

USING NEURAL NETWORKS AND OTHER MACHINE LEARNING ALGORITHMS WITH NON-STANDARD LOSS FUNCTIONS IN REALIZED VOLATILITY FORECASTING

Iakov GRIGORYEV*

New Economic School

February 3, 2021

Abstract

Neural networks have recently become widespread since they allow approximating a nonlinear dependence of any complexity. Many researchers began to use them for modeling and forecasting various time series. Such machine learning algorithm as random forest is also used to predict time series. This paper compares the predictive power of various machine learning algorithms, including neural networks and random forest, with standard econometric approaches, such as the MEM and the HAR models, while forecasting realized volatility. A distinctive feature of the work is the use of non-standard (asymmetric) loss functions such as the LinEx function and the asymmetric least squares function. The result of the work is that the considered machine learning methods are not able to outperform econometric approaches in predictive power, both with symmetric and with asymmetric loss functions.

*Master's Student at NES. Email: igrigoryev@nes.ru

Contents

1	Introduction	2
2	Literature Review	3
3	Realized Volatility	3
4	Models	5
4.1	The HAR Model	5
4.2	The MEM Model	6
4.3	The Random Forest Model	6
4.4	The LSTM Model	8
5	Loss Functions	10
5.1	The Mean Squared Error (MSE) Loss Function	10
5.2	The Linear Exponential (LinEx) Loss Function	11
5.3	The Asymmetric Least Squares (ALS) Loss Function	11
6	Data	12
7	Results	12
8	Conclusion	16
	References	16

1 Introduction

Volatility forecasting is important not only in academia, but also in business, in investment banking, in trading, and even in government decision making. Moreover, option prices, Value at Risk, Sharpe ratio and other indicators are highly dependent on volatility, which reveals the necessity of its accurate forecasts. For example, volatility is used by option traders as an input for option pricing using Black–Scholes’ (Black & Scholes (1973)) and Heston’s (Heston (1993)) models.

When high-frequency data appeared, the realized volatility (RV) measure was introduced by Andersen et al. (2003). Unlike GARCH-like measures, RV is model-free, and it is observable. In this paper we compare recently developed machine learning algorithms (such as neural networks and random forests) and classical econometric approaches (such as the HAR and the MEM models) in forecasting RV. Moreover, we will consider using non-standard loss functions — the LinEx loss function (see references in Khatun & Matin (2020)) and the asymmetric least squares (ALS) loss function (as defined in Tian (2009)).

Luong & Dokuchaev (2018) researched the possibility to forecast RV using the random forest algorithm. Vortelinos (2017) compared the HAR model against the GARCH model, the feedforward neural network model and the principal components combining model (which will not be considered in this paper). Bucci (2020) compared different recurrent neural network (RNN) architectures applied to RV in terms of predictability performance.

The goals of this paper can be formulated as follows:

1. Compare the HAR and the MEM models with the random forest algorithm and the LSTM model using the symmetric MSE loss function and criterion.
2. Compare the HAR and the LSTM models using the LinEx loss function and criterion.
3. Compare the HAR and the LSTM models using the ALS loss function and criterion.

This paper is structured as follows. In Section 2 we provide a literature review on all topics we cover in this paper: volatility, models (for loss functions see references in already presented papers). In Section 3 we give a definition of RV. In Section 4 we introduce description of all models we used to forecast realized volatility. In Section 5 we describe all loss functions we used. In Section 6 we provide a data description used in the paper. In Section 7 there are results of our empirical research and in Section 8 there is a conclusion of the paper.

2 Literature Review

Volatility has been estimated and predicted a lot (see the references in [Andersen & Bollerslev \(1997\)](#), [Dokuchaev \(2014\)](#)).

The ARCH and the GARCH models were proposed by [Engle \(1982\)](#) and [Bollerslev \(1986\)](#), respectively. These models were extended in many directions: EGARCH ([Nelson \(1991\)](#)), GJR-GARCH ([Glosten et al. \(1993\)](#)), AGARCH ([Engle \(1990\)](#)), and TGARCH ([Zakoian \(1994\)](#)). But these models cannot be appropriate estimators for RV since they were developed for low-frequency data and cannot capture intradaily information well. And [Vortelinos \(2017\)](#) showed that the family of the GARCH models cannot indeed compete with the HAR model in terms of forecasting. So, we will not consider GARCH-like models in this paper.

As we already mentioned in introduction, [Andersen et al. \(2003\)](#) proposed the RV measure. The HAR model, which was introduced by [Corsi \(2003\)](#), was developed specifically for estimating and predicting RV. The MEM model, presented by [Engle \(2002\)](#), was not developed specifically for the RV estimation, but it is widely used for RV. We will use the MEM and the HAR models in this paper.

The algorithm of random forests was introduced by [Breiman \(2001\)](#) and it is basically an ensemble of some number of decision tree algorithms, which were presented in [Breiman et al. \(1984\)](#). The random forest algorithm will be used in the paper.

The feedforward neural network model, according to [Vortelinos \(2017\)](#), is not a perfect choice for the RV forecasting. On the contrary, the recurrent neural network (RNN) architectures are applied for time series estimation and prediction not only in academia, but also in business. The LSTM neural network architecture (presented in [Hochreiter \(1997\)](#)) is one of the most popular RNNs, and it will be used in this paper.

3 Realized Volatility

Denote P_t the asset price at time $t \in [0, T]$, $p_t = \log P_t$. The asset return is $r_{t,\Delta t} = p_{t+\Delta t} - p_t$ for some Δt such that $t + \Delta t \leq T$. The assumption is that p_t can be represented by the following diffusion Ito equation:

$$dp_t = \mu_t dt + \sigma_t dW_t, \quad 0 \leq t \leq T,$$

where W_t is Brownian motion, μ_t and σ_t are predictable processes with σ_t being the standard deviation of dp_t and independent of dW_t . The term $\mu_t dt$ is called drift. From the diffusion

equation it follows that

$$r_{t,\Delta t} = p_{t+\Delta t} - p_t = \int_0^{\Delta t} \mu_{t+\tau} d\tau + \int_0^{\Delta t} \sigma_{t+\tau} dW(t+\tau).$$

It is true not only for the segment $[t, t + \Delta t]$, but also for any its subsegment $[\bar{t}, \bar{t} + \delta t] \subset [t, t + \Delta t]$. If δt is small enough, the drift term can be excluded, and we have

$$r_{\bar{t},\delta t} \approx \int_0^{\delta t} \sigma_{\bar{t}+\tau} dW(\bar{t}+\tau) = \sigma_{\bar{t}+\epsilon\delta t} (W_{\bar{t}+\delta t} - W_{\bar{t}}),$$

where $0 < \epsilon < 1$ (it follows from the first mean theorem).

Since W_t is Brownian motion, $W_{\bar{t}+\delta t} - W_{\bar{t}} \sim N(0, \delta t)$, thus, $(W_{\bar{t}+\delta t} - W_{\bar{t}})^2 \sim \delta t \times \chi^2(1)$, which has a mean of δt . We can then take δt as an approximate value of $(W_{\bar{t}+\delta t} - W_{\bar{t}})^2$ when δt is small enough. So, we have

$$r_{\bar{t},\delta t}^2 \approx \sigma_{\bar{t}+\epsilon\delta t}^2 \delta t = \sigma_{\bar{t}+\epsilon\delta t}^2 \int_0^{\delta t} 1 d\tau = \frac{\sigma_{\bar{t}+\epsilon\delta t}^2}{\sigma_{\bar{t}+\gamma\delta t}^2} \int_0^{\delta t} \sigma_{\bar{t}+\tau}^2 d\tau, \quad (3.1)$$

where $0 < \gamma < 1$ (again, from the first mean theorem).

Consider even splitting $\left\{ t_i = t + \frac{\Delta t}{m} \times i \right\}_{i=1}^m$ of the segment $[t, t + \Delta t]$. Then realized variance can be calculated as follows:

$$RV_t = \sum_{i=0}^{m-1} r_{t_i, \frac{\Delta t}{m}}^2.$$

Why is it important? From equation (3.1) it follows that

$$RV_t \xrightarrow{p} \int_0^{\Delta t} \sigma_{t+\tau}^2 d\tau, \quad m \rightarrow +\infty$$

as for each $i = 0, \dots, m-1$

$$\frac{\sigma_{t_i+\epsilon_i \frac{\Delta t}{m}}^2}{\sigma_{t_i+\gamma_i \frac{\Delta t}{m}}^2} \longrightarrow 1, \quad m \rightarrow +\infty.$$

Integrated variance (IV) is defined as follows:

$$IV_t = \int_0^{\Delta t} \sigma_{t+\tau}^2 d\tau,$$

and we showed that RV_t is a consistent estimator for IV_t (we assume the absence of jumps, according to the diffusion equation). Integrated variance is of a particular interest in academia and in business since it measures the variation of an asset within the time period Δt . Therefore, the realized variance prediction is important in research and in applications.

Finally, realized volatility (RV) is simply a square root of realized variance (and from now on, we denote it with RV_t). Starting from the next section let us assume that $\Delta t = 1$ day (so that each day we calculate RV based on intradaily data).

4 Models

4.1 The HAR Model

Let $RV_t^n = \frac{\sum_{i=0}^{n-1} RV_{t-i}}{n}$. Then, the daily Heterogeneous Autoregressive (HAR) model can be expressed as follows:

$$RV_t = \beta_0 + \beta_d RV_{t-1}^1 + \beta_w RV_{t-1}^5 + \beta_m RV_{t-1}^{22} + \epsilon_t, \quad t \in [22, T],$$

where subscripts "d", "w", and "m" stand for "day", "week", and "month", respectively. $\mathbb{E}(\epsilon_t | I_{t-1}) = 0$ for $t \in [22, T]$, where $I_{t-1} = \{RV_0, \dots, RV_{t-1}\}$. There are some extensions of this model (i.e. taking into account jumps or asymmetry in the relationship between returns and volatility), but we will not consider them in this paper.

We will estimate the HAR model with three different loss functions:

1. Mean Squared Error Loss Function (MSE);
2. Linear Exponential Loss Function (LinEx);
3. Asymmetric Least Squares Loss Function (ASL).

All of them will be considered in Section 5. Note that for asymmetric loss functions the conditional mean predictor (which is 'tied' to the HAR model) is not the optimal predictor anymore, but our purpose is to compare the predictive power of models, not their theoretical properties. The consideration of other models which were designed to be estimated with these asymmetric loss functions is included in the future research plans.

In the HAR model we will use logs of RV to deal with negative predictions.

4.2 The MEM Model

The Multiplicative Error Model (MEM) is used only for time series with non-negative elements, and RV is an example of it. The model has the following expression:

$$RV_t = \psi_t \eta_t, \quad t \in [0, T],$$

where $\psi_t = \mathbb{E}_{t-1} RV_t$, and η_t is an i.i.d. process with a mean 1 (and also positive elements). We will use $\exp(1)$ distribution. Thus, while considering the MEM model, we will use exponential quasi-maximum likelihood estimate (QMLE). We also assume that ψ_t can be expressed as follows:

$$\psi_t = \beta_0 + \beta_1 RV_{t-1} + \beta_2 \psi_{t-1}.$$

To guarantee the positivity of forecasts, we should impose $\beta_0 > 0$.

Since the exponential density function is $f(x) = e^{-x}$, log-likelihood function of the whole sample is

$$L = - \sum_{t=1}^T \left(\frac{RV_t}{\psi_t} + \log \psi_t \right),$$

where $\psi_t = \beta_0 + \beta_1 RV_{t-1} + \beta_2 \psi_{t-1}$.

4.3 The Random Forest Model

For this algorithm, as well as for the next LSTM algorithm, we will build a dependency of RV_t on some number k of previous observations $RV_{t-1}, \dots, RV_{t-k}$, where k is some hyperparameter. In this paper we take $k = 10$. So, let us call RV_t a target and $RV_{t-1}, \dots, RV_{t-k}$ features. For each $t \geq k$ we have a separate object with a target and features. Denote our sample as X (consisting of vectors $((RV_{t-1}, \dots, RV_{t-k}), RV_t)$, $t \in [k, T]$).

To fully describe the random forest algorithm, we first should define the decision tree method:

1. Let N be the root node of the tree with all available data.
2. Check whether some stop condition holds. If so, make N a leaf node and assign to it the average target of the objects in this node (meaning that while forecasting, if an out-of-sample object ends up being in this node, than we predict this value as a target). Otherwise, go ahead to the next step.
3. Select the feature F and a threshold T , such that all data is separated into two subsets: X_{TRUE} and X_{FALSE} , where data with $F < T$ is going to X_{TRUE} and otherwise in X_{FALSE} . And we select F and T maximizing some criterion $Q(X, F, T)$.

4. Assign a pair (F, T) to the node N , make two new nodes N_{TRUE} and N_{FALSE} , passing to them X_{TRUE} and X_{FALSE} , respectively.
5. Repeat steps 2-4 with both nodes each considered as a new root of a respective subtree.

To finish the definition, we should set a particular stop condition and a criterion $Q(X, F, T)$. The latter is defined as follows:

$$Q(X, F, T) = H(X) - \frac{|X_{TRUE}|}{|X|} H(X_{TRUE}) - \frac{|X_{FALSE}|}{|X|} H(X_{FALSE}),$$

where $H(X) = \min_{c \in \mathbb{Y}} \frac{1}{|X|} \sum_{((x_1^i, \dots, x_k^i), y^i) \in X} L(y^i, c)$, and $L(y, c)$ is some loss function we choose. \mathbb{Y} is a set of possible targets, in our case it is \mathbb{R} . In this paper the standard MSE loss function was used for random forests: $L(y, c) = (y - c)^2$ (note that we use non-standard loss function only for the LSTM model; using it in random forests is a future plan for extension). The stop condition is the following: stop whenever the best splitting into two subsets yields negative criterion value $Q(X, F, T)$. When it is positive, then the process of subtree construction continues. Actually, for checking this condition we need to calculate $Q(X, F, T)$ for each pair of (F, T) to take the best one, so we can use these calculations to select the optimal pair on step 3 (of course, if stop condition does not hold).

Now we can define an algorithm of random forest. Hyperparameters (besides those for decision trees) are n_{tree} (number of trees in the ensemble) and m (the number of features in each tree). The algorithm is the following:

1. Use bootstrap to generate n_{tree} samples from the original one (and use each of them while training a separate decision tree).
2. For i_{tree} from 1 to n_{tree} : train a decision tree on i_{tree} 's generated bootstrap sample, but when selecting the best splitting of each node (i.e. on step 3 of tree construction) use only m features, whose numbers are randomly chosen before each splitting.
3. Return the average of the predictions of all trees (i.e. while forecasting, we get the predictions from each tree and after that the averaged value will be a prediction from a random forest).

As we can see, all predictions in a random forest lie between the smallest and the largest values of sample targets, so there is no need to worry about negative predictions of the RV, since all targets in the sample are positive.

Random forest makes the variance of the overall algorithm smaller, which prevents overfitting. However, it does not change the bias, so it cannot help with underfitting (if it

exists). Therefore, we should use decision trees with large depth (so that they are complex enough not to be underfitted). In fact, in this paper we do not restrict the depth of trees at all (using the only stop condition described above).

In this paper we chose $n_{tree} = 100$ and $m = 10$.

4.4 The LSTM Model

The general neural network consists of the following elements:

- neurons (or nodes), grouped in
- layers, connected to each other;
- weights of neurons and
- (non-linear) activation functions for each layer.

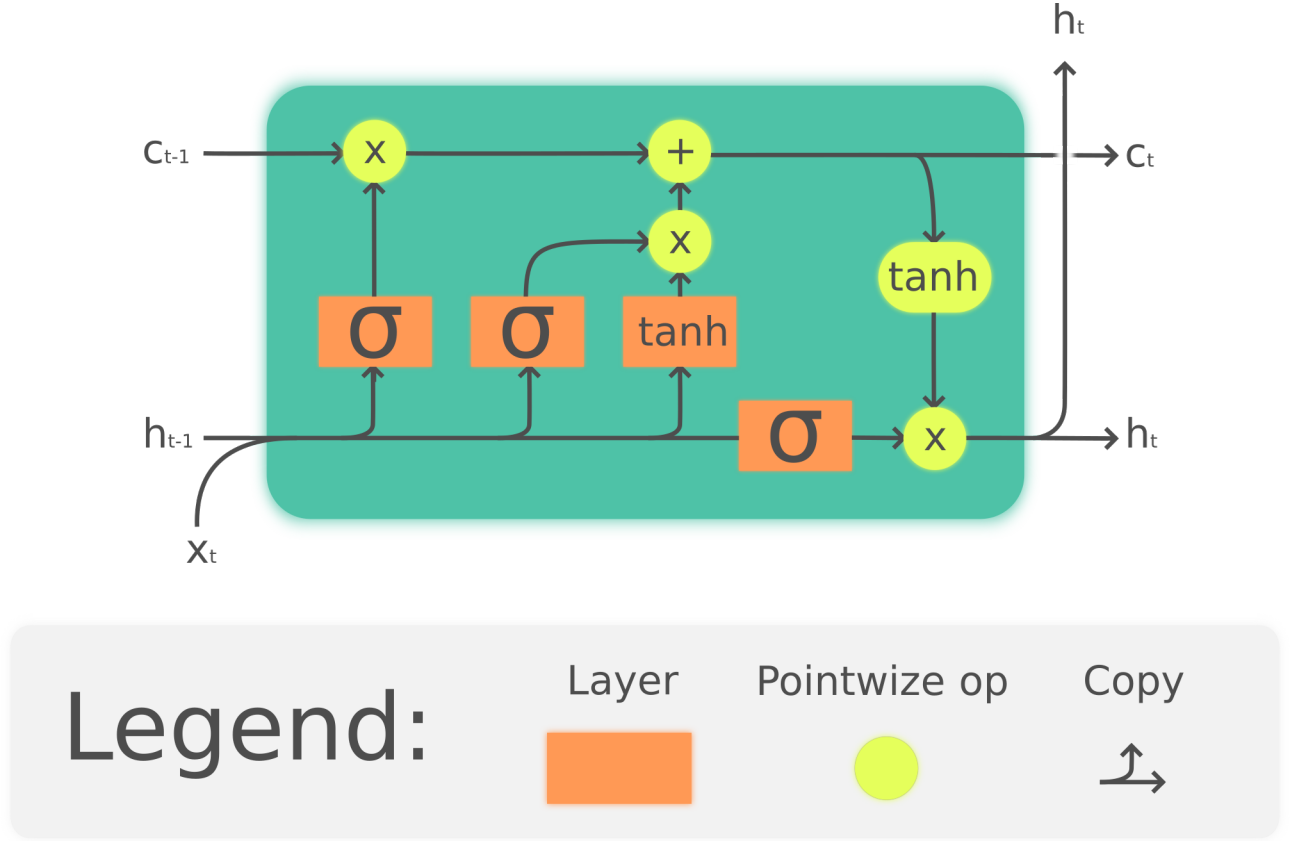
Layers are numerated from 0 (the input of the neural network) to L (the output of the neural network), all layers except the input and the output are called hidden layers. Each node in layer $l - 1$ is connected to each node in layer l via arrows and weights under arrows. Each layer l is specified by an activation function $a_l(\cdot)$, so that the value in the node of layer $l + 1$ is an activation function applied to the linear combination of values of nodes of layer l with weights under arrows entering this node, including a bias term. Activation functions need to be non-linear so that neural network can approximate non-linear dependencies. We will use the only hidden layer in this paper since [Donaldson & Kamstra \(1996\)](#) showed that a single hidden layer neural network can approximate any non-linear function given enough number of neurons in this hidden layer. Overall, the output function of one hidden layer neural network is

$$f(X, W, b) = a_1 \left(b^1 + \sum_{j=1}^q a_0(b_j^0 + XW^0)W_j^1 \right),$$

where $k, q, 1$ are sizes of layers 0, 1, 2 respectively, W^0 (of size (q, k)) is a matrix consisting of weights connecting the input layer and hidden layer, and W^1 (of size $(1, q)$) is a row vector consisting of weights connecting the hidden layer and the output layer, b^0 (of size $(q, 1)$) is a column vector consisting of bias terms in the hidden layer, and b^1 (scalar) is a bias term in the output layer. X (of size (m, k)) is an input matrix consisting of m examples, so that $f(X, W, b)$ has size $(m, 1)$ approximating the target values on these examples. Remember (from the previous point) that k is the number of features (regressors). In this paper we have $k = 10$ and $q = 100$.

As already mentioned in Section 2, feedforward neural networks are not ideal for estimating and forecasting time series. And we will use the Long Short-Term Memory (LSTM) model for it. The hidden layer in this model (which is the only as we already discussed) is represented by the so-called memory block. Each block includes one or more self-connected memory cells and is equipped with three multiplicative units called input, forget and output gates.

Figure 1: LSTM memory cell



Define the following activation functions:

$$\sigma(z) = \frac{1}{1 + e^{-z}},$$

$$\tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}.$$

The illustration of the LSTM memory cell is given on Figure 1 on page 9. x_t denotes the input vector at time t , c_t is a cell state at time t , h_t is a hidden state or an output state at time t . The input gate is i_t , whereas f_t is the forget gate and o_t is the output gate. The entire

process can be described by the following equations:

$$f_t = \sigma(W_f h_{t-1} + U_f x_t + b_f),$$

$$i_t = \sigma(W_i h_{t-1} + U_i x_t + b_i),$$

$$\bar{c}_t = \tanh(W_c h_{t-1} + U_c x_t + b_c),$$

$$c_t = f_t \circ c_{t-1} + i_t \circ \bar{c}_t,$$

$$o_t = \sigma(W_o h_{t-1} + U_o x_t + V_o c_t + b_o),$$

$$h_t = o_t \circ \tanh(c_t),$$

$$\hat{y}_t = h_t.$$

Here \circ is the Hadamard product function. As well as for the HAR model, we will use three loss functions (MSE, LinEx and ASL). And similarly to the HAR model, we will take logs of RV to deal with negative predictions.

5 Loss Functions

Each model we consider (except for the MEM model, where we use QMLE) while being estimated optimizes some cost function:

$$C(y, \hat{y}) = \frac{1}{m} \sum_{i=1}^m L(y^i, \hat{y}^i),$$

where m is a number of examples, and $L(y^i, \hat{y}^i)$ is a loss function, which estimates an error of a prediction on i 'th example (from the true target value). Let us consider some of them.

5.1 The Mean Squared Error (MSE) Loss Function

The most famous and widely used in regression problems loss function is the MSE function:

$$L(y, \hat{y}) = (y - \hat{y})^2.$$

It is continuously differentiable, and it is symmetric (meaning that errors of underestimation and overestimation are symmetric and depend only on the distance between the predicted value and the true value). However, its use often contradicts the reality, e.g. in situations where overprediction is much less preferred than underprediction (or vice versa). Therefore, using

symmetric loss functions (e.g. MSE) in these situations is undesirable. And the next two loss functions are asymmetric (in general).

5.2 The Linear Exponential (LinEx) Loss Function

The LinEx loss function is defined as follows:

$$L(y, \hat{y}) = \exp(\alpha(y - \hat{y})) - \alpha(y - \hat{y}) - 1,$$

where α is a parameter which represents a degree of asymmetry. If $\alpha > 0$, then the loss is approximately exponential for positive errors, and approximately linear for negative errors. Therefore, it is smaller for overprediction than for underprediction. We assume that this is the case with RV: it is much better to overestimate it than to underestimate (e.g. for risk management purposes, where agents are risk averse). Thus, we will use $\alpha = 0.5$ which estimating models (using different values of α is a plan for the future research). We will use the LinEx loss with two models: the HAR and the LSTM models. For the HAR model we need to calculate a derivative of the LinEx cost function with respect to the weights of the model:

$$\frac{\partial C(y, Xw)}{\partial w} = -\frac{1}{m} \left(\alpha X^T e^{y-Xw} + X^T \alpha \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix} \right)$$

(the size of the vector of ones is $(m, 1)$; we calculate a simple sum with it). Here X is a matrix of regressors. For the LSTM model we do not need a derivative since it is calculated automatically in the backpropagation algorithm.

5.3 The Asymmetric Least Squares (ALS) Loss Function

The ALS loss function is defined as follows:

$$L(y, \hat{y}) = \left| \alpha - \mathbb{1}_{(y-\hat{y}<0)} \right| \times (y - \hat{y})^2.$$

As we can see, when $y < \hat{y}$ (i.e. when there is an overprediction), the coefficient is $1 - \alpha$, otherwise α . We assumed that underprediction is less preferred than overprediction, thus, we need to have $\alpha > 1 - \alpha \Leftrightarrow \alpha > 0.5$. In this paper we use $\alpha = 0.7$ (again, using different values of α is a plan for the future research).

Note that this function is continuously differentiable with respect to $y - \hat{y}$ (in $y - \hat{y} = 0$ it is differentiable by definition and the derivative is 0, and both left and right derivatives converge

to 0 as well). Again, we will use ALS loss with two models: the HAR and the LSTM models.

6 Data

We used RV of S&P 500 Index, calculated by [Oxford-Man Institute of Quantitative Finance](#) (ticks of 5 minutes were used to calculate RV of each day). The period is from 1/3/2000 to 1/14/2020, which is splitted into an in-sample and an out-of-sample periods with a proportion of 7 : 3.

All the code and data can be found [here](#).

7 Results

The following models were estimated:

1. the MEM model with the exponential QMLE;
2. the Random Forest model with the MSE loss function;
3. the HAR model with the MSE loss function;
4. the LSTM model with the MSE loss function;
5. the HAR model with the LinEx loss function;
6. the LSTM model with the LinEx loss function;
7. the HAR model with the ALS loss function;
8. the LSTM model with the ALS loss function.

Compare the first four models using MSE for RV_t and for $\log(RV_t)$ on out-of-sample data (we used one-step-ahead predictions):

Model	MSE for RV_t	MSE for $\log(RV_t)$
MEM	4.67×10^{-6}	0.057
Random Forest	1.11×10^{-5}	0.136
HAR	1.21×10^{-5}	0.110
LSTM	3.96×10^{-5}	0.453

As we can see, the MEM model has the best result in both criteria, although RV_t were predicted, not their logs. The HAR and the random forest models are very close, and each model has lower criterion value corresponding to their loss function (the HAR was estimated using the MSE for logs and the Random Forest model used the MSE for RV_t themselves). The LSTM model shows the worst performance. On Figures 2–5 one-step-ahead predictions (on the first 200 days of out-of-sample data) of all four models are represented.

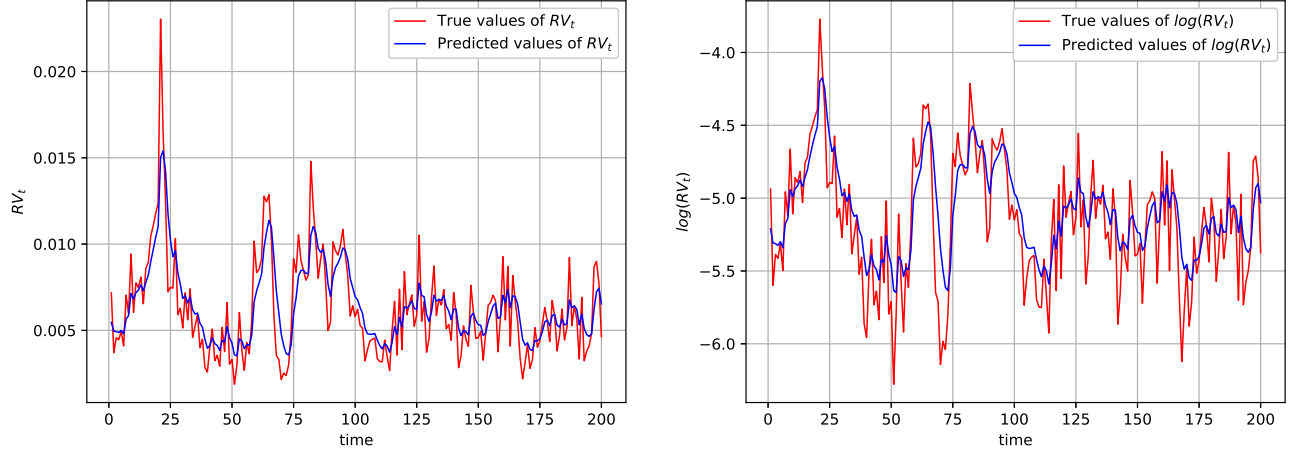


Figure 2: The MEM model with QMLE, one-step-ahead predictions

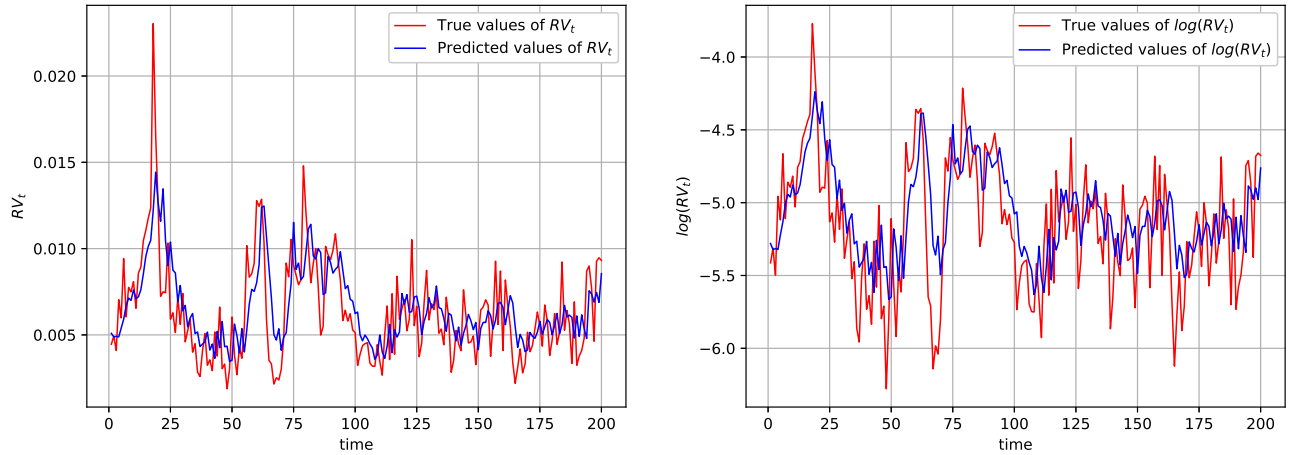


Figure 3: The Random Forest model with the MSE loss, one-step-ahead predictions

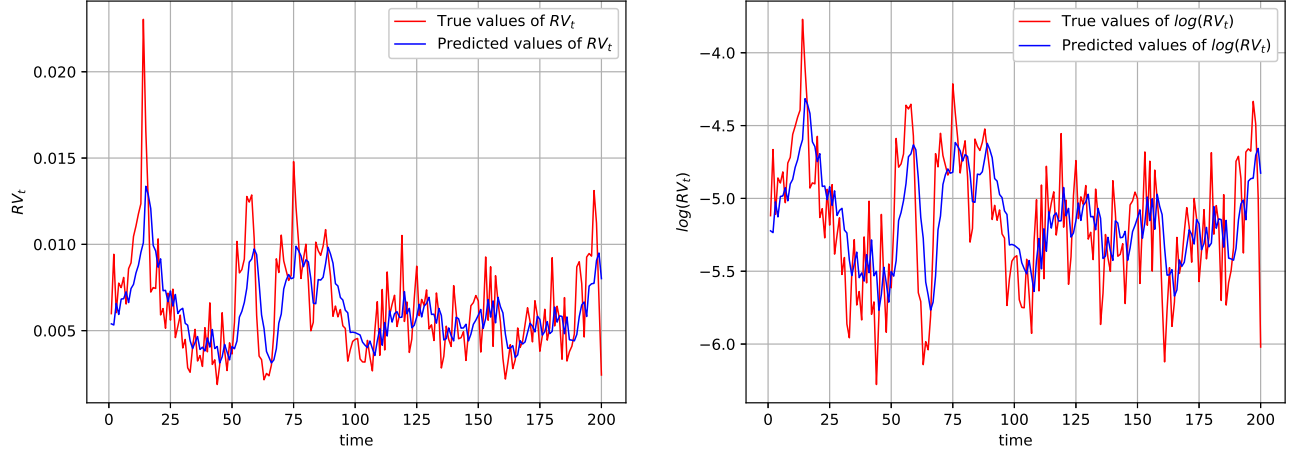


Figure 4: The HAR model with the MSE loss, one-step-ahead predictions

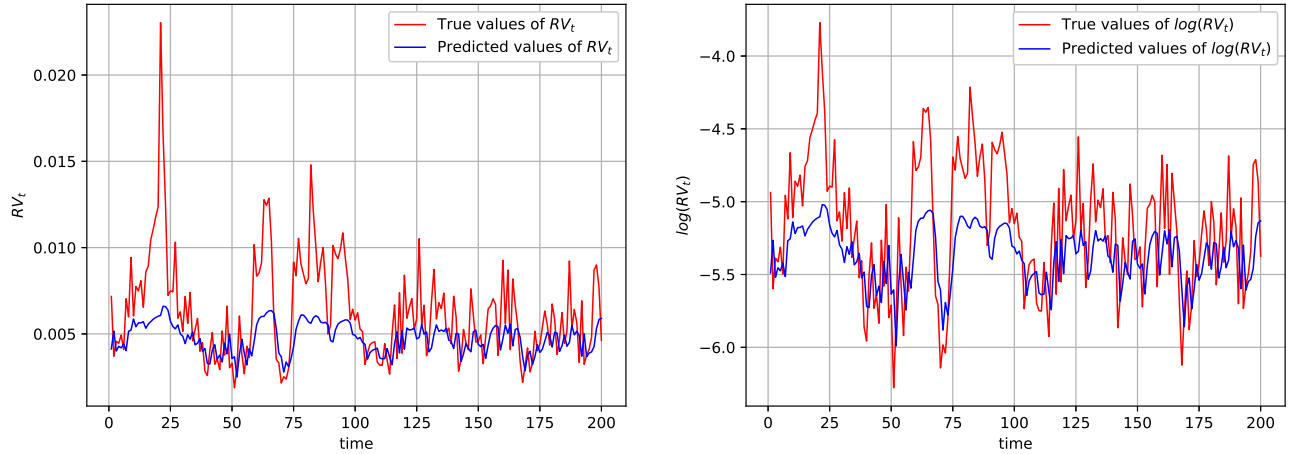


Figure 5: The LSTM model with the MSE loss, one-step-ahead predictions

But we can see that often there is an underestimation of the RV, which is undesirable. Move to asymmetric loss. Compare the HAR and the LSTM models, using the LinEx loss function/criterion and the ALS loss function/criterion (again, we used one-step-ahead predictions):

Model	LinEx for $\log(RV_t)$	ALS for $\log(RV_t)$
HAR	0.014	0.052
LSTM	0.067	0.246

As we can see, again the LSTM model predicts worse than the HAR model, even with an asymmetric criteria. On Figures 6–7 one-step-ahead predictions (on the first 200 days of out-of-sample data) of these two models (estimated using asymmetric loss functions) are represented.

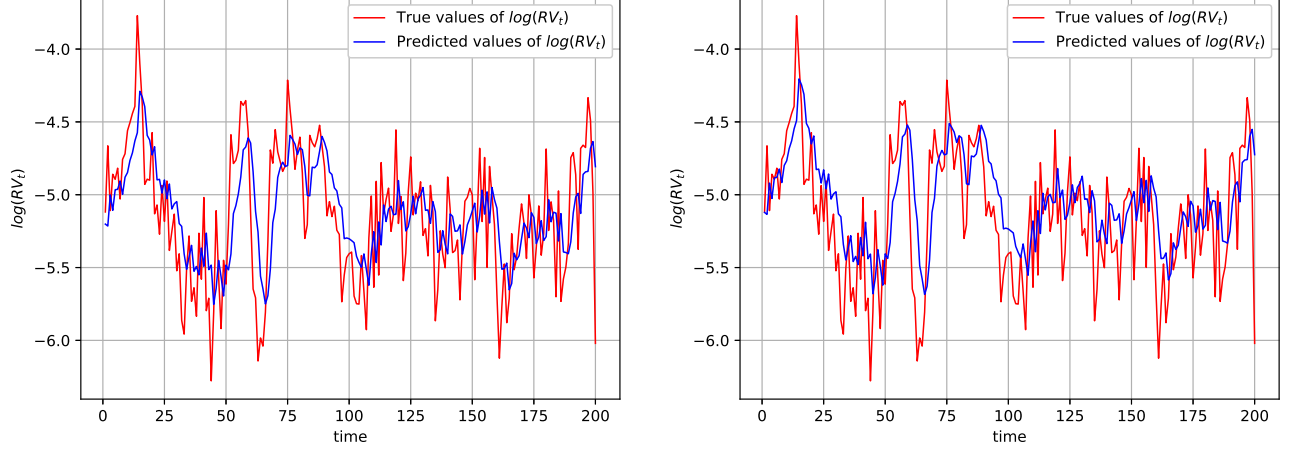


Figure 6: The HAR model with the LinEx loss (left) and the ALS loss (right), one-step-ahead predictions

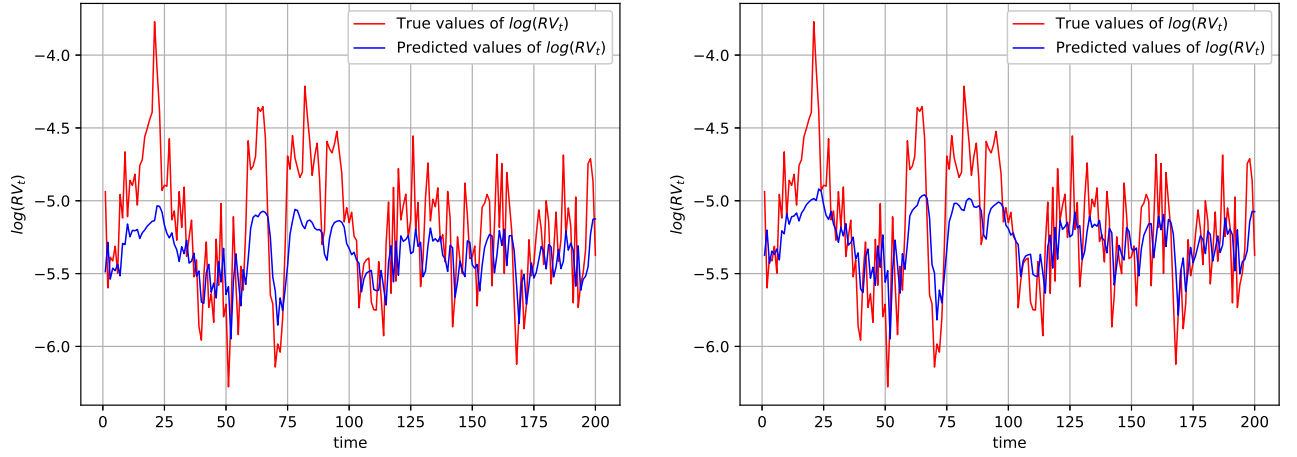


Figure 7: The LSTM model with the LinEx loss (left) and the ALS loss (right), one-step-ahead predictions

We can see on the plots that the predictions of the models estimated with asymmetric loss functions are slightly higher (in general) than with symmetric. This is because we assumed that

underprediction is less desirable than overprediction. If we want to highlight the asymmetry more, we should increase α (in both asymmetric losses).

Note that such poor results of the LSTM model are predictable since this model (as well as all neural networks in general) need much more data points than we have (more than 100,000 points is acceptable, but we have less than 10,000 points).

8 Conclusion

Firstly, we compared the predictive power of four models: the MEM model with QMLE, the HAR model with the MSE loss, the Random Forest model with the MSE loss and the LSTM model with the MSE loss — using the MSE criterion for RV_t and their logs. The result was that the MEM model was the best by margin. So, the considered machine learning algorithms were not able to outperform the MEM model, which is an econometric approach.

Secondly, we compared the predictive power of two models: the HAR model with the LinEx loss and ALS loss, and the LSTM model with the same losses — using criteria consistent with the loss functions of the models. The result was that the considered machine learning algorithm (the LSTM model) cannot outperform the HAR model with both asymmetric loss functions.

Among the further research plans there is the addition of new criteria and loss functions, new models for comparison and also the addition of jumps into the diffusion equation and the HAR model.

References

- Andersen, T. G. & Bollerslev, T. (1997). Heterogeneous information arrivals and return volatility dynamics: Uncovering the long run in high frequency data. *Journal of Finance*, 52, 975–1005.
- Andersen, T. G., Bollerslev, T., Diebold, F. X., & Labys, P. (2003). Modeling and forecasting realized volatility. *Econometrica*, 71, 529–626.
- Black, F. & Scholes, M. (1973). The Pricing of Options and Corporate Liabilities. *Journal of Political Economy*, 81, 637–54.
- Bollerslev, T. (1986). Generalized Auto Regressive Conditional Heteroskedasticity. *Journal of Econometrics*, 31, 307–27.
- Breiman, L. (2001). Random Forests. *Machine Learning*, 45, 5–32.
- Breiman, L., Friedman, J. H., Olshen, R. A., & Stone, C. J. (1984). *Classification and Regression Trees*. Monterey, CA: Wadsworth and Brooks.

- Bucci, A. (2020). Realized Volatility Forecasting with Neural Networks. *Journal of Financial Econometrics*, 18(3), 502–531.
- Corsi, F. (2003). A Simple Approximate Long-Memory Model of Realized Volatility. *Journal of Financial Econometrics*, 7, 174–96.
- Dokuchaev, N. (2014). Volatility estimation from short time series of stock prices. *Journal of Nonparametric Statistics*, 26, 373–84.
- Donaldson, G. & Kamstra, M. (1996). Forecast Combining with Neural Networks. *Journal of Forecasting*, 15, 49–61.
- Engle, R. F. (1982). Autoregressive conditional heteroskedasticity with estimates of variance of UK inflation. *Econometrica*, 50, 987–1008.
- Engle, R. F. (1990). Discussion: Stock market volatility and the crash of 87. *Review of Financial Studies*, 3, 103–6.
- Engle, R. F. (2002). New frontiers for ARCH models. *Journal of Applied Econometrics*, 17, 425–446.
- Glosten, L. R., Jagannathan, R., & Runkle, D. E. (1993). On the relationship between the expected value and the volatility of the nominal excess return on stocks. *Journal of Finance*, 46, 1779–801.
- Heston, S. L. (1993). A closed-form solution for options with stochastic volatility with applications to bond and currency options. *The Review of Financial Studies*, 6, 327–43.
- Hochreiter, S. (1997). Long Short-Term Memory. *Neural Computation*, 9(8), 1735–1780.
- Khatun, N. & Matin, M. A. (2020). A Study on LINEX Loss Function with Different Estimating Methods. *Open Journal of Statistics*, 10, 52–63.
- Luong, C. & Dokuchaev, N. (2018). Forecasting of Realised Volatility with the Random Forests Algorithm. *Journal of Risk and Financial Management*, 11, 61.
- Nelson, D. B. (1991). Conditional Heteroskedasticity in Asset Returns: A New Approach. *Econometrica*, 59, 347–70.
- Tian, J. (2009). Forecasting the unemployment rate when the forecast loss function is asymmetric.

- Vortelinos, D. I. (2017). Forecasting realized volatility: HAR against Principal Components Combining, neural networks and GARCH. *Research in International Business and Finance*, 39, 824–839.
- Zakoian, J.-M. (1994). Threshold Heteroscedastic Models. *Journal of Economic Dynamics and Control*, 18, 931–55.