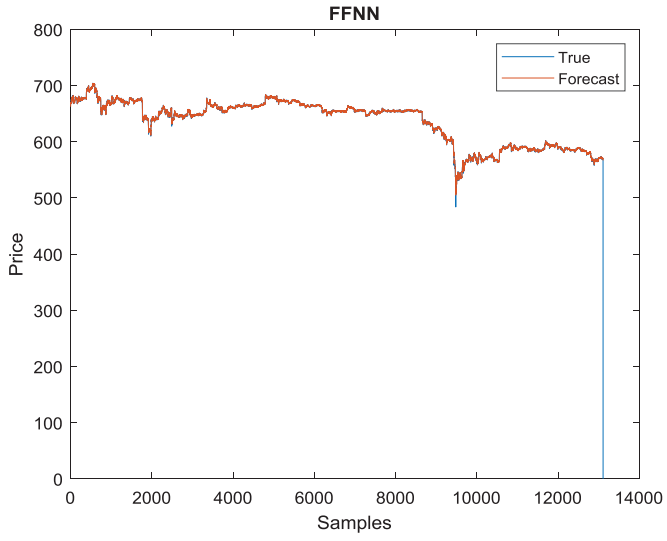


Fig. 7. Plot of the true versus forecasted values using a feedforward neural network.

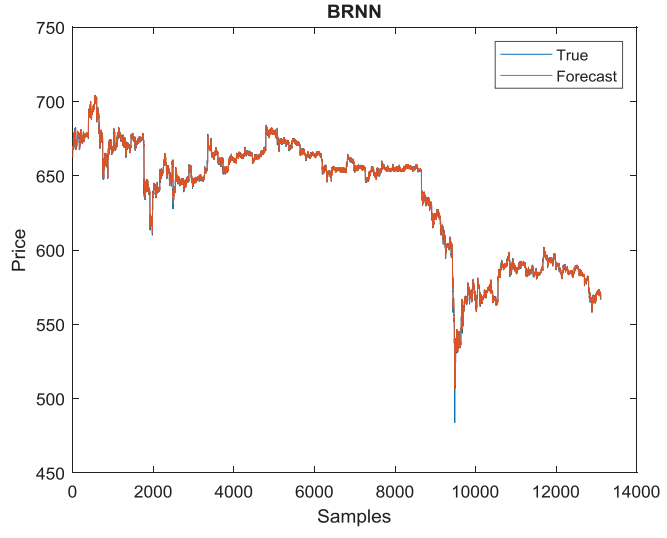


Fig. 8. Plot of the true versus forecasted values using a Bayesian regularization neural network.

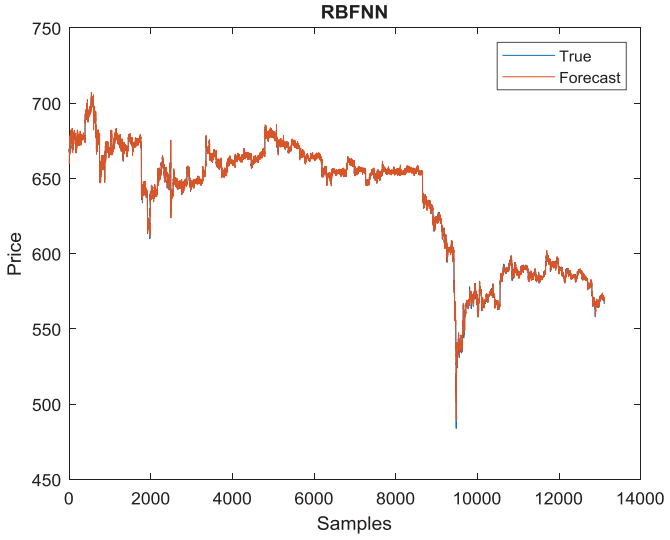


Fig. 9. Plot of the true versus forecasted values via the use of radial basis function neural networks.

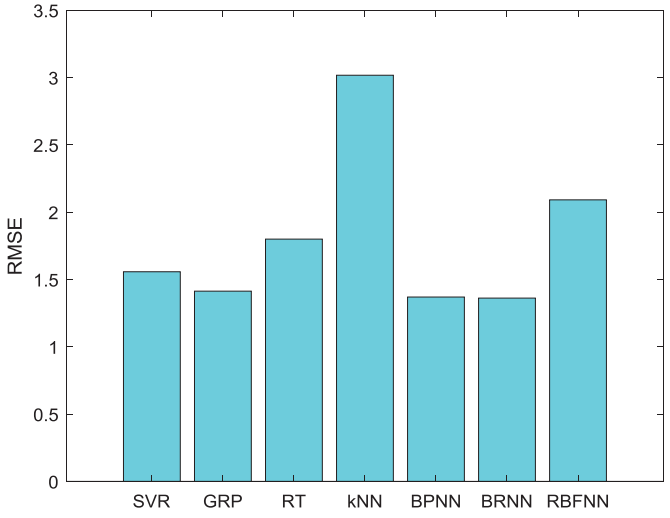


Fig. 10. Performance of each machine learning model in terms of RMSE. The comparative evaluation concerns the models of support vector regression (SVR), Gaussian Poisson regression (GRP), regression tree (RT), the *k*-nearest neighbors (*k*NN) algorithm, feedforward neural network (FFNN), Bayesian regularization neural network (BRNN), and radial basis function neural network (RBFNN). The SVR, GRP, RT, *k*NN, BPNN, BRNN, and RBFNN topologies yielded respectively 1.1139, 0.8674, 1.2231, 2.1111, 0.8475, 0.8443, and 1.3165 scores. Overall, the BRNN achieved the lowest forecasting error, whilst the *k*NN algorithm the highest error.

work, the Gaussian Poisson regression, the support vector regression model, regression trees, radial basis function neural networks, and *k*-nearest neighbours algorithm. It is worth mentioning that the support vector regression, the Gaussian regression and the *k*-nearest neighbours algorithm required a higher execution time (several minutes) compared to the other models, due to parameter optimization via Bayesian techniques. The overall superiority of the artificial neural networks can be explained by the fact that such intelligent models are computer-based and involve several parallel information-processing elements to account for nonlinear relationships between inputs and outputs, even in case of highly nonstationary and noisy data. Interestingly, they were remarkably fast even for big data sizes, as their convergence was achieved in around seven seconds on average. We noticed however, a relative underperformance of the radial basis function neural networks against the support vector and Gaussian Poisson regression models

as well as versus regression trees, which could be attributed to the fact that the width of their kernel was not optimized, whilst support vector and Gaussian kernels were optimized by Bayesian optimization. The regression tree is a nonlinear-based algorithm with no kernel involved in the training phase, hence it converged in few seconds. One drawback of the Gaussian regression, which belongs to statistical machine learning techniques, is that it naively assumes normality, stationarity and sample independency.

The effectiveness of the Bayesian regularization neural network system is overwhelming. As opposed to feedforward and radial basis functions, the Bayesian one involved an additional term and an objective function to penalize large weights in order to obtain smoother mapping of the input-output plane. In fact, the Bayesian regularization utilized probability distribution modeling to find the optimal hyper-parameters. Hence, it proved to be more accurate and faster compared to all the other machine learning approaches considered in our work. The Bayesian regularization neural network delivered an outstanding accuracy in forecasting intraday Bitcoin prices, while at the same time its convergence performance was unhindered and direct. Our future follow-up studies include the combination of AI and deep learning to further enhance accuracy in cryptocurrency predictability.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] Available online: https://coinmarketcap.com/ (accessed on 04 December 2019).
[2] Stosic D, Stosic D, Ludermir TB, Stosic T. Exploring disorder and complexity in the cryptocurrency space. Physica A 2019;525:548–56.
[3] Song JY, Chang W, Song JW. Cluster analysis on the structure of the cryptocurrency market via Bitcoin-Ethereum filtering. Physica A 2019;527 Article 121339.
[4] Lahmiri S, Bekiros S, Salvi A, memory Long-range. distributional variation and randomness of bitcoin volatility. Chaos Solitons Fractals 2018;107:43–8.
[5] Pele DT, Mazurencu-Marinescu-Pele M. Using high-frequency entropy to forecast bitcoin's daily value at risk. Entropy 2019(21). doi:10.3390/e21020102.
[6] Cheng Q, Liu X, Zhu X. Cryptocurrency momentum effect: DFA and MF-DFA analysis. Physica A 2019;526 Article 120847.
[7] Stavroyiannis S, Babalos V, Bekiros S, Lahmiri S, Uddin GS. The high frequency multifractal properties of Bitcoin. Physica A 2019;520:62–71.
[8] Lahmiri S, Bekiros S. Big data analytics using multi-fractal wavelet leaders in high-frequency Bitcoin markets. Chaos Solitons Fractals 2019:109472.
[9] Zhang X, Yang L, Zhu Y. Analysis of multifractal characterization of Bitcoin market based on multifractal detrended fluctuation analysis. Physica A 2019;523:973–83.
[10] Borri N. Conditional tail-risk in cryptocurrency markets. J Empir Finance 2019;50:1–19.
[11] Peng Y, Melo Albuquerque PH, Camboim de Sá JM, Padula AJA, Montenegro MR. The best of two worlds: forecasting high frequency volatility for cryptocurrencies and traditional currencies with support vector regression. Expert Syst Appl 2018;97:177–92.
[12] Lahmiri S, Bekiros S. Cryptocurrency forecasting with deep learning chaotic neural networks. Chaos Solitons Fractals 2019;118:35–40.
[13] McNally S, Roche J, Caton S. Predicting the price of Bitcoin using machine learning. IEEE 26th euromicro international conference on parallel, distributed and network-based processing; 2018. doi:10.1109/PDP2018.2018.00060.
[14] Sutiksno DU, Ahmar AS, Kurniasih N, Susanto E, Leiwakabessy A. Forecasting historical data of Bitcoin using ARIMA and α-Sutte indicator. IOP Conf Ser J Phys 2018;1028:012194.
[15] Catania L, Grassi S. Francesco Ravazzolo, Forecasting cryptocurrencies under model and parameter instability. Int J Forecast 2019;35:485–501.
[16] Akcora CG, Dey AK, Gel YR, Kantarcioglu M. Forecasting Bitcoin price with graph chainlets. Advances in knowledge discovery and data mining lecture notes in computer science , 10939. Springer, Cham; 2018.
[17] Atsalakis GS, Atsalaki IG, Pasiouras F, Zopounidis C. Bitcoin price forecasting with neuro-fuzzy techniques. Eur J Oper Res 2019;276:770–80.
[18] Xiaolei S, Mingxi L, Zeqian S. A novel cryptocurrency price trend forecasting model based on LightGBM. Finance Res Lett 2018. doi:10.1016/j.frl.2018.12.032.
[19] Vapnik V, Golowich S, Smola A. Support vector machine for function approximation, regression estimation, and signal processing. Adv Neural Inf Process Syst 1996;9:281–7.
[20] Rasmussen CE, Williams CK. Gaussian processes for machine learning. Massachusetts Institute of Technology: MIT-Press; 2006.
[21] Morgan JN, Sonquist JA. Problems in the analysis of survey data, and a proposal. J Am Stat Assoc 1963;58:415–34.
[22] Cover TM, Hart PE. Nearest neighbor pattern classification. IEEE Trans Inf Theory 1967;13:21–7.
[23] Haykin SO. Neural networks and learning machines. 3rd ed. Pearson; 2008.
[24] Burden F, Winkler D. Bayesian regularization of neural networks. Methods Mol Boil 2008;458:25e.
[25] Cybenko G. Approximations by superpositions of a sigmoidal function. Math Control, Signals Syst 1989;2:303–14.
[26] M. Gelbart, J. Snoek, R.P. Adams, Bayesian optimization with unknown constraints, (2014). https://arxiv.org/abs/1403.5607
[27] Peng C-K, Buldyrev SV, Havlin S, Simons HE Stanley M, Goldberger AL. Mosaic organization of DNA nucleotides. Phys Rev E 1994;49:1685–9.
[28] Richman JS, Moorman JR. Physiological time-series analysis using approximate entropy and sample entropy. Am J Physiol Heart Circulat Physiol 2000;278:H2039–49.
[29] Lempel A, Ziv J. On the complexity of finite sequences. IEEE Trans Inf Theory 1976;22:75–81.
[30] Kolmogorov AN. Combinatorial foundations of information theory and the calculus of probabilities. Russian Math Surv 1983;38:29–40.