Crypto volatility forecasting: ML vs GARCH

Bruno Spilak
Wolfgang Karl Härdle

Ladislaus von Bortkiewicz Chair of Statistics

C.A.S.E.-Center for Applied Statistics and Economics

International Research Training Group

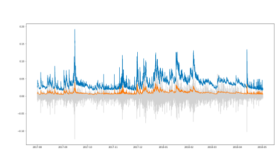
Humboldt-Universität zu Berlin
lvb.wiwi.hu-berlin.de

www.case.hu-berlin.de

irtg1792.hu-berlin.de

Why predicting realized volatility ?

- Trading volatility derivative products (VIX, VDAX, Forex, volatility swaps)
- Build trading strategies with options
- Dynamic risk management
- Crypto market is highly volatile: need for efficient risk management
- Opportunities for new financial products
- ETF on VCRIX ?



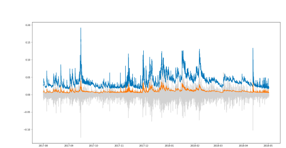
ML vs GARCH

Econometrics

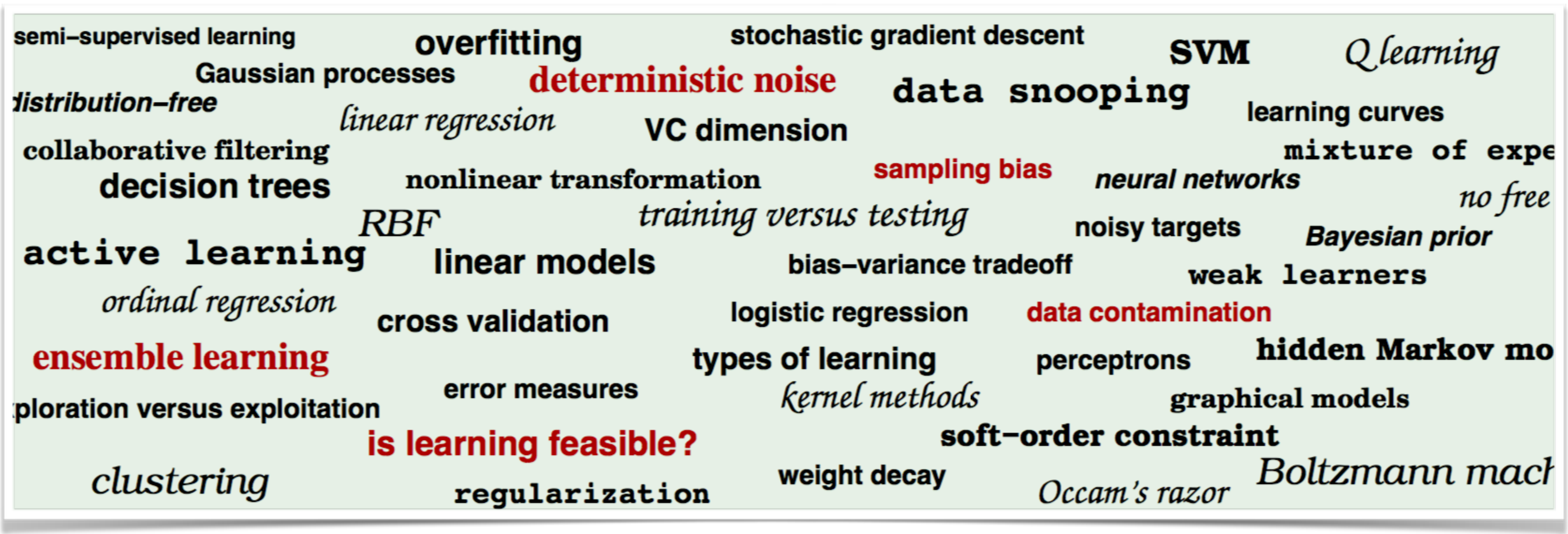
- Observe price
- make assumptions on its dynamics
- Find a formula to price an instrument

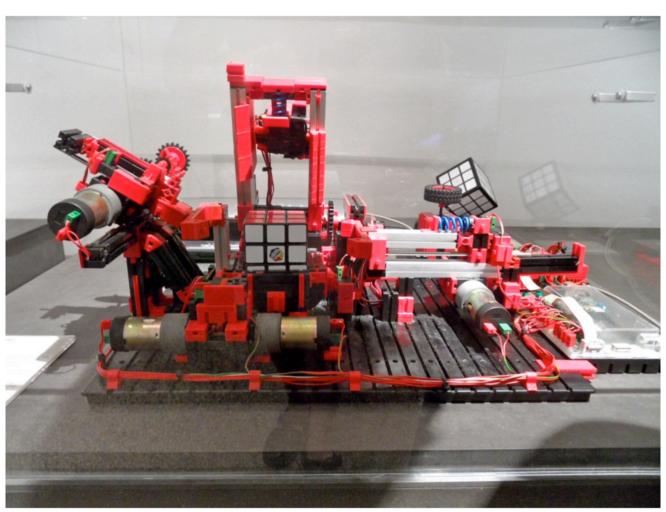
Machine learning

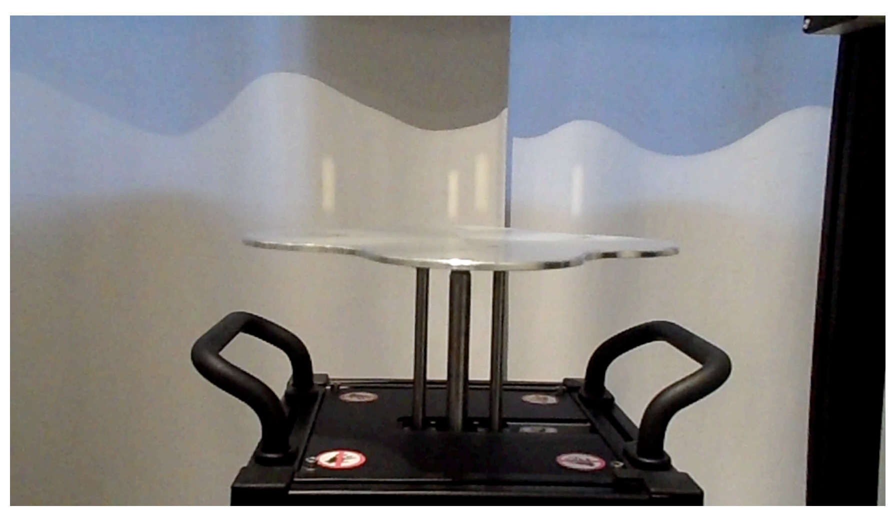
- Observe price
- Feed it into a neural network (kernel machine, random forest)
- We got a “model” !



ML Machine Learning

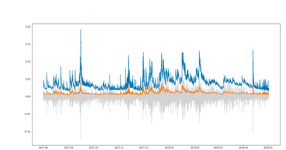






work.caltech.edu/library/181.html

W Phillips (curve), hydro engineer/economist, MONIAC
Monetary National Income Analogue Computer



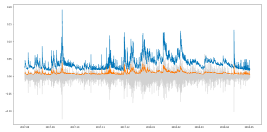
ML vs GARCH

Econometrics

- Strong assumptions
- Structural breaks
- Fat tails, skewness, long memory

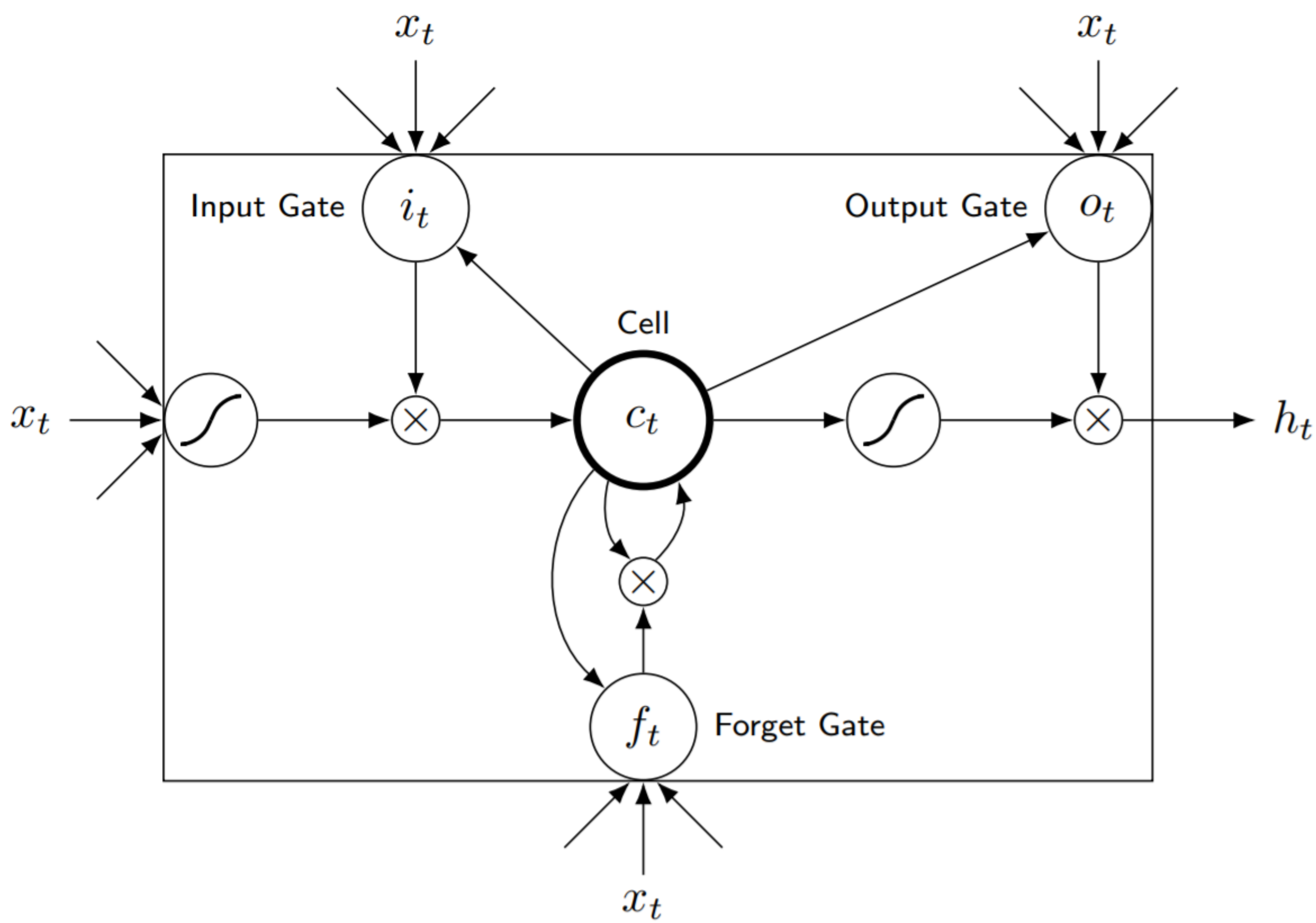
Machine learning

- Not enough data
- Imbalance class problems
- Blackbox

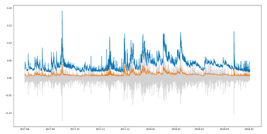


ML vs GARCH

- Predictive accuracy
- Robustness
- Computation
- Flexibility
- Interpretability



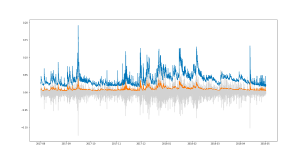
How to use econometrics to make the black box transparent ?



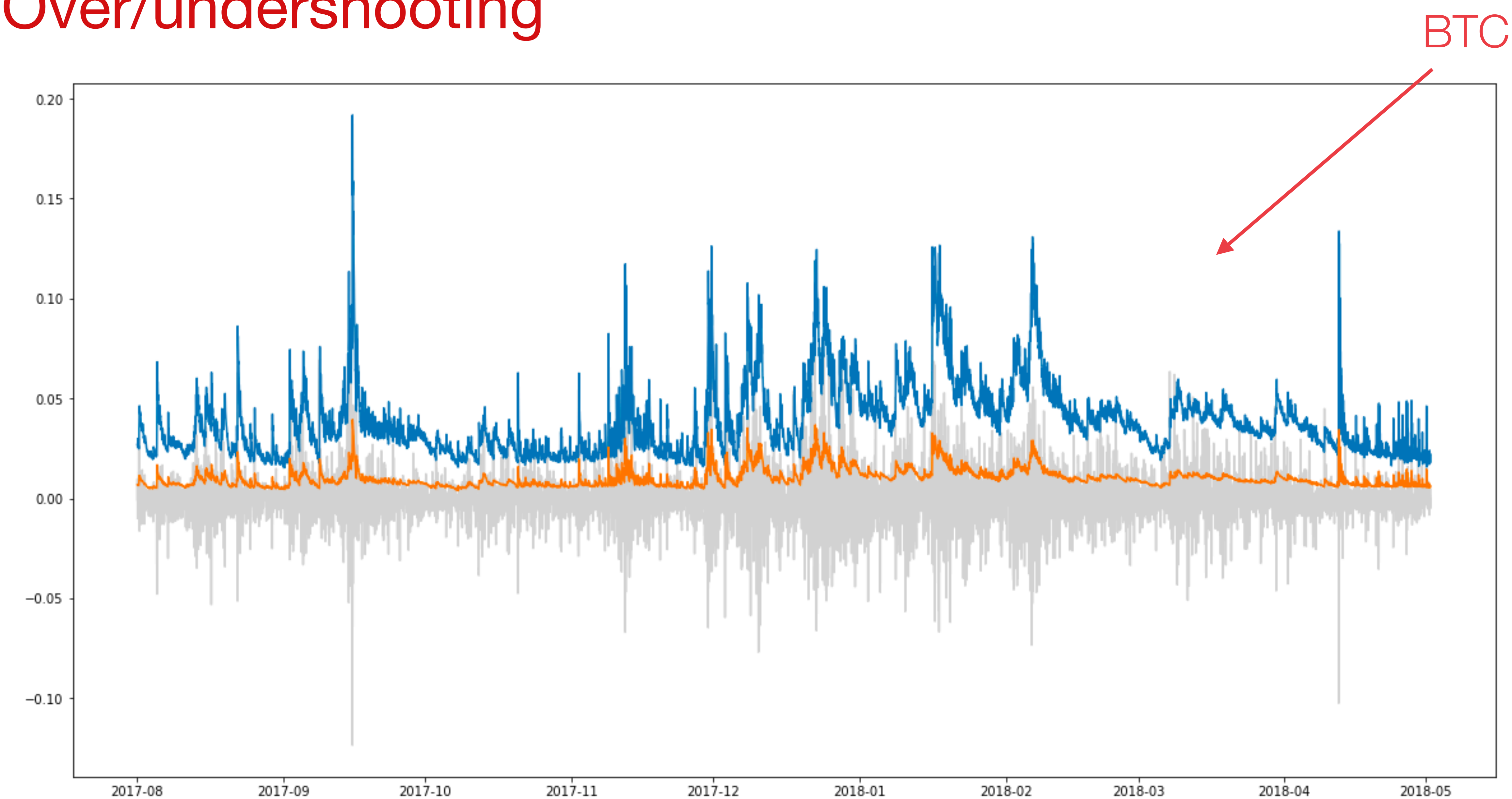
Risk management

- Prediction of future extreme loss (X_t) rather than overshooting or undershooting
- Build metric for undershooting evaluation
- Build metric for overshooting evaluation
- Overshooting of $\widehat{\mathrm{VaR}}_t\,\mathrm{GARCH}$ forecast
- Undershooting of $\widehat{\mathrm{VaR}}_t$ forecast as historical VaR_t

How to use ML and ETRIX on top of simple strategies in order to build a well calibrated risk management?

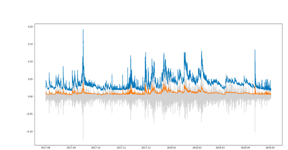


Over/undershooting



Undershooting and **Overshooting** risk managers for the **loss**

If we could predict all exceedances over the undershooting risk manager, we would have a perfect strategy



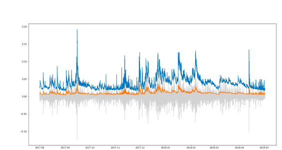
Tail Loss for risk management

- Invest when tail estimator ($\widehat{\mathrm{VaR}}_t$) is small
- De-invest when $\widehat{\mathrm{VaR}}_t$ is large
- P_t, position size at time *t*, (capital invested in risky asset):

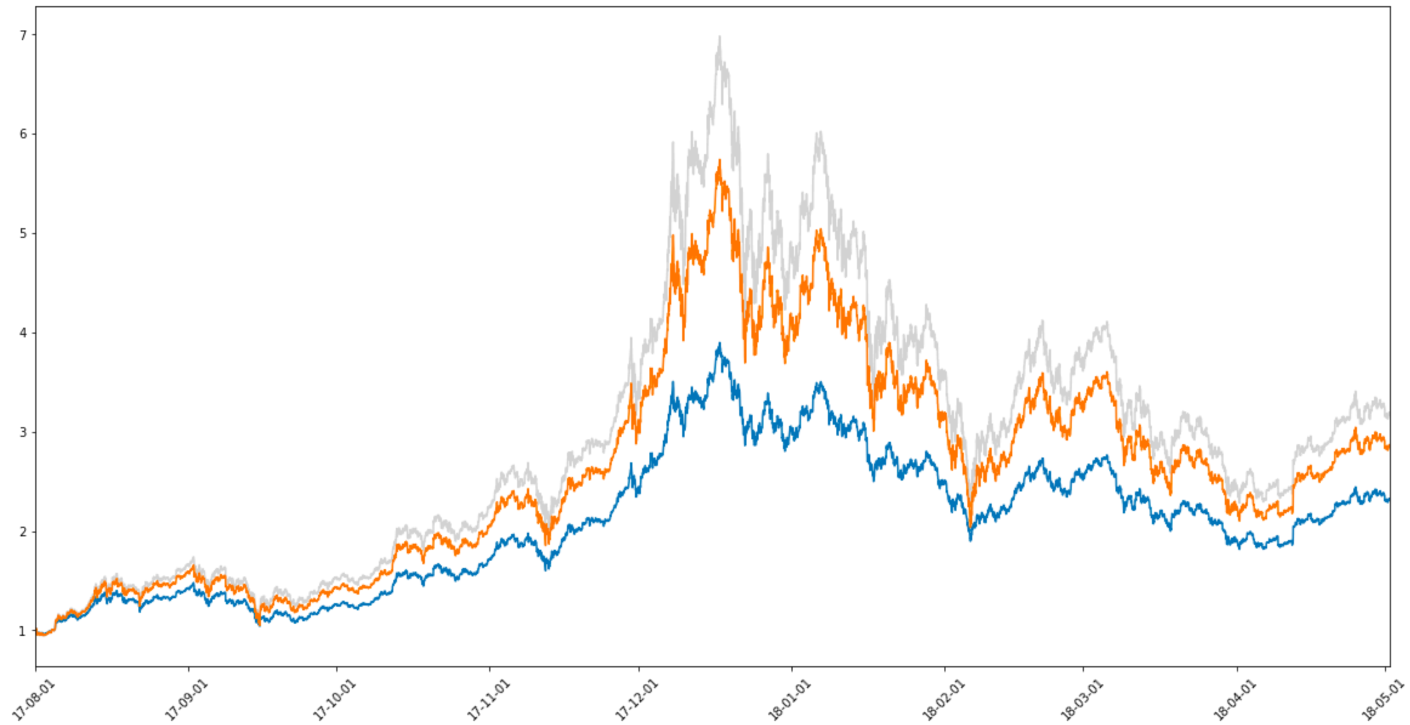
$$\mathrm{P}_t = k / \widehat{\mathrm{VaR}}_t^{\,p}$$

Where *k* is the budgeted risk (predefined) per trade and *p* is used to penalise extreme losses (for now *p=1*)

Goal: reduce drawdowns without losing trading opportunities

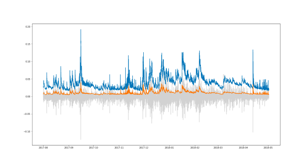


Tail Loss for risk management



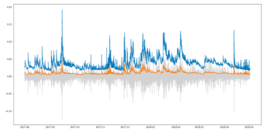
Metrics	Undershooting	Overshooting
Average position size	0.91	0.69
Average gains	0.008	0.006
Average loss	0.008	0.006
Max gain	0.109	0.076
Max loss	0.079	0.059

Undershooting and **Overshooting** risk managers for the **btc return** with position sizing 20170801 - 20180501



Outline

1. ETRIX
2. ML
3. ML vs GARCH
4. Results for two risk management strategies



Presentation of models

- ARIMA(p, d, q) :

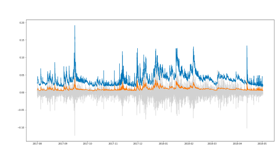
$$\Delta y_t = a_1 \Delta y_{t-1} + a_2 \Delta y_{t-2} + \ldots + a_p \Delta y_{t-p}$$
$$+ \varepsilon_t + b_1 \varepsilon_{t-1} + b_2 \varepsilon_{t-2} + \ldots + b_q \varepsilon_{t-q}$$

Where $\Delta y_t = y_t - y_{t-1}$ is the differenced series and $\varepsilon_t \sim \mathrm{N}\left(0, \sigma^2\right)$

- GARCH(p, q) :

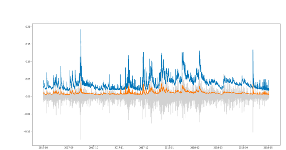
$$\varepsilon_t = Z_t \sigma_t$$
$$Z_t \sim N(0,1)$$
$$\sigma_t^2 = \omega + \sum_{i=1}^{p} \beta_i \sigma_{t-i}^2 + \sum_{j=1}^{q} \alpha_j \varepsilon_{t-j}^2$$

Where $\omega > 0, \quad \alpha_i \geq 0, \quad \beta_i \geq 0; \quad \sum_{i=1}^{p} \beta_i + \sum_{j=1}^{q} \alpha_j < 1$



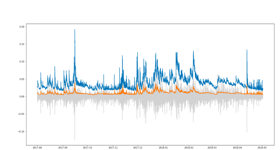
Extreme value theory (EVT) for ETRIX

- GARCH captures time-varying volatility behaviour
- GARCH innovation (Z_t) heavy tails
- Need to take into consideration extreme tail events



Extreme value theory (EVT) for ETRIX

- $\mathrm{GARCH} - \mathrm{EVT}(p, q)$ approach (N. Packham et al (2016))
- Fit simple GARCH on loss (negative return) $r_t = Z_t \sigma_t$ via Quasi Maximum Likelihood Estimation (QMLE)
- Get volatility forecast $\hat{\sigma}_t$ and residuals $\varepsilon_t = r_t / \hat{\sigma}_t$
- Define threshold u corresponding to a certain quantile of loss
- Fit ε_t, where $\varepsilon_t \geq u$ to new distribution: Generalized Pareto distribution (GPD)

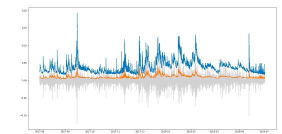


Generalized Pareto Distribution (GPD)

$$G_{\xi,\beta}(x) = \begin{cases} 1 - (1 + \xi x/\beta)^{-1/\xi}, & \xi \neq 0 \\ 1 - \exp^{-x/\beta}, & \xi = 0 \end{cases}$$

where $\beta > 0$, $x \geq 0$, when $\xi \geq 0$ and $0 < x \leq -\beta/\xi$, when $\xi < 0$

- Describes *max domain of attraction* McNeil et al., 2005
- Pareto distribution is heavy-tailed, exponential distribution is light-tailed and Pareto type II distribution is short-tailed
- GPD as proxy of excess distribution (Pickands, Balkema, de Haan Theorem)



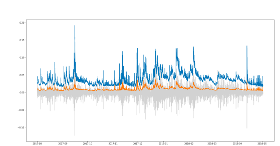
ETRIX for Risk Management

- Fit GARCH model to data
- Fit GARCH innovations to various distributions (normal, GPD)
- Build mean $\hat{\mu}_{t+1}$ and volatility $\hat{\sigma}_{t+1}$ forecast from estimated GARCH
- Forecast $\widehat{\mathrm{VaR}}_t^{(q)} = \mathrm{VaR}_t^{(q)}(X_{t+1}) = \hat{\mu}_{t+1} + \hat{\sigma}_{t+1} \cdot \mathrm{VaR}_t^{(q)}(Z)$

where

- $\mathrm{VaR}_t^q(Z) = F^{-1}(q)$ where F is the distribution function of Z
- For ex: if $Z \sim GPD(u, \sigma, \xi)$,

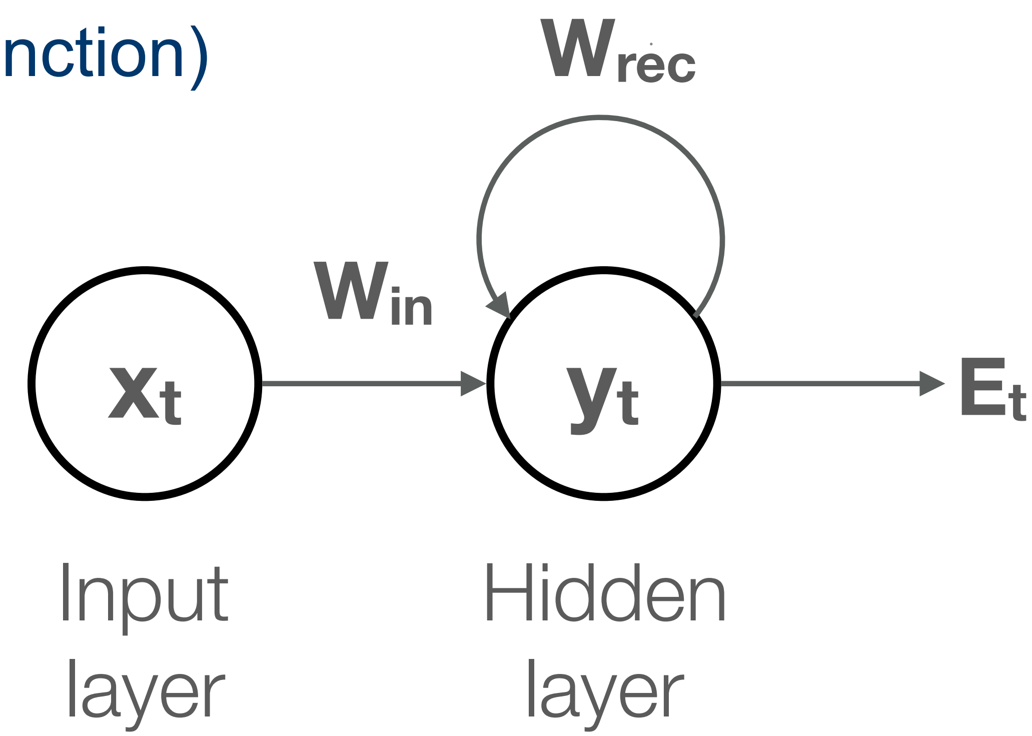
$$\mathrm{VaR}_t^q(Z) = u + \sigma/\xi \left[\left((1-q)/(\zeta_u)\right)^{-\xi} - 1 \right]$$



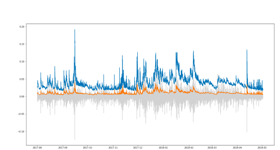
Recurrent neural network

$$y_t = f_\theta(X_t)$$

- Here f_θ is a neural network
- hyper parameters (depth, width, activation function)
- Estimate the weights with BPTT

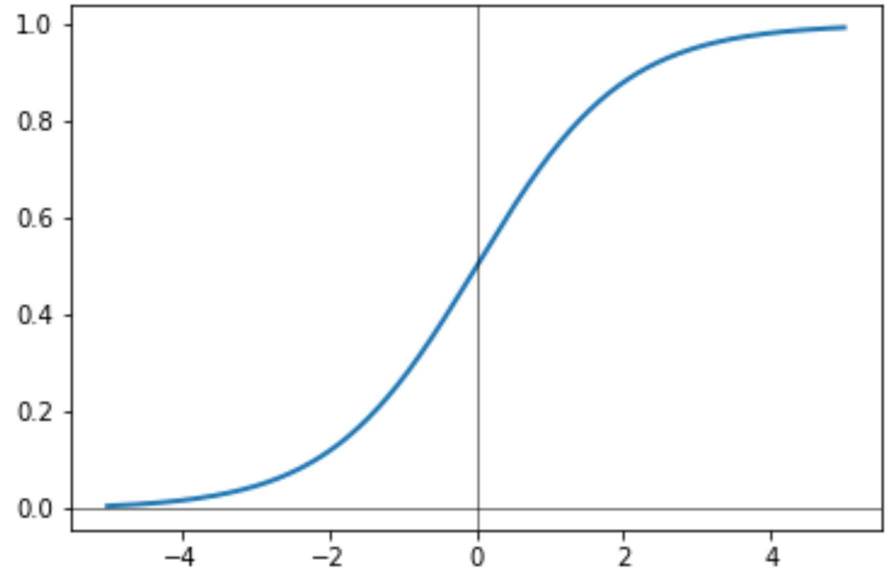


$$y_t = f_\theta(y_{t-1}, x_t) = W_{rec}\sigma(x_{t-1}) + W_{in}x_t + b,$$
$$\textit{where } \sigma \textit{ is the sigmoid activation function}$$

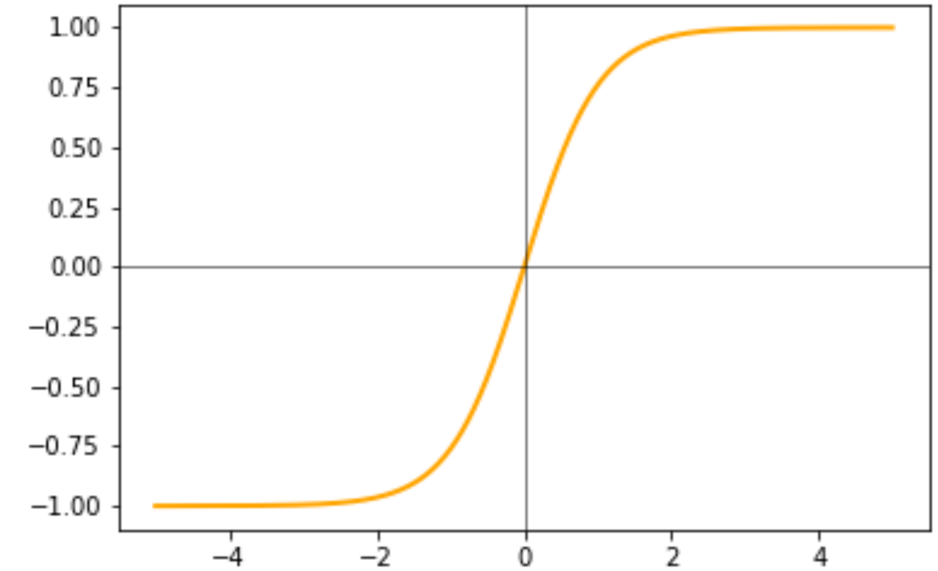


Activation functions

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

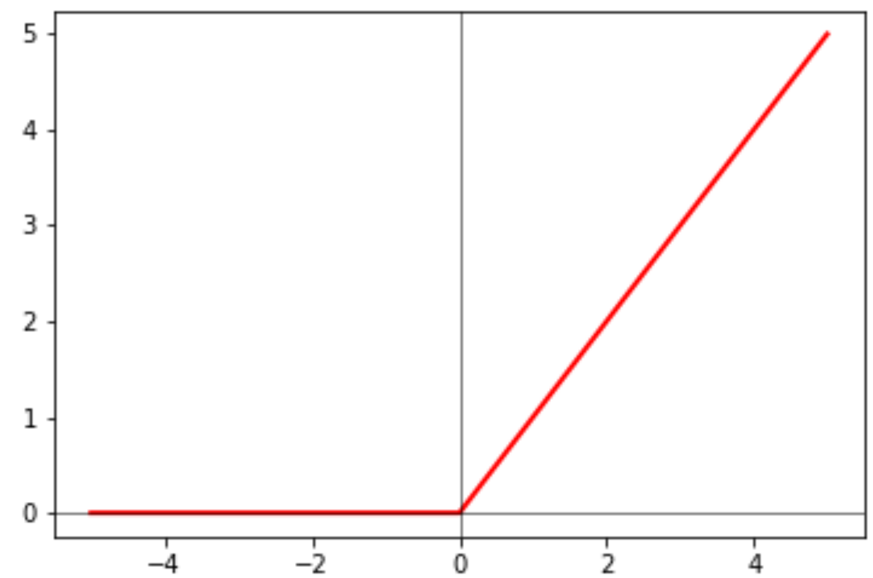


$$\tanh(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$$

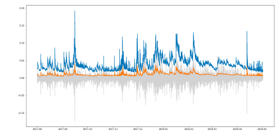


softmax

hinge



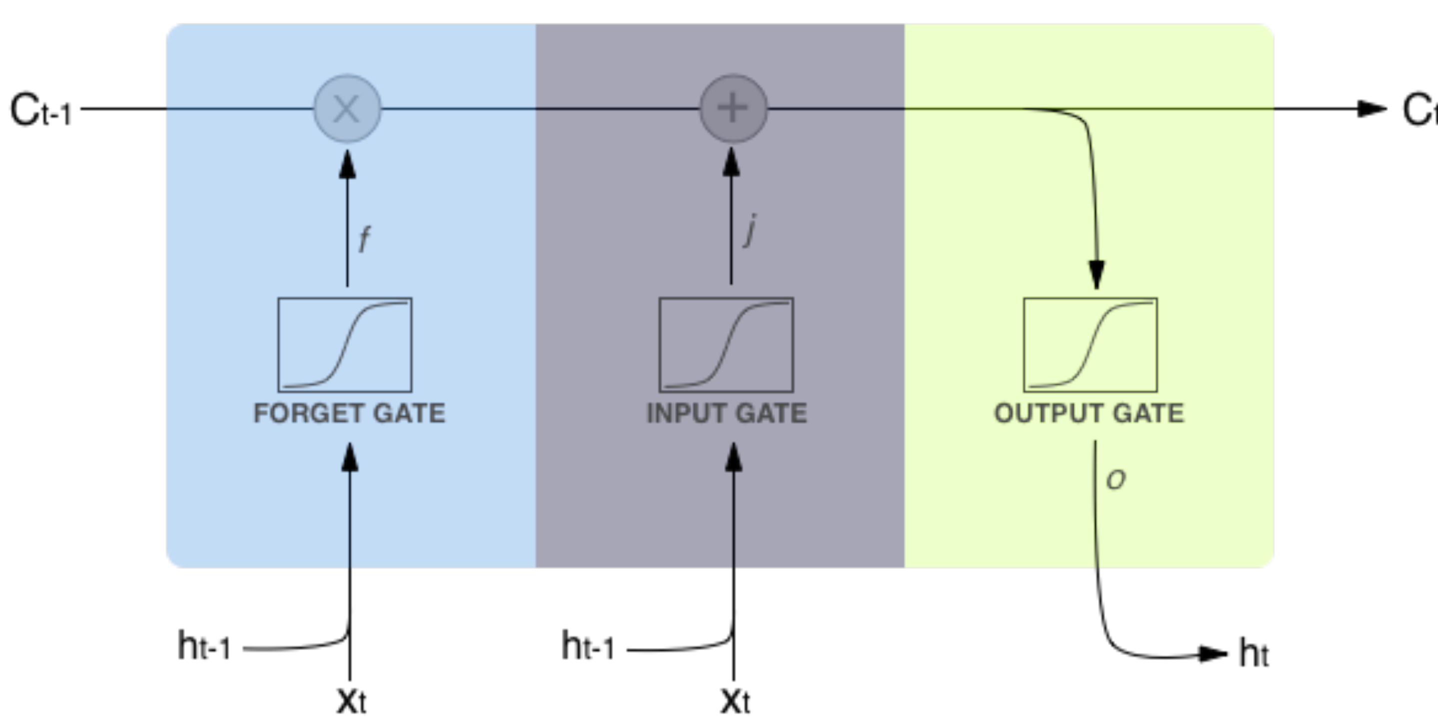
$$ReLU(x) = \max(0,x)$$

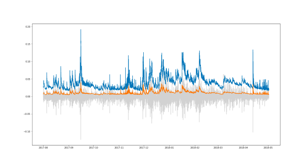


LSTM memory block

- Self-connected memory LSTM cells: superset of RNN
- Hidden units can see their previous output
- Sequential memory
- Long term dependencies
- Three multiplicative units: input, output, forget gates (write, read, reset)

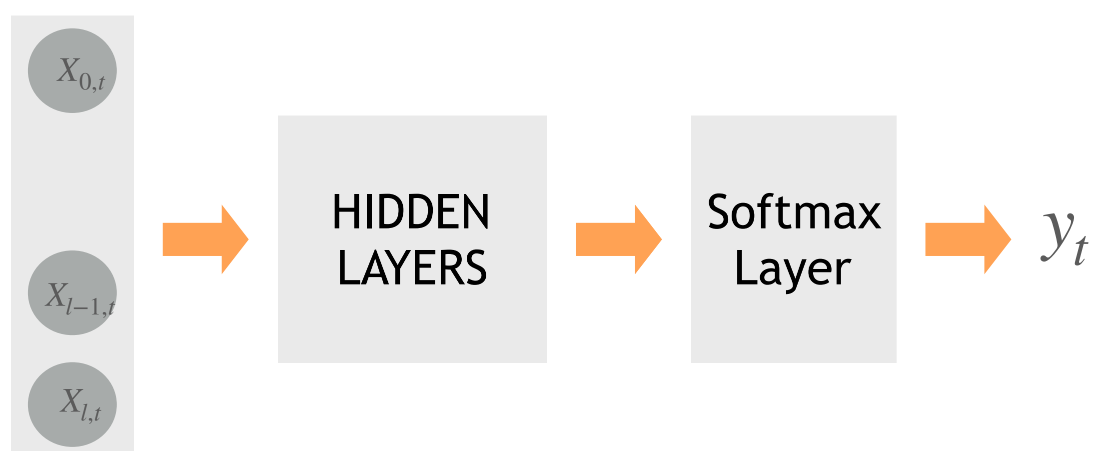
LSTM Memory Cell

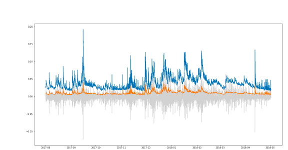




Specific task for deep learning

- Build training data $\{(X_1, y_1), \ldots, (X_n, y_n)\}$
- Input: X_t for a given window size *l*: $X_t = (\frac{p_{t-l+1}}{p_{t-l}}, \ldots, \frac{p_{t+1}}{p_t})$
- y_t depends on risk management strategy





$\widehat{\mathrm{histVaR}}_t^{(0.1)}$ undershoots risk

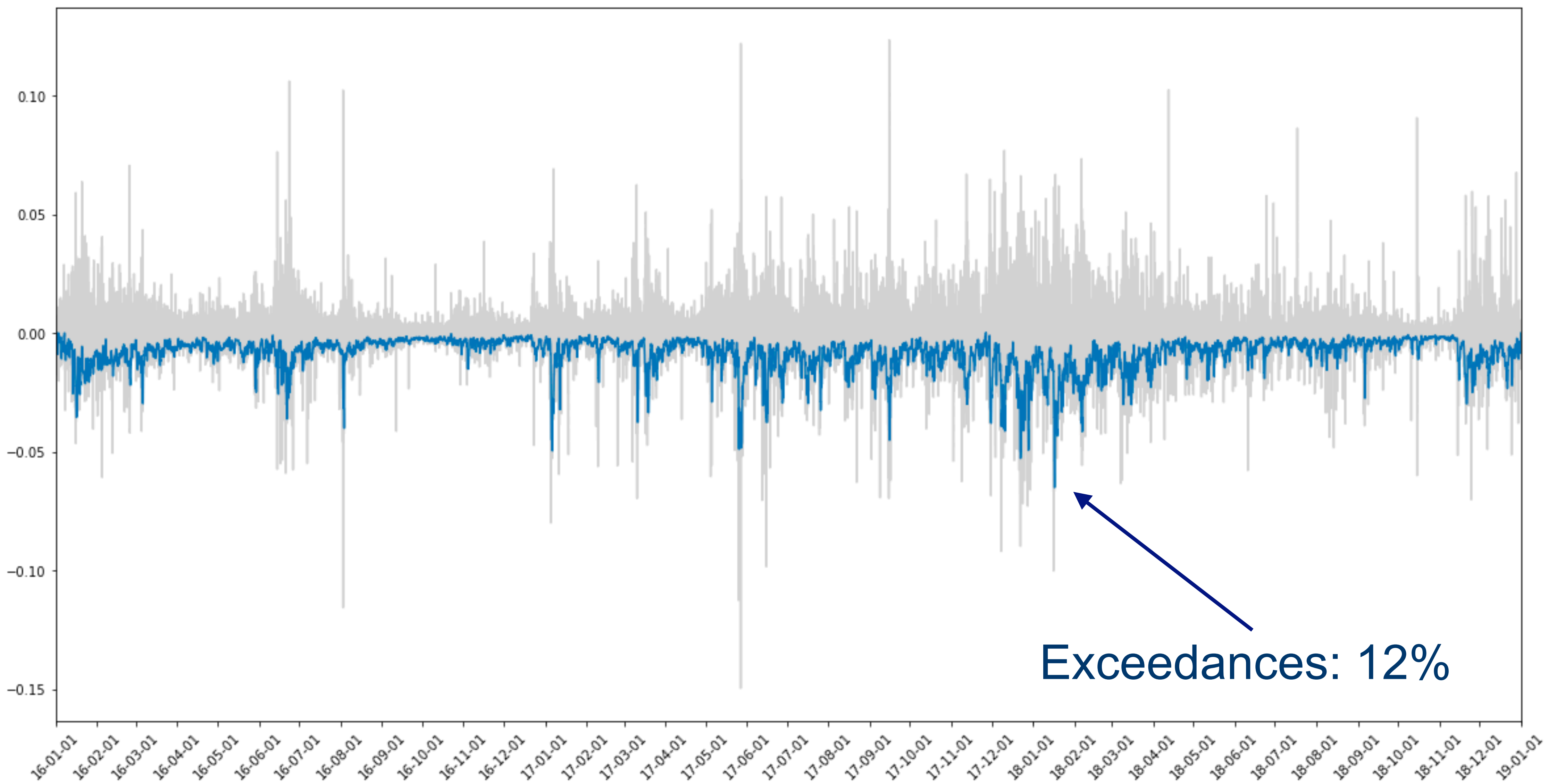
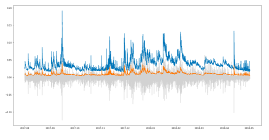


Figure: $\widehat{\mathrm{histVaR}}_t^{(0.1)}$ hourly forecast and btc **returns**



Dynamic volatility forecast for ML VaR calibration

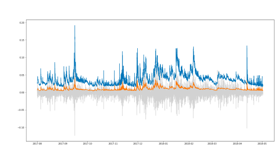
- Include future information from training set to build target variable
- Target variable:

$$y_t = \begin{cases} 0, & \text{if } \widehat{\mathrm{histVaR}}_t^{q} \le r_{t+1} \le \widehat{\mathrm{histVaR}}_t^{1-q}, \\ 1, & \text{if } r_{t+1} \ge \widehat{\mathrm{histVaR}}_t^{1-q}, \\ 2, & \text{if } r_{t+1} \le \widehat{\mathrm{histVaR}}_t^{q} \end{cases}$$

- Define J_t^w as:

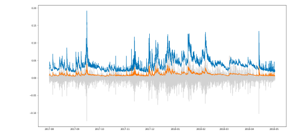
$$J_t^w = \begin{cases} 0, & \text{if } y_t = 0 \text{ or } y_t = 1 \\ 1 & \text{if } y_t = 2 \end{cases}$$

Can we accurately predict $\widehat{\mathrm{histVaR}}_t^{q}$ exceedances ?



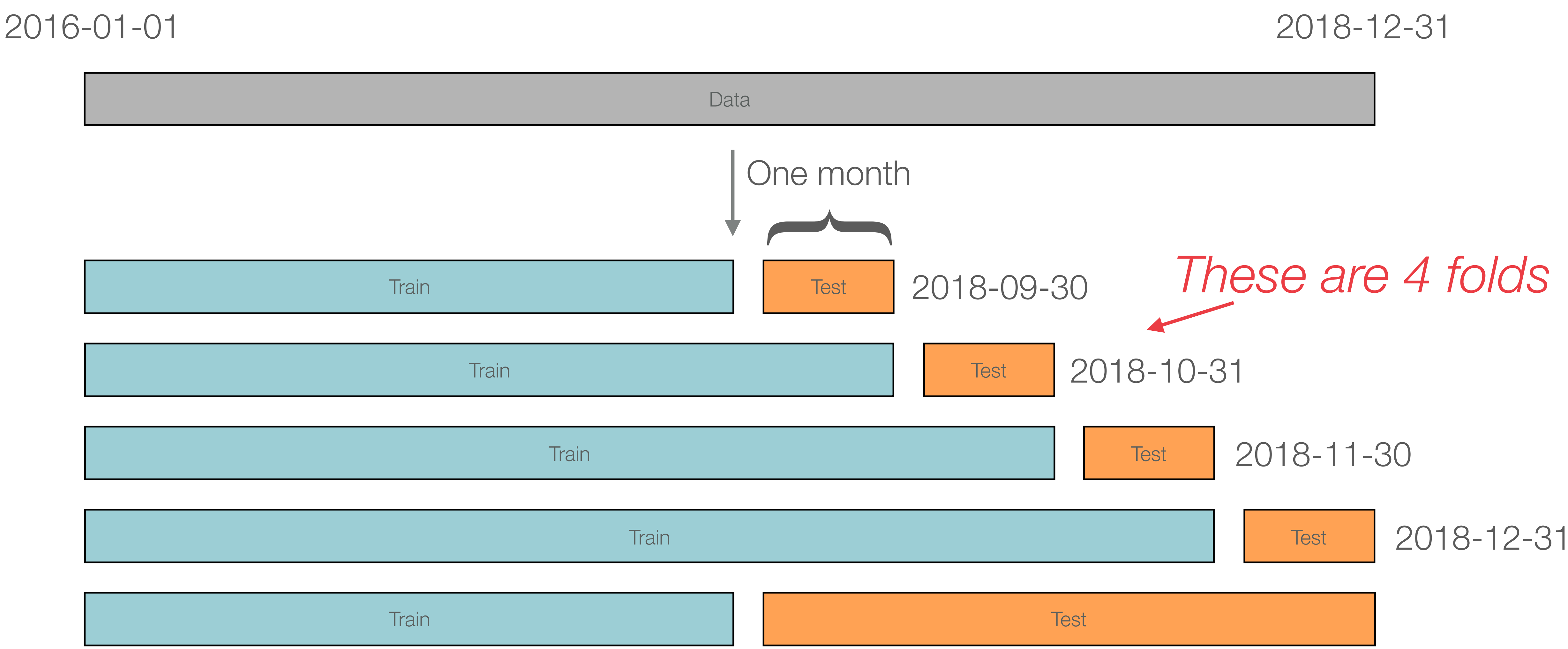
NN Training

- Loss function: cross-entropy
- Highly imbalanced class by definition (through threshold *q*)
- To make training more efficient: weighted loss



Hyperparameter tuning

- 10-fold crossvalidation
- 1M moving window
- Robust evaluation of model



GARCH VaR calibration backtest measure

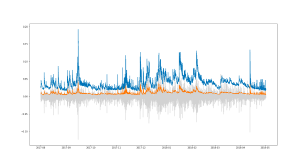
- Build VaR forecast, de-investment in period of high $\widehat{\mathrm{VaR}}_t^{0.1}$
- $\widehat{\mathrm{VaR}}_t$ violation (exceedance):

$$\Psi_t^{(1)} = \mathrm{I}_t(r_t \leq \widehat{\mathrm{VaR}}_{t-1}^{0.1})$$

- Define $J_t^w(\mathrm{GARCH})$ as:

$$J_t^w(\mathrm{GARCH}) = \begin{cases} 0, & \text{if } \widehat{\mathrm{VaR}}_t^{0.1} \leq \widehat{\mathrm{histVaR}}_t^{0.1} \leq r_{t+1} \\ 1 & \text{if } r_{t+1} \leq \widehat{\mathrm{histVaR}}_t^{0.1} \leq \widehat{\mathrm{VaR}}_t^{0.1} \end{cases}$$

Is $\widehat{\mathrm{VaR}}_t^{0.1}$ a better estimator than ML for $\widehat{\mathrm{histVaR}}_t^{0.1}$ exceedances ?



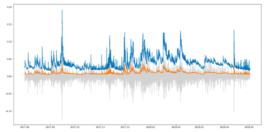
ML VaR calibration backtest measure

- Class 2 is the tail event of interest
- Compare corresponding One vs All confusion matrix by grouping other classes
- Type I error: wrongly classified as tail event (false positive: **Overshooting**)
- Type II error: wrongly classified as normal event (false negative: **Undershooting**)
- Type II error out-of-the-blue event (N. Packham et al (2016))

ML prediction

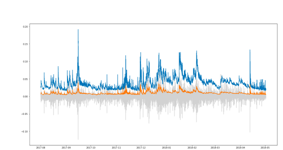
Confusion matrix $\left(J^{w}, \widehat{J}^{w}\right) = \mathrm{CM}_{\mathrm{w}}^{\mathrm{ml}}$

$\widehat{\mathrm{histVaR}}_t^{0.1}$ violation: $\mathrm{FN}^{ml} = \mathrm{CM}_{\mathrm{w}}^{\mathrm{ml}}[2,1]$



Metrics for undershooting evaluation: type II errors

- $\widehat{\mathrm{VaR}}_t^{0.1}$ calibration, average exceedances: $\Psi^{(1)} = 1/T \sum_{t=1}^{T} \Psi_t^{(1)}$
- Correspond to $\mathrm{FNR} = 1/T \,.\, \mathrm{FN}^{ml}$(false negative) for the ML case
- Both metrics must be smaller than for good calibration of tail events for the level $q = 0.1$



Metrics for overshooting evaluation: type I errors

Overshooting

- With ML minimising type II error is very easy (predict positive class all the time)

- If we predict a drop ($\widehat{\mathrm{histVaR}}_t^{0.1} \geq \widehat{\mathrm{VaR}}_t^{0.1}$ or $\hat{J}_t^w = 1$), set position to 0, otherwise apply tail loss with $\widehat{\mathrm{histVaR}}_t^{0.1}$

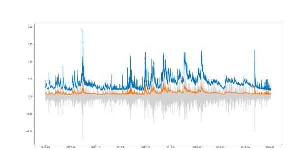
- If type I error is high, we will miss trading opportunities

Compare missed opportunities between models

Data

- Intraday data: 1h close price of Bitcoin (BTC)
- 20160101 to 20181231 (26305 observations)
- train (20160101/20180930)/validation (20181001/20181231)
- Keep the rest for later out-of-sample test
- Retrain every day

price (USD) Market Cap
20190121 - 20190128
coinmarketcap.com



Data

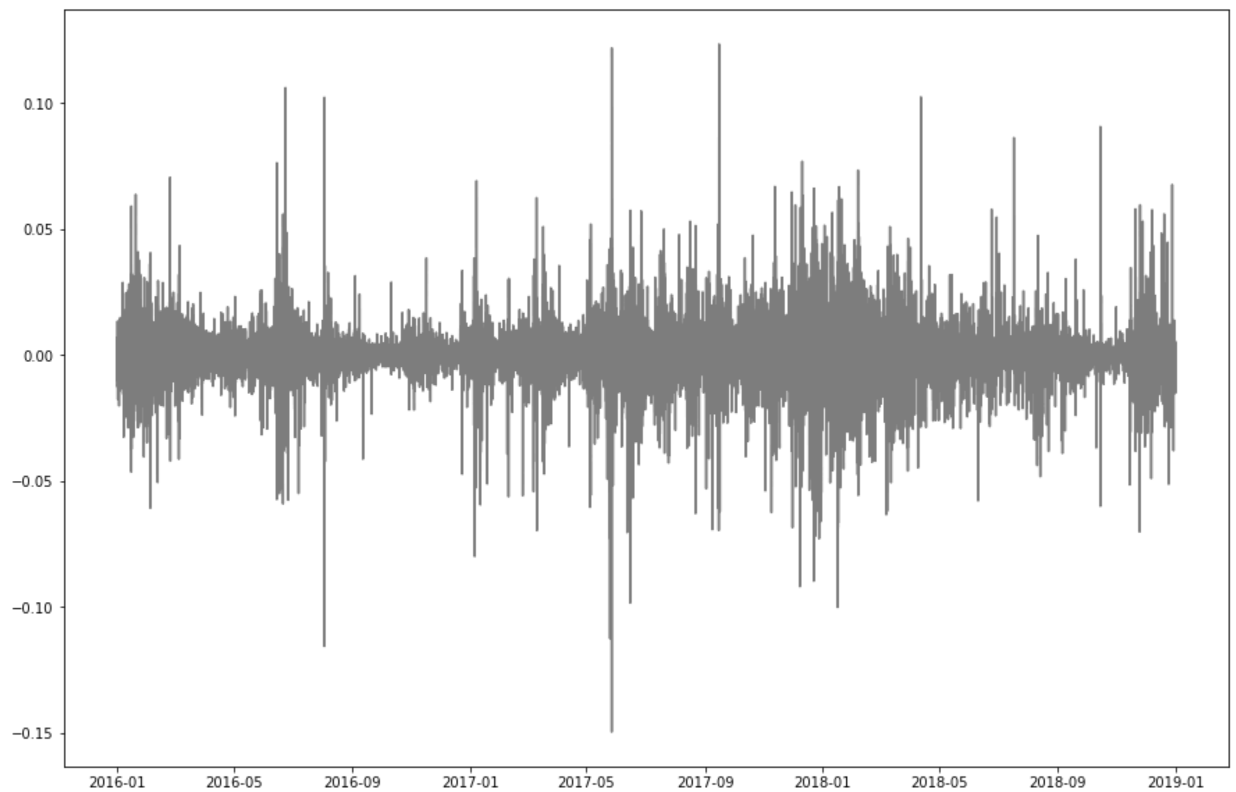


Figure: BTC log returns

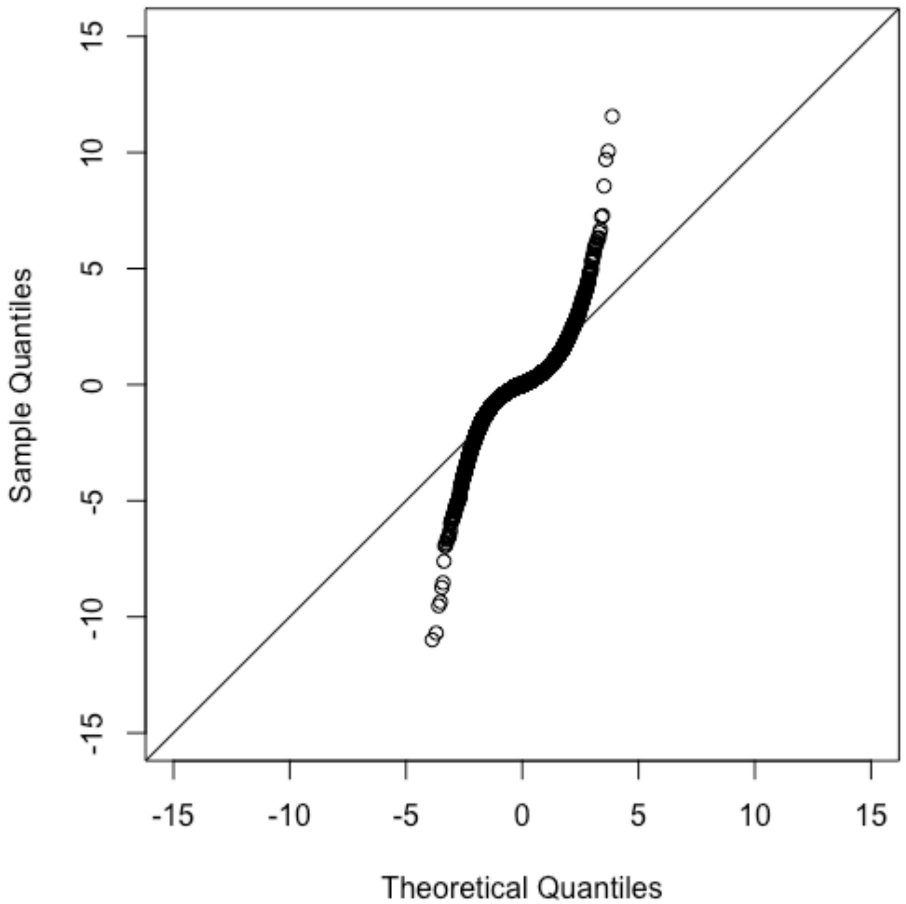
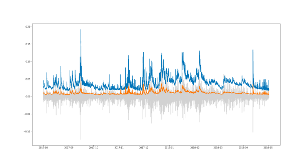
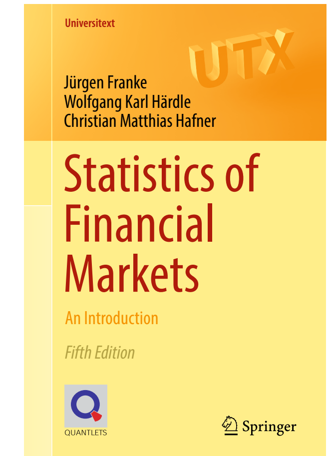


Figure: QQ-plot of BTC log returns

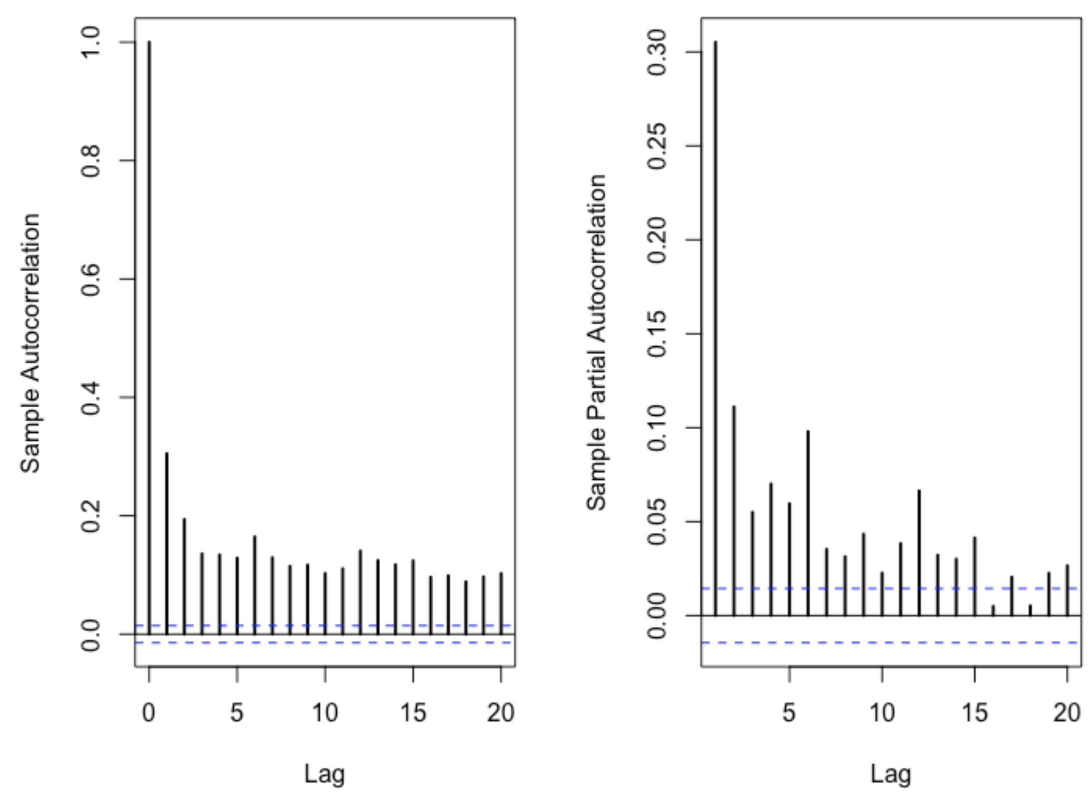
MLvsGARCHecon



ARIMA

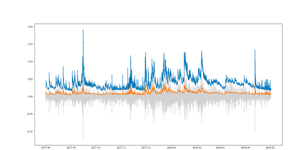


- Classical methodology, Franke et al (2019)
- Chen et al (2017) A first econometric analysis of the CRIX family
- Box-Jenkins method to estimate *ARIMA(3,0,1)* with AIC

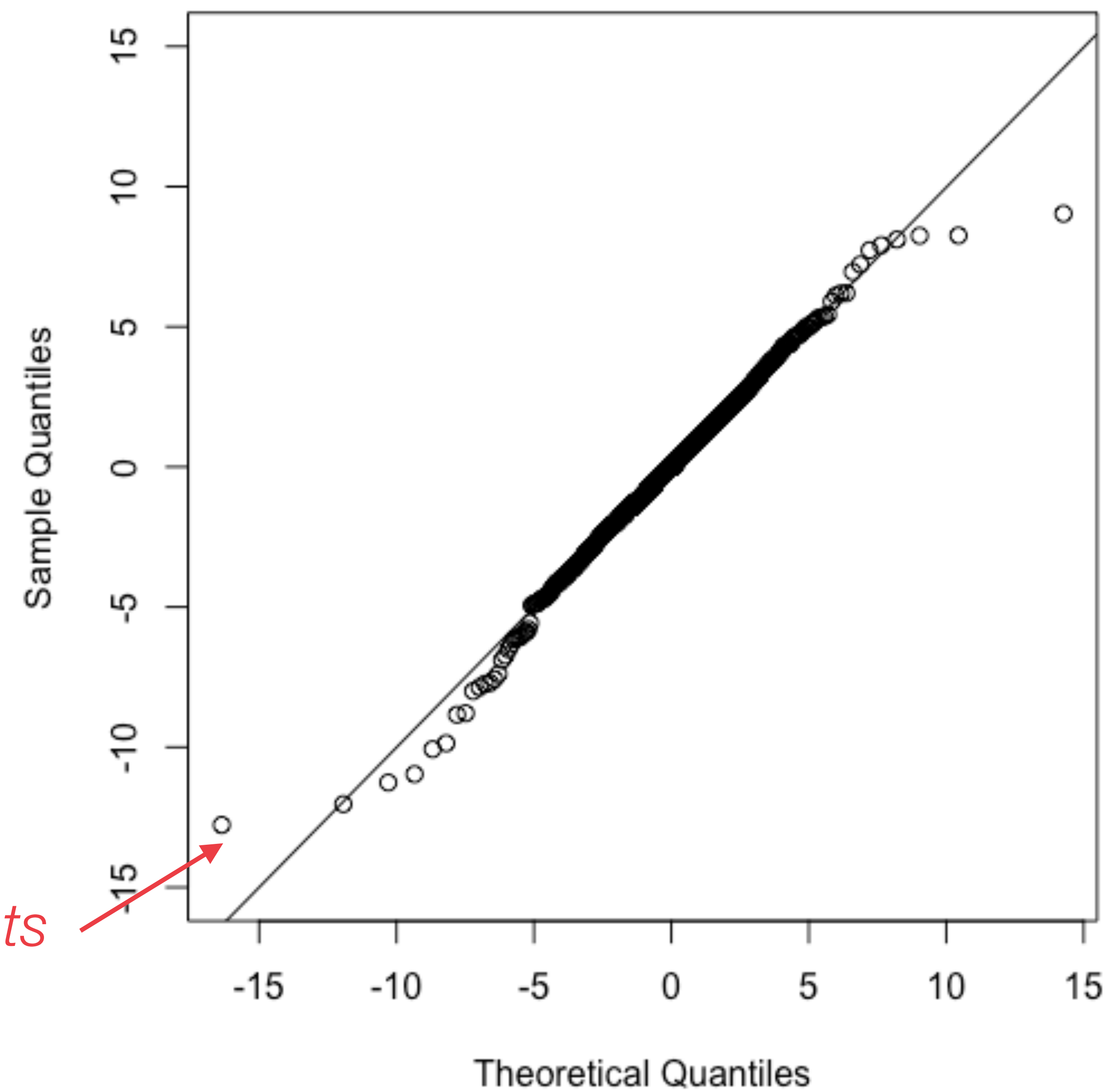


MLvsGARCHecon

Figure: ACF and PACF of squared residuals of *ARIMA(3,0,1)*



Final GARCH(1,2) model



Left tail events

Figure: QQ-plot of residuals of GARCH(1,2)

Final $\mathrm{EVTGARCH}(1,2)$ model

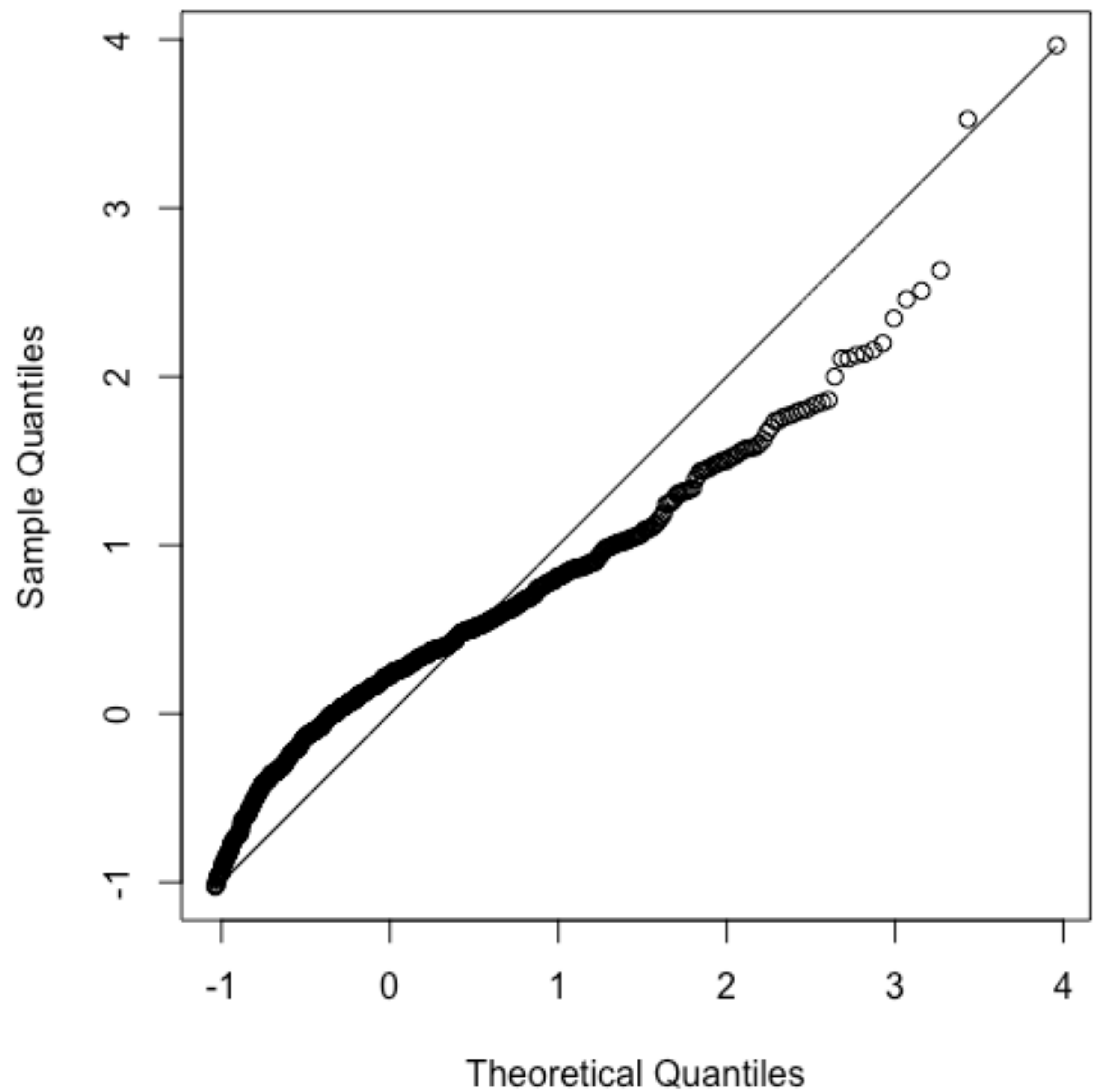
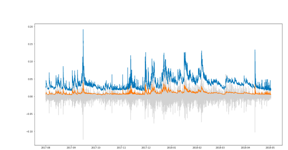


Figure: QQ-plot $\mathrm{GARCH}(1,2)$ residuals for GPD distribution (sample below 10% threshold, u = - 1.04



RESULTS EVT

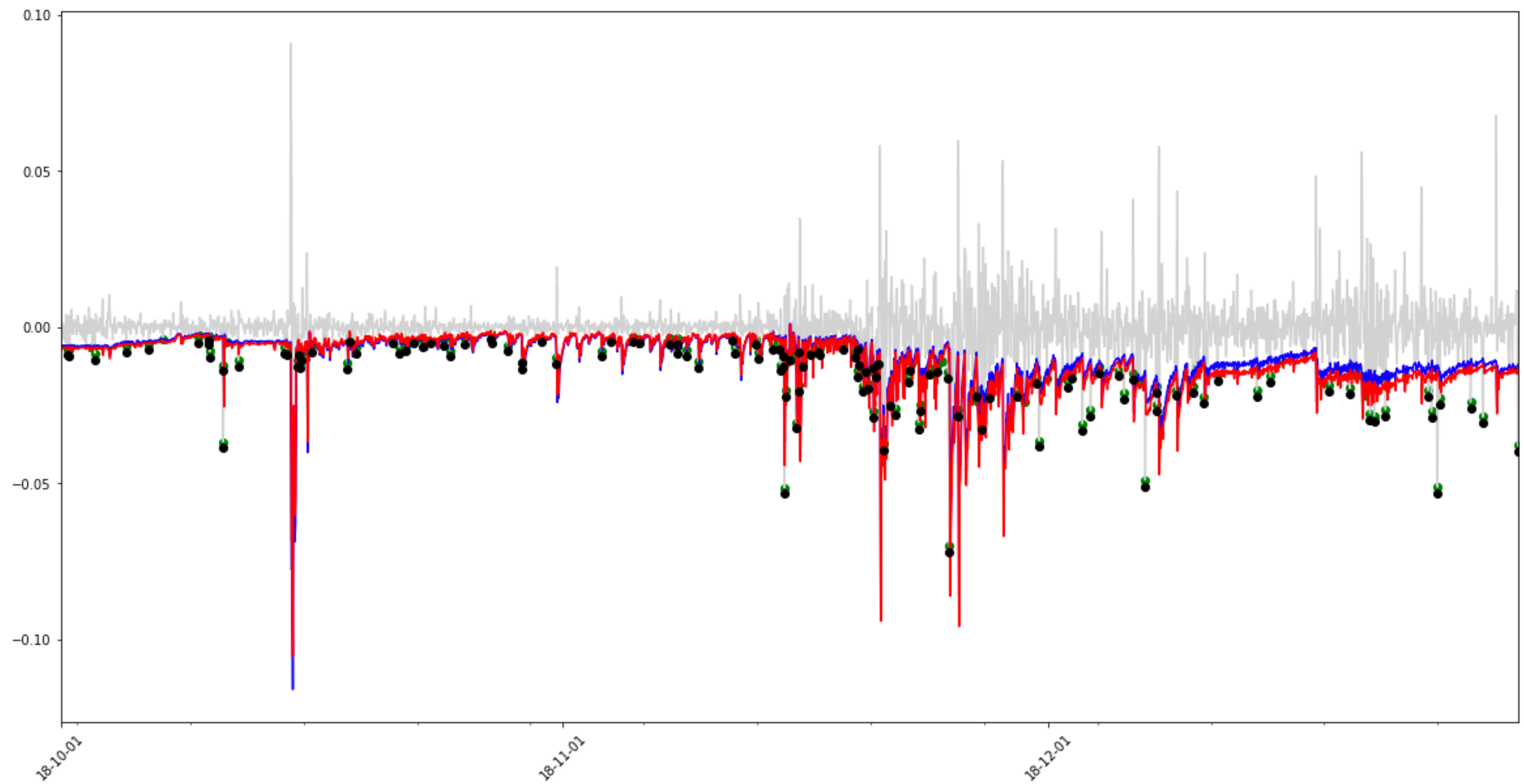
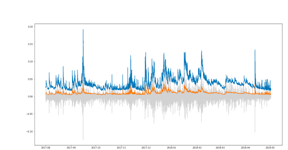


Figure: Rolling (6 months) 10%VaR hourly forecast with **normal** and **GPD** distributions for the innovations

Overshooting ?

VaR GARCH Exceedances: $\Psi^{(1)}_{evt} = 0.07 < 0.1$

VaR EVTGARCH Exceedances: $\Psi^{(1)}_{norm} = 0.06 < 0.1$

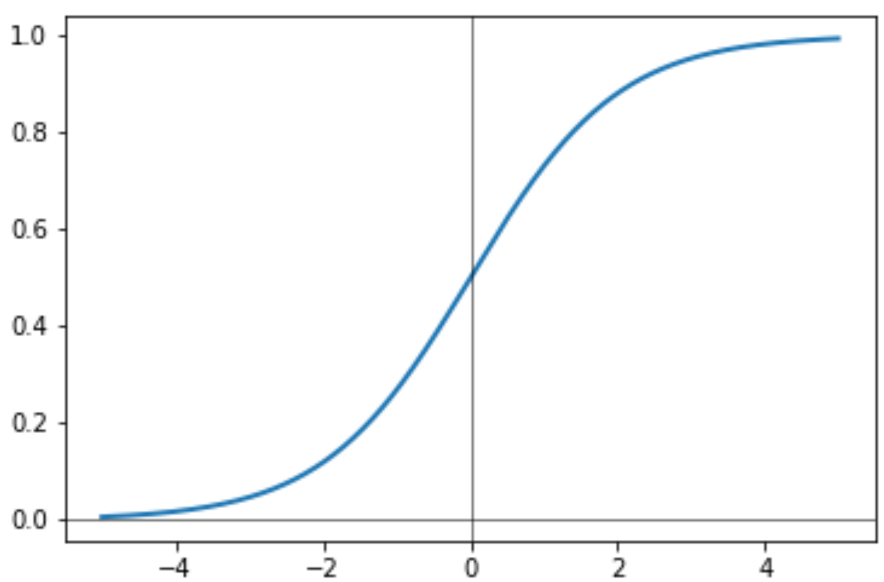


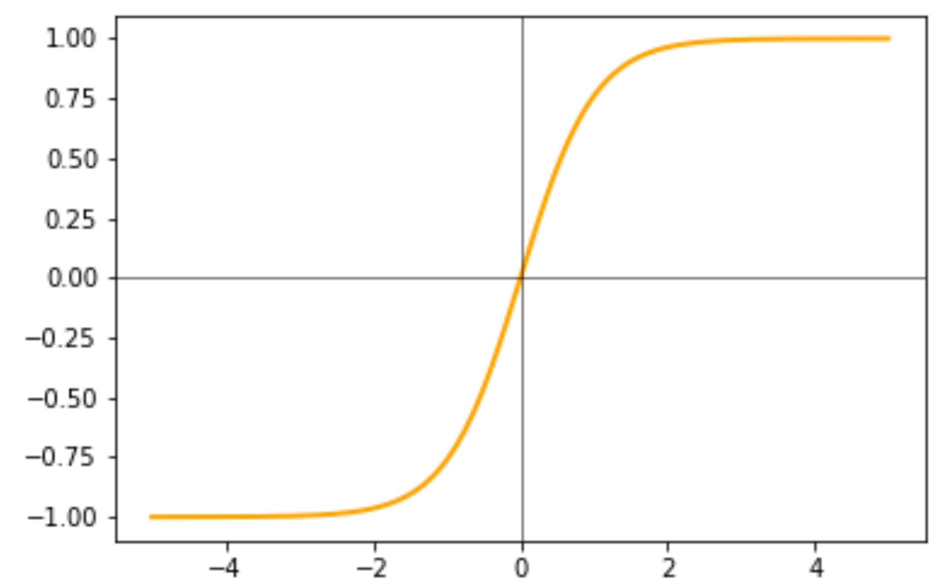
DL Architecture

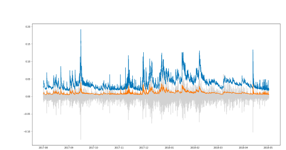
- One LSTM layer with 4 neurons
- One Dense layer with 2 neurons with *tanh* activation function
- One output layer with softmax activation function

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

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







LSTM classification performance

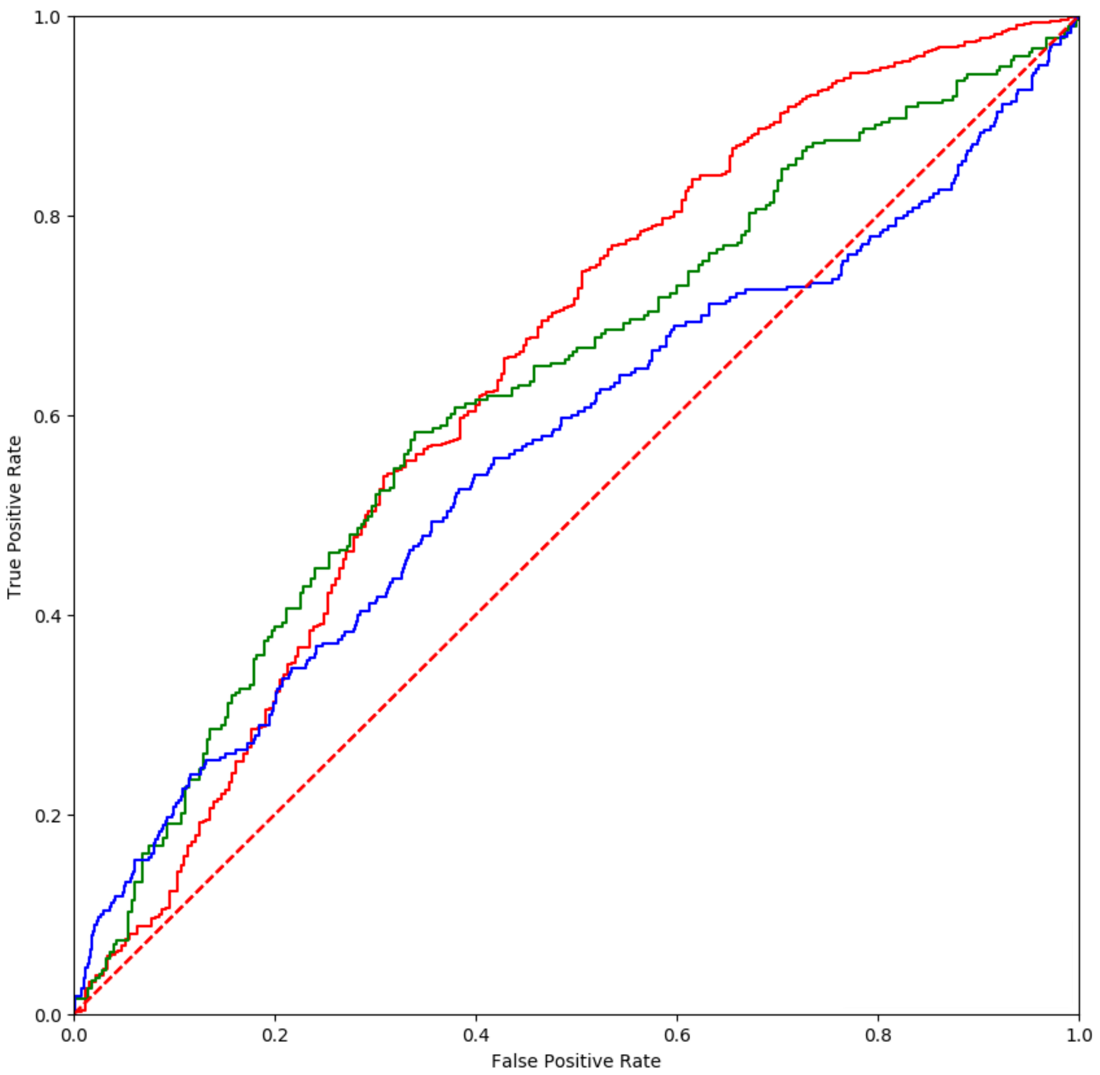


Figure: ROC curve for **class 0 vs 1 (AUC 0.64)**, **class 0 vs 2 (AUC 0.56)**, **class 1 vs 2 (AUC 0.63)**

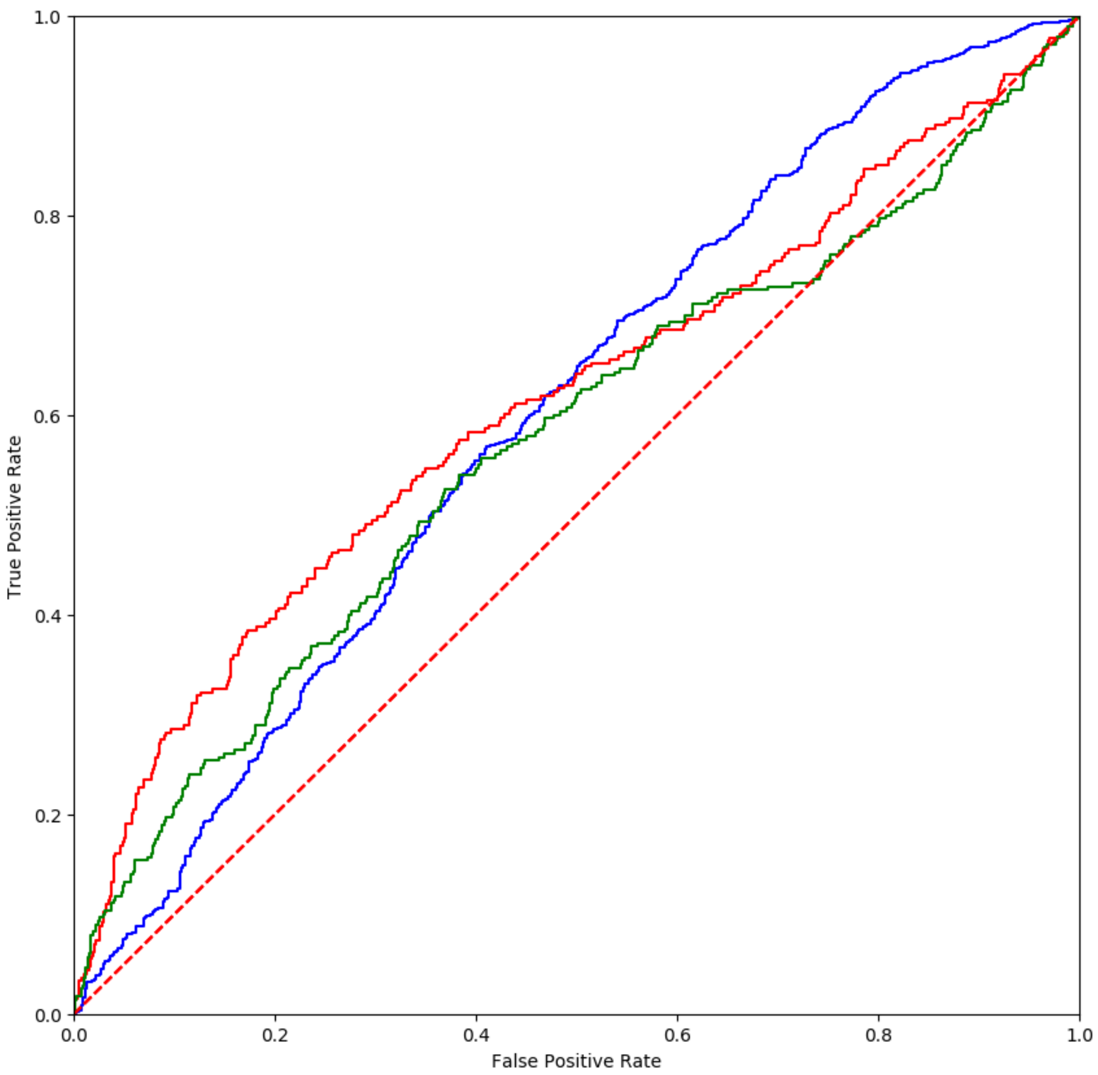
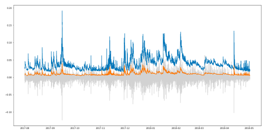


Figure: ROC curve for **class 1 vs (0,2) (AUC 0.60)**, **class 0 vs (1,2) (AUC 0.61)**, **class 2 vs (0,1) (AUC 0.57)**

Better classification for right tail events than left ones



Undershooting

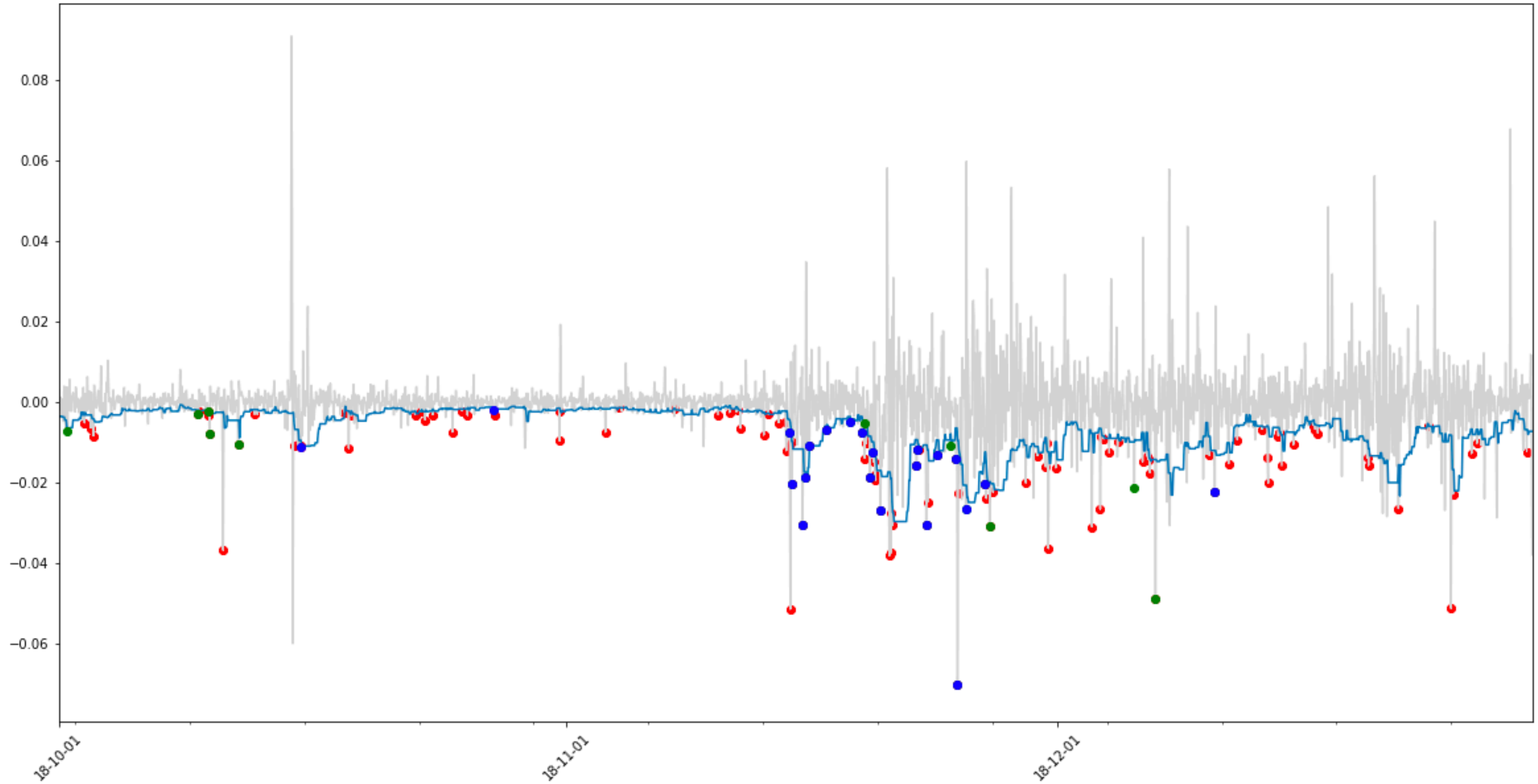


Figure: Exceedances of $\widehat{\mathrm{VaR}}_t^{0.1}$ with **normal** and **GDP** distribution and of **LSTM** compared to $\widehat{\mathrm{histVaR}}_t^{0.1}$

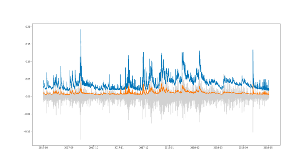
Undershooting correction

Model	Metric	Exceedances (%)
$\widehat{\mathsf{histVaR}}_t^{0.1}$	$\psi^{(1)}$	0.129
VaR_{norm}	J_t^w(GARCH)	0.014
VaR_{evt}	J_t^w(EVTGARCH)	0.010
LSTM	FNR	0.056

Table: Missed drops (exceedance) where $r_{t+1} \leq \widehat{\mathrm{histVaR}}_t^{0.1}$ for different models

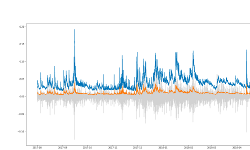
ETRIX is better at predicting drops with $\widehat{\mathrm{VaR}}_t^{0.1}$ than LSTM

ETRIX gives good correction of simple RM based on $\widehat{\mathrm{histVaR}}_t$ for undershooting



Overshooting correction

- Apply corrected tail loss strategy with different models
- Compare strategy return to real return when $r_{t+1} \geq \widehat{\mathrm{VaR}}_t^{0.1}$ for ETRIX or J_t^w = 0 for ML
- We know, $\mathrm{P}_t = 1/(\widehat{\mathrm{VaR}}_t + 1)$, thus we want to have $\widehat{\mathrm{VaR}}_t^p$ close to 0 when we have positive returns

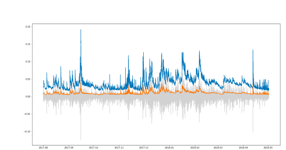


Overshooting correction

- Apply corrected tail loss strategy based on $\widehat{\mathrm{VaR}}_t^{0.1}(\mathrm{GARCH})$ (GARCH-STRAT), $\widehat{\mathrm{VaR}}_t^{0.1}(\mathrm{EVTGARCH})$ (EVTGARCH-STRAT) and $\widehat{J}_t^w$ (ML-STRAT)

- Build corresponding position size at time t, P_t

- Compare $\overline{\mathrm{P}}^{(m)} = 1/T \sum_{t=1}^{T} \mathrm{P}_t^m$ for each model when $r_t \geq 0$



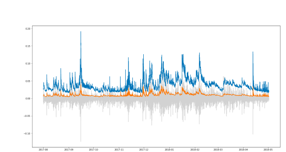
Overshooting correction

Model	$\overline{\text{P}}$ (%)
GARCH	0.16
EVTGARCH	0.12
ML-STRAT	0.53

Table: Average position size for positive return

GARCH overestimate risk in period of positive returns

EVTGARCH is the most conservative model



What is best ?

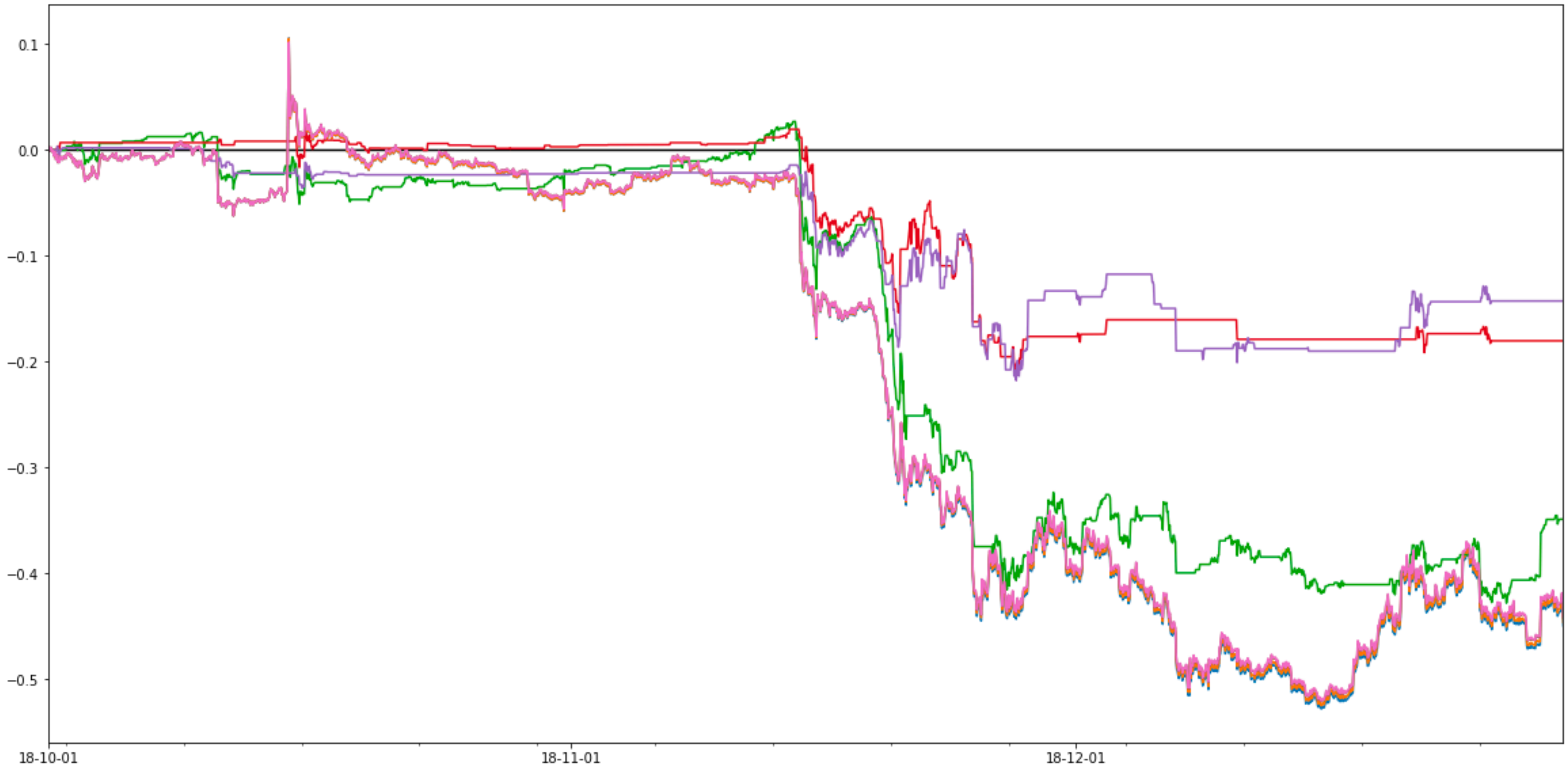
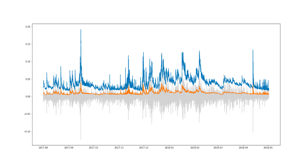
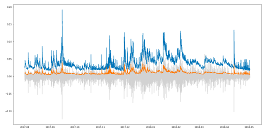


Figure: Corrected tail loss strategy return for GARCH-STRAT, EVTGARCH-STRAT, ML-STRAT compared to original $\widehat{\mathrm{histVaR}}_t^{0.1}$, $\widehat{\mathrm{VaR}}_t^{0.1}(\mathrm{GARCH})$, $\widehat{\mathrm{VaR}}_t^{0.1}(\mathrm{EVTGARCH})$ and btc



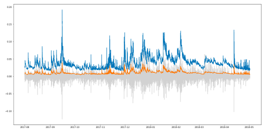
Take home message

- ETRIX outperforms simple ML model at predicting extreme loss for a predefined level (lower type II error): conservative strategy
- ML outperforms ETRIX in terms of overshooting extreme loss for a predefined level (lower type I error): aggressive strategy
- Which model is the best ? Depends on the investor's goal and market condition



Future work

- Compare with CaViaR
- Hyperparameter tuning: ML performance can be greatly improved
- Different horizon forecasts



References

Chen S, Chen CYH, Härdle WK, Lee TM, Ong B (2017) A first econometric analysis of the CRIX family, in Handbook of Blockchain, Digital Finance and Inclusion, Vol 1, Cryptocurrency, FinTech, InsurTech, and Regulation, David LEE Kuo Chuen Robert Deng, eds. ISBN: 9780128104415, Academic Press, Elsevier

Coles S. (2001) An Introduction to Statistical Modelling of Extreme Values, Springer

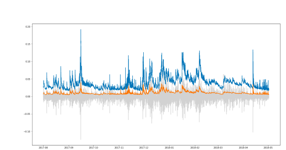
Franke J, Härdle WK, Hafner CM (2019) Statistics of Financial Markets - An introduction, 5th edition, Springer Verlag

Packham N, Papenbrock J, Schwendner P, Woebbeking F (2017), Tail-Risk Protection Trading Strategies, Quantitative Finance, 17 (5), 729-744, https://doi.org/10.1080/14697688.2016.1249512

Kjellson, B. (2013), Forecasting Expected Shortfall: An Extreme Value Approach, ISSN 1654-6229, https://github.com/BenjaK/Thesis2013

Bee, M., Trapin, L. (2018), Estimating and Forecasting Conditional Risk Measures with Extreme Value Theory: A Review, Risks, 6 (2), 1-16

Wong, Z.Y., Chin W.C., Tan S.H. (2016) Daily value-at-risk modelling and forecast evaluation: The realised volatility approach, The Journal of Finance and Data Science



Crypto volatility forecasting: ML vs GARCH

Bruno Spilak
Wolfgang Karl Härdle

Ladislaus von Bortkiewicz Chair of Statistics

C.A.S.E.-Center for Applied Statistics and Economics

International Research Training Group

Humboldt-Universität zu Berlin
lvb.wiwi.hu-berlin.de

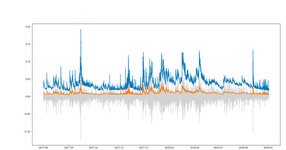
www.case.hu-berlin.de

irtg1792.hu-berlin.de

GARCH parameters

Parameters	Average	std
ar1	0.41	0.028
ar2	0.051	0.01
ar3	-0.0024	0.0013
ma1	-0.59	0.037
alpha1	-0.031	0.0052
beta1	0.54	0.0322
gamma1	0.28	0.0066
mu	0	0

Table 3: Parameters stability



LSTM equations

Input gate, i_t at time t and candidate cell state, C_t^*:

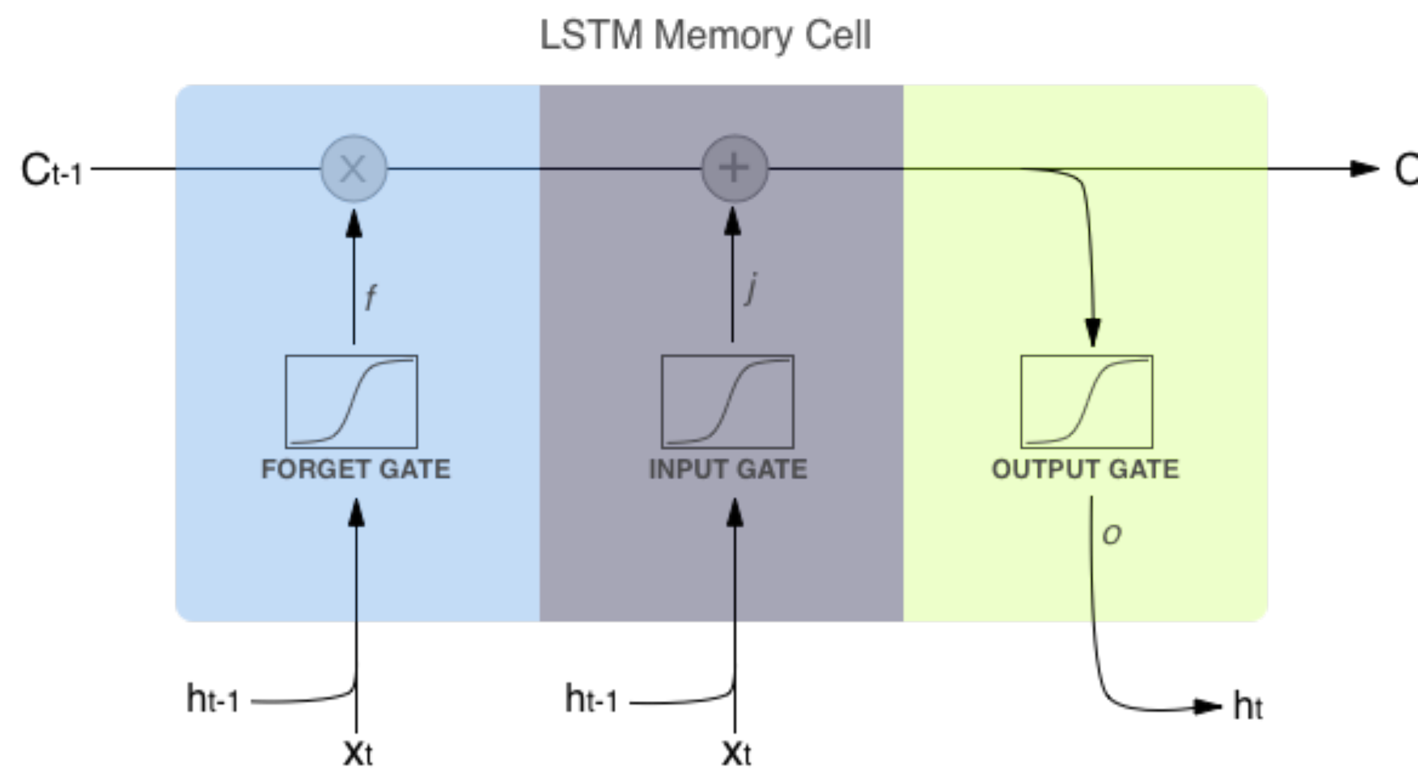
$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i)$$
$$C_t^* = \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c)$$

Activation of the memory cells' forget gate, f_t at time t and new state, C_t:

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f)$$
$$C_t = f_t C_{t-1} + i_t C_t^*$$

Activation of the cells' output gate, o_t, at time t and their final outputs, h_t:

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o)$$
$$h_t = o_t \tanh(C_t)$$



where W_{ab} is the weighted matrix from gate a to gate b