# A Time Series Approach to Explainability for Neural Nets with Applications to Risk-Management

Marc Wildi*and Branka Hadji Misheva†

May 4, 2023

**Abstract**

Artificial intelligence (AI) is creating one of the biggest revolution across technology-driven application fields. For the finance sector, it offers many opportunities for significant market innovation and yet broad adoption of AI systems heavily relies on our trust in their outputs. Trust in technology is enabled by understanding the rationale behind the predictions made. To this end, the concept of eXplainable AI (XAI) emerged introducing a suite of techniques attempting to explain to users how complex models arrived at a certain decision. For cross-sectional data, classic XAI approaches provide insights about the models' inner workings, but some of these techniques can conflict with the dependence structure of longitudinal data (time series). We here propose a novel non-invasive XAI-technique for deep learning methods (DL) which preserves data-integrity as well as the natural time ordering and dependence structure of the data. In addition to its original explainability intent, a real-time monitoring of the XAI-tool extends the scope of the method to risk-management applications.

## 1 Introduction

Deep learning (DL) has become highly popularized in many aspects of data science and has increasingly gained traction in the field of financial and economic time series forecasting ([1], [2], [3], [4]) with application of DL methods to stock and forex market which significantly outperform traditional counterparts ([5],[6], [7], [8]). This trend was also confirmed in more recent M4 and M5 installments of the Makridakis Forecasting Competitions , where a hybrid Exponential Smoothing Recurrent Neural Network method and LightGBM, won the competitions, respectively ([9];[10]).
The introduction of DL methods for financial time series forecasting potentially enables higher predictive accuracy but this comes at the cost of higher complexity and thus lower interpretability. DL-methods are often referred to as 'black-boxes' because it is difficult to understand how variables are jointly related to arrive at a certain output. This reduced ability to understand the inner workings and mechanisms of DL models unavoidably affects their trustworthiness and the willingness among practitioners to deploy such methods in sensitive domains such as finance. As a result, the scientific interest in the field of eXplainable artificial intelligence (XAI) has grown tremendously within the last few years ([11], [12], [13]), giving rise to a taxonomy of methods, see e.g. [14], [15], [16].
Many of the 'classic' XAI-techniques were developed for cross-sectional data, assuming independently and identically distributed observations, so that extensions to autocorrelated time series can pose problems, in particular when resampling or reshuffling data. Recent XAI-methods for time series proposed in [17], [18], [19] or [20], generally assume data- or model-structures which are not ideally suited for the application to non-stationary time series or financial data with their somehow notorious stylized facts (trends, conditional heteroscedasticity, kurtosis). Alternative approaches, such as reviewed in [21], emphasize particular aspects of explainability which are to some extent

*ZHAW Zurich University of Applied Sciences, IDP; e-mail: wlmr@zhaw.ch
†ZHAW Zurich University of Applied Sciences, IDP; email: hadj@zhaw.ch

too specific and specialized or too complex for the purpose at hand. Instead, we here propose a time series approach with the purpose of monitoring the multivariate dependence structure of the data, as captured by the 'black-box' model. Given recent successes in forecast competitions, we emphasize neural nets and we propose to address explainability in terms of partial derivatives of the net-output with respect to the explanatory variables. This approach is non-invasive in the sense that it relies on infinitesimal changes of the explanatory variables so that data-integrity, information-content and the natural time-ordering are preserved. Moreover, the approach is obtained in closed-form, it is exact and computationally effective, requiring a single pass through the net only. By extension, our tool can be used to monitor the dependence structure of the data so that novel risk-management strategies can be derived by emphasizing departures from linearity as an indicator for increased market-complexity.

## 2 The Utility of Classical Approaches for Applications Featuring Financial Time Series

Key limitation of many classical methods is the fact that they ignore feature dependence which is a defining property of financial data. Specifically, the procedures of perturbation-based methods like the PDP, PFI, SHAP, LIME, etc., start with producing artificial data points, obtained either through replacement with permuted or randomly select values from the background data; or through the generation of new "fake" data (simulation), that are consequently used for model predictions. Such steps result in several concerns:

- if features are correlated, the artificial coalitions created will lie outside of the multivariate joint distribution of the data,
- if the data are independent, coalitions can still be meaningless; perturbation-based methods are fully dependent on the ability to perturb samples in a meaningful way which is not always the case with financial data (ex. one-hot encoding)
- generating artificial data points through random replacement disregards the time sequence hence producing unrealistic values for the feature of interest.

In any case and notwithstanding the possibility of mixing unrelated trend-levels and volatility-clusters, the eventuality of re-combining artificially data from remote past and current time seems counter-intuitive not least from a purely application-based 'meta-explainability' perspective. Yet another concern with classical approaches is that they can lead to misleading results due to feature interaction. Namely, as demonstrated by ([22]) both PDP and ALE plots can lead to misleading conclusions in situations in which features interact.

Looking specifically at the SHAP framework, both conditional and marginal distributions can be used to sample the absent features and both approaches have their own issues. For example, the TreeSHAP is a conditional method and under conditional expectations a feature that has no influence on the prediction function (but is correlated with another feature that does) can get a TreeSHAP estimate different from zero ([23]) and can affect the importance of the other features, see [24].

On the other hand, sampling from the marginal distribution, instead of the conditional, would ignore the dependence structure between present and absent features.Further discussion and examples on the mathematical issues that arise from the estimation procedures used when applying Shapley values as feature importance measures can be found in [25].

## 3 XAI for Neural Nets: a Time Series Approach

Until recently, computationally intensive methods had a hard time competing against classic (linear) time series approaches see [26] for a review. However, the recent M4 and M5 competitions brought forward hybrid approaches mixing neural nets and classic exponential smoothing, see [27].

These results push forward the need to address the 'black-box' problem in a way that is compliant with the dependence structure of longitudinal data.

## 3.1 Introduction

Let $y_t, x_{it}$, $i = 1, ..., n$, $t = 1, ..., T$ denote a set of dependent (target) and explanatory series and consider a model

$$y_t = o(x_{1t}, ..., x_{nt}) + \epsilon_t$$

where $o_t := o(x_{1t}, ...x_{nt})$ is a function of the data. For a linear model $o_t = b + \sum_{i=1}^{n} w_i x_{it}$ the weights $w_i$ are straightforwardly interpretable in terms of partial derivatives of $y_t$ with respect to $x_{it}$ (or partial correlations). The intrinsic interpretability of partial derivatives carries over to non-linear models and in particular to neural nets. Specifically, we here analyze partial derivatives $w_{it} := \partial o_t / \partial x_{it}$, $i = 1, ..., n$, $t = 1, ..., T$, of a a neural net with output $o_t$ (generalizations to multiple output neurons are straightforward). For simplicity of exposition and for reasons related to the nature of financial time series, we assume a feedforward net-architecture for fitting the data. Indeed, the Efficient Market Hypothesis (EMH) or, more generally, the so-called typical spectral shape of economic time series in [28], suggest evidence of a short memory, defeating the purpose of recurrent nets. While we here allow for departures from a strict acception of the EMH, by admitting lagged data in the input layer, our intention is also to control and to limit data-dependence explicitly, countering once again the purpose of recurrent nets. In any case, classic unfolding of recurrent into feedforward structures would allow for a formal extension of our results to recurrent nets. Concerning the 'black-box' problem of neural nets, the sequences $w_{it}$ of partial derivatives offer a natural explanation in terms of time-dependent weights, slopes or sensitivities and their natural ordering, along the time axis, allows for a monitoring of the net's reactiveness to input data. By extension, we can interpret $w_{it}, i = 1, ..., n$, $t = 1, ..., T$ as a representation of the momentary multivariate dependence structure of the data, as sensed by the net, and changes of $w_{it}$ are potentially indicative of non-linearity or of non-stationarity of the underlying data generating process. In order to complete our XAI-tool, we also propose a new synthetic intercept defined as

$$b_t := o_t - \sum_{i=1}^{n} w_{it} x_{it} \tag{1}$$

The resulting sequence $\mathbf{LPD}_t := (b_t, w_{1t}, ..., w_{nt})'$ is referred to as Linear Parameter Data: the matrix LPD obtained by stacking the $(n+1)$-dimensional row-vectors $\mathbf{LPD}_t$ has dimension $T * (n+1)$, irrespective of the complexity of the neural net, and it can be interpreted in terms of time-dependent point-wise exact linear replication of the net, obtained in closed-form (see below), without resampling or reshuffling.

## 3.2 Derivation of the LPD

We assume a feedforward net with $p-1$ hidden layers of dimension $n_k$, $k = 1, ..., p-1$ each; for ease of exposition we also assume that all neurons are equipped with a sigmoid activation function, acknowledging that straightforward modifications would apply in the case of alternative (piecewise differentiable) specifications. Let $\mathbf{A}_t^{(k)}$ denote a column-vector of dimension $n_k$ corresponding to the vector of outputs of the $k$-th layer at time $t$ and let $\mathbf{W}^k$ designate the matrix of dimension $(n_{k-1}, n_k)$ of weights linking the neurons of layer $k-1$ to the neurons of layer $k$ in the fully-connected net, whereby $n_0 = n$ is the dimension of the input-layer: if the net is not fully connected then silent connections receive value zero in $\mathbf{W}^k$. The outputs at layers $k$ are

$$\mathbf{A}_t^{(1)} = \sigma\left(\mathbf{W}^{1\,\prime} \mathbf{x}_t\right) \tag{2}$$

$$\mathbf{A}_t^{(k)} = \sigma\left(\mathbf{W}^{k\,\prime} \mathbf{A}_t^{(k-1)}\right) \tag{3}$$

where the prime-superscript of $\mathbf{W}^{k\,\prime}$ refers to ordinary matrix transposition, $\mathbf{x}_t$ is the $n$-dimensional vector of input-data and where $\sigma()$ symbolizes the sigmoid activation function. Denote further by

$\mathbf{dA}^{\mathbf{f}}{}_t^{(k)}$ the $(n, n_k)$-dimensional matrix of partial derivatives of the vector $\mathbf{A}_t^{(k)}$ with respect to the explanatory variables $x_{it}, i = 1, ..., n$: the LPD (without the intercept) is identified with $\mathbf{dA}^{\mathbf{f}}{}_t^{(p)}$ computed at the output layer. The superscript $\mathbf{f}$ in $\mathbf{dA}^{\mathbf{f}}{}_t^{(k)}$ refers to the forward direction, computing derivatives from left (input-layer) to right (output-layer). Below, we will introduce a backward derivative, moving from right to left along the chain-rule of differentiation: both expressions will be useful when deriving higher order partial derivatives in section 3.4. Note that $\mathbf{dA}^{\mathbf{f}}{}_t^{(p)}$ differs from the classic gradient of the mean-square forecast error with respect to net-parameters (biases and weights), as used by 'backpropagation' for parameter-fitting. For the first hidden layer, $k = 1$, we then obtain from 2

$$\mathbf{dA}^{\mathbf{f}}{}_t^{(1)} = \left(\mathbf{W}^{(1)\prime} \cdot \mathbf{A}_t^{(1)} \cdot (\mathbf{e}^{(1)} - \mathbf{A}_t^{(1)})\right)' \tag{4}$$

where $\mathbf{e}^{(1)}$ is a column-vector of ones of dimension $n_1$, matching $\mathbf{A}_t^{(1)}$, where the dot-symbol indicates row-wise multiplication (the $i-th$ row of a matrix is multiplied with the $i$-th element of a vector) and where $\mathbf{A}_t^{(1)} \cdot (\mathbf{e}^{(1)} - \mathbf{A}_t^{(1)})$ is obtained from the derivative $\dot{\sigma}(\cdot) = \sigma(\cdot)(1 - \sigma(\cdot))$ of the sigmoid activation function (straightforward modifications apply in the case of alternative differentiable activations). Hence, $\mathbf{dA}^{\mathbf{f}}{}_t^{(1)}$ is a matrix of dimension $(n, n_1)$. Having all necessary algebraic elements in place, we can now proceed iteratively for $k = 2, ..., p$, through all layers, obtaining the derivative of 3 as

$$\mathbf{dA}^{\mathbf{f}}{}_t^{(k)} = \left(\left(\mathbf{W}^{(k)\prime}\mathbf{dA}^{\mathbf{f}}{}_t^{(k-1)\prime}\right) \cdot \mathbf{A}_t^{(k)} \cdot (\mathbf{e}^{(k)} - \mathbf{A}_t^{(k)})\right)' \tag{5}$$

where $\mathbf{W}^{(k)\prime}\mathbf{dA}^{\mathbf{f}}{}_t^{(k-1)\prime}$ is the ordinary matrix-product of the $(n_k, n_{k-1})$-dim $\mathbf{W}^{(k)\prime}$ and the $(n_{k-1}, n)$-dim $\mathbf{dA}^{\mathbf{f}}{}_t^{(k-1)\prime}$, obtaining a $(n, n_k)$-dim matrix $\mathbf{dA}^{\mathbf{f}}{}_t^{(k)}$, after transposition. Finally, $\mathbf{LPD}_t$ is obtained by completing $\mathbf{dA}^{\mathbf{f}}{}_t^{(p)}$ with the intercept 1.
We now proceed to a backward computation of the LPD which will be used in section 3.4: for consistency with later expressions we here derive the LPD for a single output neuron $o_{jt}$ (extensions are straightforward). Starting at the output layer

$$dA^{b}{}_{jt}^{(p)} = A_{jt}^{(p)}(1 - A_{jt}^{(p)}) \tag{6}$$

where the superscript 'b' of $dA^{b}{}_{jt}^{(p)}$ refers to the backward direction. Then, for the last hidden layer $k = p - 1$ we obtain

$$\mathbf{dA}^{\mathbf{b}}{}_{jt}^{(p-1)} = \left(\mathbf{W}_j^{(p)} dA^{b}{}_{jt}^{(p)}\right) \cdot \mathbf{A}_t^{(p-1)} \cdot (\mathbf{e}^{(p-1)} - \mathbf{A}_t^{(p-1)}) \tag{7}$$

Here, $\mathbf{W}_j^{(p)}$ designates the $j$-th column vector of $\mathbf{W}^{(p)}$, linking the neurons in the last hidden layer $p - 1$ to the $j$-th output neuron. $\mathbf{dA}^{\mathbf{b}}{}_{jt}^{(p-1)}$ inherits the dimension $n_{p-1}$ directly from $\mathbf{W}_j^{(p)}$. If $p > 2$ then

$$\mathbf{dA}^{\mathbf{b}}{}_{jt}^{(k)} = \left(\mathbf{W}^{(k+1)}\mathbf{dA}^{\mathbf{b}}{}_{jt}^{(k+1)}\right) \cdot \mathbf{A}_t^{(k)} \cdot (\mathbf{e}^{(k)} - \mathbf{A}_t^{(k)}) \tag{8}$$

for $k = p - 2, p - 3, ..., 1$. The $i$-th row of $\mathbf{W}^{(k+1)}$ feeds the derivatives $\mathbf{dA}^{\mathbf{b}}{}_{jt}^{(k+1)}$ of the next layer $k + 1$ 'backward' to the $i$-th neuron in layer $k$ or, equivalently, the $m$-th column of $\mathbf{W}^{(k+1)}$ pushes the derivatives of layer $k$ 'forward' to the $m$-th neuron of the next layer $k + 1$. Digging out dimensions, $\mathbf{W}^{(k+1)}$ is $(n_k, n_{k+1})$, the column-vector $\mathbf{dA}^{\mathbf{b}}{}_{jt}^{(k+1)}$ is $n_{k+1}$ and therefore $\mathbf{dA}^{\mathbf{b}}{}_{jt}^{(k)}$ is $n_k$. Finally, for the input layer $k = 0$ we obtain

$$\mathbf{dA}^{\mathbf{b}}{}_{jt}^{(0)} = \mathbf{W}^{(1)}\mathbf{dA}^{\mathbf{b}}{}_{jt}^{(1)} \tag{9}$$

where $\mathbf{A}_t^{(0)} \cdot (\mathbf{e}^{(0)} - \mathbf{A}_t^{(0)})$ is replaced by the identity since input neurons are exempt of activation functions. An equivalent expression for $\mathbf{LPD}_t$ is obtained by completing $\mathbf{dA^b}_{jt}^{(0)}$ with the intercept 1. In the case of multiple output neurons, $n_p > 1$, we can append the column-vectors $\mathbf{dA^b}_{jt}^{(k)}$ of dimension $n_k$ into matrices $\mathbf{dA^b}_{t}^{(k)}$ of dimension $(n_k, n_p)$ and $\mathbf{dA^b}_{t}^{(0)}$ is the Jacobian-matrix of dimension $(n, n_p)$ collecting all partial derivatives of all output neurons at once.

### 3.3 Alternative Explanation Purposes: X-Functions

The LPD is based on partial derivatives of $\mathbf{o}_t = \mathbf{I}(\mathbf{o}_t)$ where $\mathbf{I}$ is the identity. However, alternative functions of the net output might draw attention such as for example the aggregate mean-square error $\sum_{t=1}^{T}(\mathbf{y}_t - \mathbf{o}_t)'(\mathbf{y}_t - \mathbf{o}_t)$: partial derivatives of the latter can help identify high-impact data potentially affecting the fitted net-parameters (robustness). More generally, a so-called 'explainability' or X-function $xf(\mathbf{o}_t)$ of the net-output can highlight specific research purposes, like for example trading performances, and a monitoring of partial derivatives of $xf(\mathbf{o}_t)$ informs about potential data-impact on the performance measure, either in real-time, at $t = T$, or historically, for $t \leq T$. For (piece-wise) differentiable $xf(\cdot)$, the compound sensitivities can be obtained by augmenting equation 5 with the $n_p$-dimensional gradient $\nabla\mathbf{xf}(\mathbf{A}_t^{(p)})$ of $xf(\cdot)$ i.e.

$$\mathbf{d}(\mathbf{xf} \circ \mathbf{A^f})_t^{(p)} := \left(\nabla\mathbf{xf}(\mathbf{A}_t^{(p)}) \cdot \mathbf{dA^f}_t^{(p)\ \prime}\right)' \tag{10}$$

where the $(n, n_p)$-dimensional $\mathbf{d}(\mathbf{xf} \circ \mathbf{A^f})_t^{(p)}$ denotes the Jacobian-matrix of the sought-after sensitivities and where the $n_p$-dimensional (column-) vector $\nabla\mathbf{xf}(\mathbf{A}_t^{(p)})$ collects the derivatives $\dot{xf}(o_{jt})$, $j = 1, ..., n_p$, of $xf(\cdot)$ at the output layer. An equivalent extension applies to backward-computations initialized by 6

$$\left(\nabla\mathbf{xf}(\mathbf{A}_t^{(p)}) \cdot \mathbf{dA^b}_t^{(p)}\right) \tag{11}$$

where $\mathbf{dA^b}_t^{(p)} = (dA^{b(p)}_{1t}, ..., dA^{b(p)}_{n_p t})'$. In contrast to 10, which is also the final LPD expression, 11 must be backpropagated through all layers to obtain the corresponding compound LPD at the input layer, denoted by $\mathbf{d}(\mathbf{xf} \circ \mathbf{A^b})_t^{(0)}$. We next derive a closed-form solution for higher-order partial derivatives of $\mathbf{o}_t$.

### 3.4 Derivation of Quadratic Parameter Data (QPD) and of Twice-Differentiable X-Functions

Second order partial derivatives $\frac{\partial^2 o_{jt}}{\partial x_{it} \partial x_{kt}}$, referred to as Quadratic Parameter Data or QPD for short, are a measure of momentary change of the LPD, as a function of the data and at each time point $t$, and can be interpreted as a measure of non-linearity of the net and, by extension, of the underlying data generating process. For each output neuron $o_{jt}, j = 1, ..., n_p$ the QPD is a three-dimensional array of dimension $T * n * n$ but we are often mainly interested in the diagonal elements $\frac{\partial^2 o_{jt}}{\partial^2 x_{it}}, i = 1, ..., n$ only. In order to avoid cumbersome indexing we now derive the QPD for each output neuron $o_{jt}$ separately: formal expressions can be obtained by differentiating the backward-equations 6 to 9, breaking-up the chain-rule of differentiation into forward branch, for the inner functions, and backward branch, for the outer function. Specifically, starting at the output layer and taking the derivative of 6 with respect to $o_{jt}$ we obtain

$$dda_{jt}^{(p)} = A_{jt}^{(p)}(1 - A_{jt}^{(p)})(1 - 2A_{jt}^{(p)}) \tag{12}$$

$$\mathbf{ddA}_{jt}^{(p)} = dda_{jt}^{(p)} \mathbf{dA^f}_t^{(p-1)} \mathbf{W}_j^{(p)} \tag{13}$$

where 12 corresponds to the second order derivative of the sigmoid $\ddot{\sigma}() = \sigma()(1 - \sigma())(1 - 2\sigma())$, $\mathbf{dA^f}_t^{(p-1)}$ in 13 is the forward-derivative obtained in 5 and $\mathbf{W}_j^{(p)}$ is the $j$-th column of $\mathbf{W}^{(p)}$, of

dimension $n_{p-1}$, linking the neurons in layer $p-1$ to the $j$-th output neuron. Note that $\mathbf{dA}_t^{\mathbf{f}(p-1)}$ is a $(n, n_{p-1})$ matrix so that $\mathbf{dA}_t^{\mathbf{f}(p-1)}\mathbf{W}_j^{(p)}$ and hence $\mathbf{ddA}_{jt}^{(p)}$ are $n$-dim column vectors. As claimed, $dda_{jt}^{(p)}$ is the derivative of the outer function and the forward-term $\mathbf{dA}_t^{\mathbf{f}(p-1)}\mathbf{W}_j^{(p)}$ is the derivative of the inner function(s). We can now 'backpropagate' from to the $j$-th output neuron $o_{jt}$ backwards, to the last hidden layer $k = p-1$:

$$\begin{aligned}
\mathbf{dda}_t^{(p-1)} &= \mathbf{A}_t^{(p-1)} \cdot (\mathbf{e}^{(p-1)} - \mathbf{A}_t^{(p-1)}) \cdot (\mathbf{e}^{(p-1)} - 2\mathbf{A}_t^{(p-1)}) \\
\mathbf{ddA}_{jt}^{(p-1)} &= \left( \left( \mathbf{W}_j^{(p)} \mathbf{ddA}_{jt}^{(p)\prime} \right) \cdot \mathbf{A}_t^{(p-1)} \cdot (\mathbf{e}^{(p-1)} - \mathbf{A}_t^{(p-1)}) \right)' \qquad (14) \\
&\quad + \left( \mathbf{W}_j^{(p)} dA^{b(p)}_{jt} \cdot \mathbf{dda}_t^{(p-1)} \cdot \left( \mathbf{W}^{(p-1)\prime} \mathbf{dA}_t^{\mathbf{f}(p-2)\prime} \right) \right)' \qquad (15)
\end{aligned}$$

To see this, note that 7 can be split into the product of $\mathbf{Z}_{1t} := \mathbf{W}_j^{(p)} dA^{b(p)}_{jt}$ and $\mathbf{Z}_{2t} := \mathbf{A}_t^{(p-1)} \cdot (\mathbf{e}^{(p-1)} - \mathbf{A}_t^{(p-1)})$. Therefore, the derivative can be split into the sum $\dot{\mathbf{Z}}_{1t}\mathbf{Z}_{2t} + \mathbf{Z}_{1t}\dot{\mathbf{Z}}_{2t}$: the first summand corresponds to 14 and the second summand to 15. In the latter case $\mathbf{dda}_t^{(p-1)}$ is the (second order) derivative of the outer (sigmoid) activation function and $\mathbf{W}^{(p-1)\prime}\mathbf{dA}_t^{\mathbf{f}(p-2)\prime}$ is the derivative of the inner function(s). The $i$-th component $W_{ij}^{(p)}$ of $\mathbf{W}_j^{(p)}$ connects the derivative $dA^{b(p)}_{jt}$ of the output neuron $o_{jt}$ 'backwards' to the $i$-th neuron of layer $p-1$; similarly, the $i$-th row of $\mathbf{W}^{(p-1)\prime}$ collects, weights and connects the derivatives $\mathbf{dA}_t^{\mathbf{f}(p-2)}$ of the previous layer $p-2$ 'forward' to the $i$-th neuron of layer $p-1$. Digging-out dimensions, $\mathbf{W}_j^{(p)}$ is a column vector of dimension $n_{p-1}$, $\mathbf{ddA}_{jt}^{(p)\prime}$ is a row-vector of dimension $n$ so that the right-hand side of 14 is of dimension $(n, n_{p-1})$, after transposition. On the other hand, $\mathbf{W}^{(p-1)\prime}\mathbf{dA}_t^{\mathbf{f}(p-2)\prime}$ is of dimension $(n_{p-1}, n)$ and therefore 15 is of dimension $(n, n_{p-1})$ also, after transposition. If $p > 2$, then we can iterate backwards from (hidden-) layer $k+1$ to $k$ according to

$$\begin{aligned}
\mathbf{dda}_t^{(k)} &= \mathbf{A}_t^{(k)} \cdot (\mathbf{e}^{(k)} - \mathbf{A}_t^{(k)}) \cdot (\mathbf{e}^{(k)} - 2\mathbf{A}_t^{(k)}) \\
\mathbf{ddA}_{jt}^{(k)} &= \left( \left( \mathbf{W}^{(k+1)} \mathbf{ddA}_{jt}^{(k+1)\prime} \right) \cdot \mathbf{A}_t^{(k)} \cdot (\mathbf{e}^{(k)} - \mathbf{A}_t^{(k)}) \right)' \qquad (16) \\
&\quad + \left( \left( \mathbf{W}^{(k+1)} \mathbf{dA}^{\mathbf{b}(k+1)}_{jt} \right) \cdot \mathbf{dda}_t^{(k)} \cdot \left( \mathbf{W}^{(k)\prime} \mathbf{dA}_t^{\mathbf{f}(k-1)\prime} \right) \right)' \qquad (17)
\end{aligned}$$

The $i$-th neuron of layer $k$ is connected to the next layer's derivatives by the $i$-th row of $\mathbf{W}^{(k+1)}$ and to the previous layer's derivatives by the $i$-th column of $\mathbf{W}^{(k)}$. The row-wise products '$\cdot$' with the derivatives of the outer (sigmoid activation) functions $\mathbf{A}_t^{(k)} \cdot (\mathbf{e}^{(k)} - \mathbf{A}_t^{(k)})$ in 16 and $\mathbf{dda}_t^{(k)}$ in 17 then complete the compound chain-rule. Digging-out dimensions implies that $\mathbf{ddA}_{jt}^{(k)}$ is $(n, n_k)$. This generic expression for $\mathbf{ddA}_{jt}^{(k)}$ simplifies in the case of the first hidden layer $k = 1$:

$$\begin{aligned}
\mathbf{dda}_t^{(1)} &= \mathbf{A}_t^{(1)} \cdot (\mathbf{e}^{(1)} - \mathbf{A}_t^{(1)}) \cdot (\mathbf{e}^{(1)} - 2\mathbf{A}_t^{(1)}) \\
\mathbf{ddA}_{jt}^{(1)} &= \left( \left( \mathbf{W}^{(2)} \mathbf{ddA}_{jt}^{(2)\prime} \right) \cdot \mathbf{A}_t^{(1)} \cdot (\mathbf{e}^{(1)} - \mathbf{A}_t^{(1)}) \right)' \qquad (18) \\
&\quad + \left( (\mathbf{W}^{(2)} \mathbf{dA}^{\mathbf{b}(2)}_t) \cdot \mathbf{dda}_t^{(1)} \cdot \mathbf{W}^{(1)} \right)'
\end{aligned}$$

where $\mathbf{dA}_t^{\mathbf{f}0} = \mathbf{Id}$ because the input-layer is exempt of activation functions. If the net is made of a single hidden layer then this is also the first as well as the last (hidden) layer so that the above simplification can be merged with 14 and 15. Finally, at the input layer $k = 0$ we obtain

$$\mathbf{ddA}_{jt}^{(0)} = \left( \mathbf{W}^{(1)} \mathbf{ddA}_{jt}^{(1)\prime} \right)' = \mathbf{W}^{(1)} \mathbf{ddA}_{jt}^{(1)} = \mathbf{QPD}_{jt,-1} \qquad (19)$$

because the input-layer is not equipped with an activation function so that the identity **Id** can be substituted for the first order derivative $\mathbf{A}_t^{(0)} \cdot (\mathbf{e} - \mathbf{A}_t^{(0)})$ and that the second-order derivatives vanish $\mathbf{dda}_t^{(0)} = \mathbf{0}$. The $(n, n)$-dimensional $\mathbf{ddA}_{jt}^{(0)}$ is symmetric and corresponds to the QPD, at least up to the intercept (symbolized by the additional -1 in the QPD-subscript of 19). For the latter

$$b_{jt} := o_{jt} - \sum_{i=1}^{n} w_{ijt} x_{it} = o_{jt} - \mathbf{LPD}_{jt,-1}\mathbf{x}_t$$

where $\mathbf{LPD}_{jt,-1}$ designates the $t$-th row vector, without intercept, and $\mathbf{x}_t$ is the $t$-th data column-vector. The corresponding QPD-entries or partial derivatives are

$$\nabla \mathbf{b}_{jt} := \mathbf{LPD}_{jt,-1} - \left(\mathbf{QPD}_{jt,-1}\mathbf{x}_t + \mathbf{LPD}_{jt,-1}\right) = -\mathbf{QPD}_{jt,-1}\mathbf{x}_t$$

thus completing our specification.

In analogy to section 3.3, second-order derivatives of arbitrary twice differentiable X-functions $xf(o_{jt})$ of the $j$-th output neuron can be obtained by substituting the composite activation function $xf(\sigma(\cdot))$ for $\sigma(\cdot)$ at the output neurons. Specifically, consider the derivative of the backward-expression $\dot{xf}(A_{jt}^{(p)}) dA^{b}{}_{jt}^{(p)}$ for the $j$-th output neuron in 11:

$$\mathbf{dd}(\mathbf{xf} \circ \mathbf{A})_{jt}^{(p)} \quad := \quad \dot{xf}(A_{jt}^{(p)}) \mathbf{ddA}_{jt}^{(p)} + \ddot{xf}(A_{jt}^{(p)}) \mathbf{dA}^{\mathbf{f}}{}_{jt}^{(p)} dA^{b}{}_{jt}^{(p)} \tag{20}$$

where $dA^{b}{}_{jt}^{(p)} = A_{jt}^{(p)}(1 - A_{jt}^{(p)})$, $\mathbf{ddA}_{jt}^{(p)}$ and $\mathbf{dA}^{\mathbf{f}}{}_{jt}^{(p)}$ are defined by 6, 13 and 5: the column vectors $\mathbf{ddA}_{jt}^{(p)}$ and $\mathbf{dA}^{\mathbf{f}}{}_{jt}^{(p)}$ are both of dimension $n$. In order to obtain the requested compound second-order partial derivatives of $xf(o_{jt})$, the generalized expression 20 must be substituted for $\mathbf{ddA}_{jt}^{(p)}$ in 14 and propagated backwards through 16 to 19.

To conclude, the time-dependent $\mathbf{QPD}_{jt}$ differs from the traditional time invariant parameter-Hessian, i.e. second-order partial derivatives of the mean-square error with respect to net-parameters as found in optimization and inference, thus motivating the above derivations.

### 3.5 Discrete Proxies

The above closed-form expressions for the LPD can be approximated by the discrete derivative

$$\Delta_{it}(\delta | \delta > 0) := \frac{o(x_{it} + \delta) - o(x_{it})}{\delta}$$

which allows for straightforward extensions of our approach to finite-sized perturbations (discrete-valued data or classes) and to alternative machine learning techniques. However, finite-sized changes introduce new artificial observations $x_{it} + \delta$ which might potentially conflict with the multivariate dependence structure of the data; moreover, discrete proxies $\Delta_{it}^{xf}(\delta | \delta > 0)$ are reliant on the selection of $\delta$ as well as on numerical precision (numerical cancellation); finally, higher order derivatives such as the above QPD typically magnify numerical issues when discretized.

### 3.6 Analyzing and Monitoring the Entire Net-Structure

In principle, the above computations of LPD or QPD could be interrupted at any hidden-layer $k$, $1 \leq k \leq p-1$ so that the impact of each hidden neuron on the net-output could be evaluated at all time points $t = 1, ..., T$. This additional information can be used to highlight topological features such as information paths, i.e. paths of least resistance, and changes thereof which are potentially indicative of non-stationarity or non-linearity of the underlying data generating process. In this sense, topological changes are potentially indicative of conditions affecting normal operation mode and of increased risk. We here leave this topic for future research and proceed by illustrating the proposed time-series XAI-tool and its application to risk-management.

# 4 Interpretability: LPD

For illustration, we derive LPD and QPD for the Bitcoin crypto-currency (BTC) whereby the neural net relies on [29], see fig.1. The net is richly parameterized for the purpose at hand, with a total of $6 * 100 + 100 = 700$ weights and $100 + 1 = 101$ biases, and it is a 'black-box' in the sense that fitted parameters can not be interpreted in some straightforward meaningful way. Training relies on forecasting next day's return (mean-square error norm) based on the last six lagged returns as determined by statistical analysis, see [29]. The training sample covers an episode of roughly four years, from the first quotation on the Bitstamp crypto-exchange in 2014 up to a peak of the currency in December 2017 and the test sample, which stretches up to 2023-03-27, is subject to severe draw-downs and strong recoveries providing ample opportunity for 'smart' market-positioning. While alternative design-considerations, such as training-samples or net architectures, can have an impact on net-performances, the proper explainability aspect remains mostly unaffected by our choice.

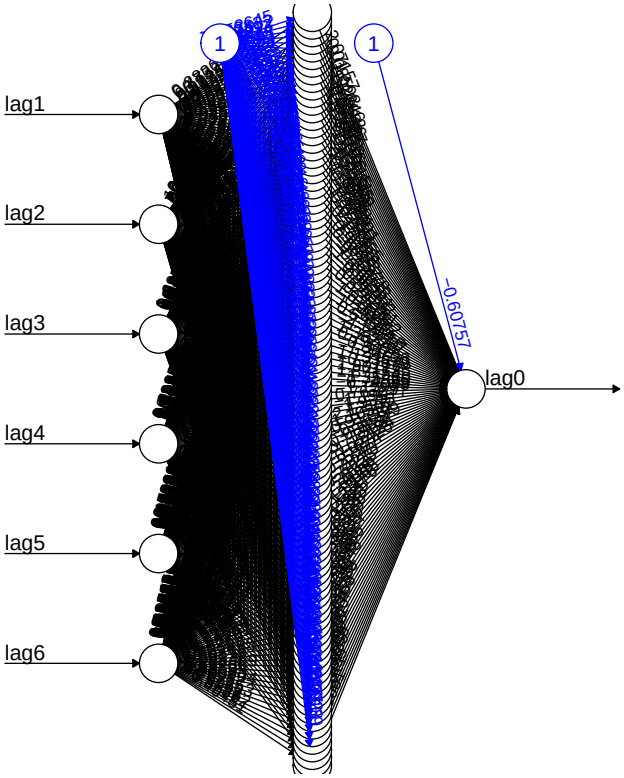


Figure 1: Neural net BTC: feedforward net with a single hidden-layer of dimension 100 and an input layer of dimension six, collecting the last six days of return data

Fig.2 displays cumulated out-of-sample trading performances[1] of the net which are benchmarked against a simple buy-and-hold strategy (black line): the net seems to stabilize performances, as indicated by a comparison of Sharpe ratios. In order to gain additional insight we now compute its LPD, see fig.3.

[1]Trading is based on the simple sign-rule: buy or sell depending on the sign of the out-of-sample net forecast.

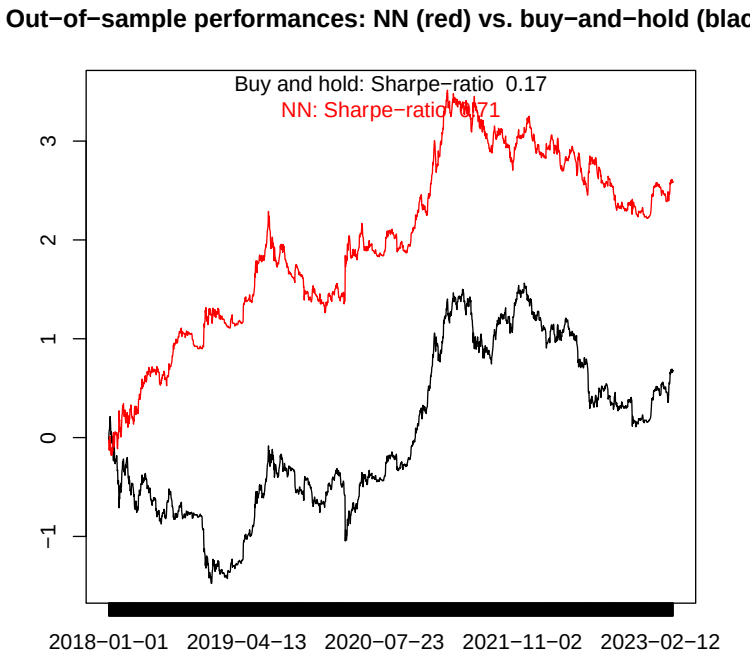


Figure 2: Cumulated log-performances out-of-sample based on sign-rule (buy or sell depending on sign of forecasted return): neural net (red) vs. buy-and-hold (black)

Remarkably, the plot suggests that the net can be

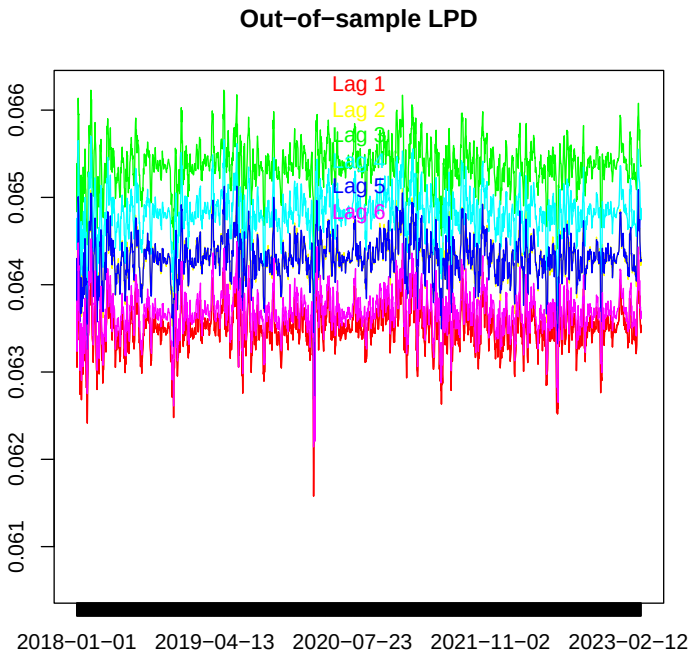


Figure 3: Out-of-sample LPD for the lagged BTC input data

approximated by a simple equally-weighted MA(6)

$$o_t = \mathbf{LPD}_t \begin{pmatrix} 1 \\ \mathbf{x}_t \end{pmatrix} \approx 0.0017 + 0.064 \sum_{j=1}^{6} x_{jt} \tag{21}$$

where the fixed MA-coefficients correspond to the mean-LPD, aggregated over time and across predictors (the fixed intercept is the mean of 1). We can now assign trustworthiness to the net by a statistical analysis of the pertinence of the equally-weighted MA(6) forecast heuristic, see [29] for corresponding results.

# 5 Risk-Management: QPD

## 5.1 Introduction

Risk-management concerns the analysis and mitigation of down-side risks related to an exposition to market dynamics. A well-known strategy consists in a broad diversification of investments across assets with different responses to shocks. We here propose an alternative approach relying on the QPD, whereby our measure of non-linearity serves as an indicator for changing conditions and higher uncertainty. Fig.4 displays the mean-QPD, aggregated across lagged explanatory variables (BTC-returns), together with a rolling upper quantile: exceedances of the quantile by the QPD mark episodes of stronger non-linearity of the net. If we are willing to interpret such episodes in terms of increased market-complexity, then a possible conclusion would be to down-size market-exposure accordingly.

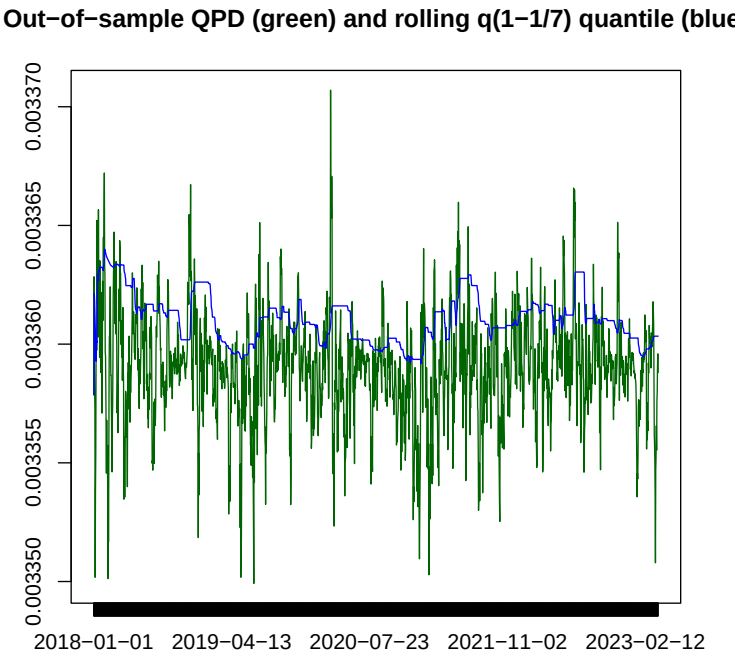


Figure 4: Out-of-sample QPD: mean accross all (lagged) BTC-inputs (dark green) and rolling q(1-1/7) quantile (blue). Non-linearity is considered strong if the QPD lies above the quantile.

Fig.5 benchmarks the resulting exit-strategy, based on quantile-exceedances in fig.4[2]: the shaded areas correspond to exit-signals, lagged by one day (series are arbitrarily scaled so as to fit into a single plot). For a better evaluation, fig. 6 displays next day's cumulated BTC log-returns conditional on today's QPD being either large (non-linear), small (linear) or mid-sized: the drifts in the top panels have correct signs and appear systematic (statistical significance is assessed further down); in contrast, time-points tagged by mid-sized QPD (bottom panel) are less systematic and with 'flatter' dynamics overall[3]. Note that the performance of the active risk-management strategy in fig.5 (red line) corresponds to time-splices of top-right and bottom panels in fig.6, while buy-and-hold (black line) is obtained by splicing all three panels together: the top-left panel (large QPD) corresponds to the increasing spread between buy-and-hold (black) and the active strategy (red) in fig.5. Also, the three panels in fig.6 might suggest alternative down-sizing or leveraging rules than the proposed simple risk-management exit-strategy in fig.5.

[2]The empirical quantile $q_t(1-1/7)$ of the QPD is based on a rolling window of length one quarter and its probability-level 1-1/7 assumes a market-exit probability of 1/7 i.e. one day per week in the mean.

[3]The sample-length $T(1-2/7)$ in the bottom panel exceeds the lengths $T/7$ in the top panels.

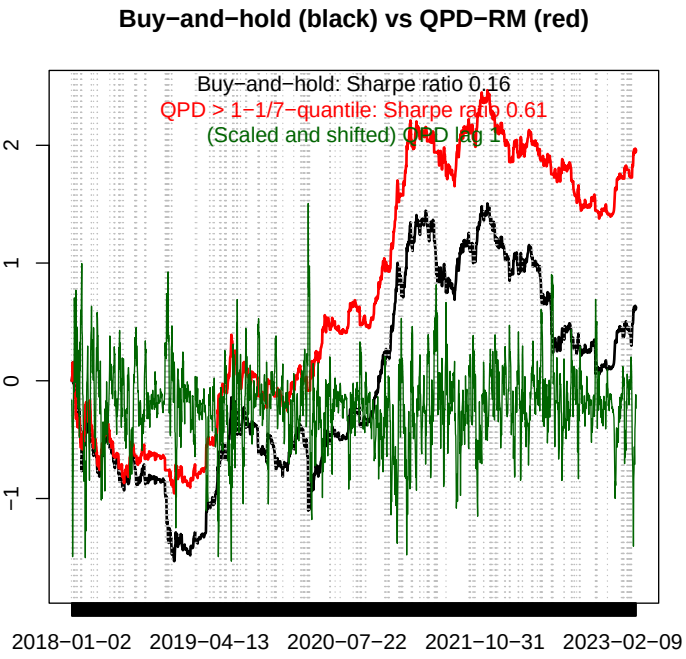


Figure 5: Buy-and-hold (black) vs. out-of-sample (mean-) QPD 'Risk-Management' strategy (red): exits (shaded in grey) occur if the lagged out-of-sample QPD (green) exceeds the rolling q(1-1/7) quantile. Series are arbitrarily scaled so as to fit into a single plot.

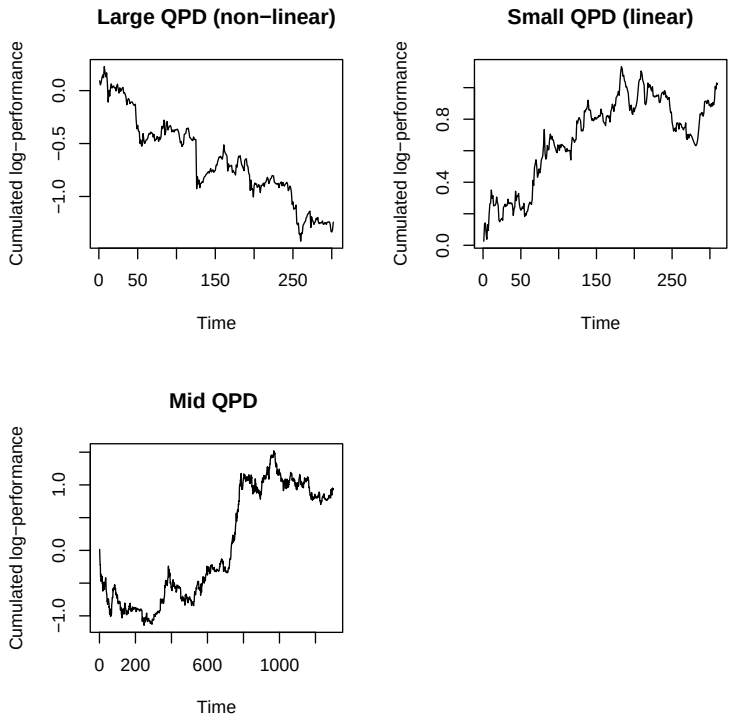


Figure 6: Cumulated next day's BTC return conditional on today's QPD: large QPD (top left: |QPD|>q(1-1/7)), small QPD (top right: |QPD|<q(1/7)) and mid-sized QPD (bottom: q(1/7)<|QPD|<q(1-1/7))

Table 1 quantifies the observed effects: according to hit ratios in the first column, today's QPD is uninformative about the sign of next-day's return; instead, the degree of non-linearity as measured by the QPD seems to support information about the skewness of next day's return and, by extension, about its *average* value (second column). While the individual means in the

| | Proportion of positive signs | Average next days' returns |
|---|---|---|
| Critical time points | 53% | -0.411% |
| Neutral time points | 51% | 0.072% |
| Auspicious time points | 52.3% | 0.329% |
| All time points | 51.5% | 0.037% |

Table 1: Proportions of positive signs (first column) and average next days' returns (second column) based on critical time points (|QPD| > q(1-1/7): strong non-linearity), neutral time points (q(1/7) < |QPD| < q(1-1/7): medium non-linearity), auspicious time points (|QPD| < q(1/7): weak non-linearity) and all time points.

second column are not statistically significant, due in part to shortened lengths $T/7$, the mean of the concatenation, obtained by merging critical and (sign-transformed) auspicious time points, is significant with an absolute t-value of 2, thus providing additional statistical evidence of the proposed concept.

# 6 Conclusion

The need for opening the "black box" of machine learning has gained great traction in the past decade, as the need for controlling these models and regulatory concerns have increased. Solutions to this issue fall within the so-called explainable AI field which aims at producing methods that enable users to understand and appropriately trust outputs generated from AI-based systems. Although the literature is offering an ever-growing suite of such XAI techniques, research on XAI methods specifically suited for financial time series remains limited. Furthermore, classical XAI approaches and their implementations cannot be easily adjusted to correctly account for the time dependency of financial data which in turn makes their application to this domain very limited. We here propose a non-invasive time-series approach to explainability which preserves data-integrity and dependence-structure by emphasizing partial derivatives of the net-output. First-order derivatives or LPD trace the momentary sensitivity of the net to the explanatory variables and, by extension, the multivariate dependence structure of the data; second-order derivatives or QPD measure the susceptibility of the LPD to changes of the explanatory variables. Exact closed-form expressions are derived for LPD and QPD and an application to BTC illustrates explainability and risk-management whereby increased non-linearity is interpreted as an indicator for changing market-conditions and higher risk.

# References


[1] Alexiei Dingli and Karl Fournier. Financial time series forecasting - a machine learning approach. *Machine Learning and Applications: An International Journal*, 4:11–27, 09 2017.

[2] Luca Di Persio and Oleksandr Honchar. Recurrent neural networks approach to the financial forecast of google assets. 2017.

[3] Jaydip Sen and Sidra Mehtab. Design of robust deep learning models for stock price prediction. 05 2021.

[4] Parley Ruogu Yang. Forecasting high-frequency financial time series: an adaptive learning approach with the order book data, 2021.

[5] Gunho Jung and Sun-Yong Choi. Forecasting foreign exchange volatility using deep learning autoencoder-lstm techniques. *Complexity*, 2021:1–16, 03 2021.

[6] Deniz Can Yıldırım, Ismail Hakkı Toroslu, and Ugo Fiore. Forecasting directional movement of Forex data using LSTM with technical and macroeconomic indicators. *Financial Innovation*, 7(1):1–36, December 2021.

[7] Jerzy Korczak and Marcin Hernes. Deep learning for financial time series forecasting in a-trader system. pages 905–912, 09 2017.

[8] Zexin Hu, Yiqi Zhao, and Matloob Khushi. A survey of forex and stock price prediction using deep learning. *Applied System Innovation*, 4(1):9, Feb 2021.

[9] Spyros Makridakis, Evangelos Spiliotis, and Vassilios Assimakopoulos. The m4 competition: 100,000 time series and 61 forecasting methods. *International Journal of Forecasting*, 36(1):54–74, 2020. M4 Competition.

[10] Spyros Makridakis, Evangelos Spiliotis, and Vassilios Assimakopoulos. The m5 competition: Background, organization, and implementation. *International Journal of Forecasting*, 2021.

[11] Pantelis Linardatos, Vasilis Papastefanopoulos, and Sotiris Kotsiantis. Explainable ai: A review of machine learning interpretability methods. *Entropy*, 23(1), 2021.

[12] Richard Dazeley, Peter Vamplew, Cameron Foale, Charlotte Young, Sunil Aryal, and Francisco Cruz. Levels of explainable artificial intelligence for human-aligned conversational explanations. *Artificial Intelligence*, 299:103525, Oct 2021.

[13] David Gunning, Mark Stefik, Jaesik Choi, Timothy Miller, Simone Stumpf, and Guang-Zhong Yang. Xai—explainable artificial intelligence. *Science Robotics*, 4:eaay7120, 12 2019.

[14] Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador García, Sergio Gil-López, Daniel Molina, Richard Benjamins, Raja Chatila, and Francisco Herrera. Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai, 2019.

[15] Szymon Maksymiuk, Alicja Gosiewska, and Przemyslaw Biecek. Landscape of r packages for explainable artificial intelligence. 09 2020.

[16] Wojciech Samek, Gregoire Montavon, Sebastian Lapuschkin, Christopher J. Anders, and Klaus-Robert Muller. Explaining deep neural networks and beyond: A review of methods and applications. *Proceedings of the IEEE*, 109(3):247–278, Mar 2021.

[17] Veerappa Manjunatha, Anneken Mathias, Burkart Nadia, and F. Huber Marco. Validation of xai explanations for multivariate time series classification in the maritime domain. *Journal of Computational Science, Elsevier, vol. 58*, 2022.

[18] Balázs Hidasi and Csaba Gáspár-Papanek. Shifttree: An interpretable model-based approach for time series classification. In Dimitrios Gunopulos, Thomas Hofmann, Donato Malerba, and Michalis Vazirgiannis, editors, *Machine Learning and Knowledge Discovery in Databases*, pages 48–64, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[19] En Yu Hsu, Chien-Liang Liu, and Vincent Shin-Mu Tseng. Multivariate time series early classification with interpretability using deep learning and attention mechanism. In Min-Ling Zhang, Zhi-Hua Zhou, Zhiguo Gong, Qiang Yang, and Sheng-Jun Huang, editors, *Advances in Knowledge Discovery and Data Mining - 23rd Pacific-Asia Conference, PAKDD 2019, Proceedings*, Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), pages 541–553, Germany, January 2019. Springer Verlag. null ; Conference date: 14-04-2019 Through 17-04-2019.

[20] Cedric Schockaert, Reinhard Leperlier, and Assaad Moawad. Attention mechanism for multivariate time series recurrent model interpretability applied to the ironmaking industry, 2020.

[21] Thomas Rojat, Raphaël Puget, David Filliat, Javier Del Ser, Rodolphe Gelin, and Natalia Díaz-Rodríguez. Explainable artificial intelligence (xai) on timeseries data: A survey, 2021.

[22] Christoph Molnar, Gunnar König, Julia Herbinger, Timo Freiesleben, Susanne Dandl, Christian A. Scholbeck, Giuseppe Casalicchio, Moritz Grosse-Wentrup, and Bernd Bischl. General pitfalls of model-agnostic interpretation methods for machine learning models, 2021.

[23] Christoph Molnar. *Interpretable Machine Learning*. 2019.

[24] Hugh Chen, Joseph D. Janizek, Scott Lundberg, and Su-In Lee. True to the model or true to the data?, 2020.

[25] I. Elizabeth Kumar, Suresh Venkatasubramanian, Carlos Scheidegger, and Sorelle Friedler. Problems with shapley-value-based explanations as feature importance measures, 2020.

[26] Sven F. Crone, Michèle Hibon, and Konstantinos Nikolopoulos. Advances in forecasting with neural networks? empirical evidence from the nn3 competition on time series prediction. *Journal of Forecasting, Elsevier, vol. 27(3), pages 635-660*, 2011.

[27] Smyl Slawek. Advances in forecasting with neural networks? empirical evidence from the nn3 competition on time series prediction. *Journal of Forecasting, Elsevier, vol. 36(1), pages 75-85*, 2020.

[28] C.W.J. Granger. The typical spectral shape of an economic variable. *Econometrica*, 34:150–161, 1966.

[29] Marc Wildi and Nils Bundi. Bitcoin and market-(in)efficiency: a systematic time series approach. *Digital finance 1 (2)*, 2019.