Forecasting and Analyzing Financial Markets with Deep Learning Methods

Yuriy Sosnin   Vladimir Pyrlik

HSE University

June 9, 2022

Outline

Introduction

- Financial time series forecasting is a difficult task
- Models used for it are still mostly linear
- At the same time, Deep Learning is a powerful prediction tool
- Literature about it is inherently technical

Introduction

Do fundamental macroeconomic factors improve forecasts of the Russian stock market?

What we do:

1. Train 3 different Neural Network architectures on IMOEX
2. Assess whether including certain groups of macroeconomic features improves model performance
3. Interpret model predictions with SHAP, a feature attribution method

Data

IMOEX, daily data on 2018-07-04 – 2019-11-01

- Shares of top IMOEX constitutes
- Moscow Exchange sectoral indexes
- Euro and USD to Ruble exchange rates
- Commodity prices of Oil, Natural Gas, and Gold
- Foreign market indexes
- Yields of Russian government bonds

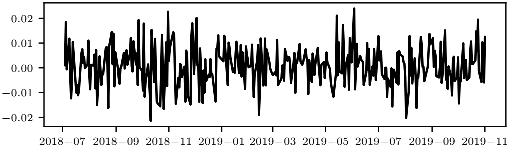


Figure: Target Series

Data

IMOEX, daily data on 2018-07-04 – 2019-11-01

- Shares of top IMOEX constitutes
- Moscow Exchange sectoral indexes
- Euro and USD to Ruble exchange rates
- Commodity prices of Oil, Natural Gas, and Gold
- Foreign market indexes
- Yields of Russian government bonds

- Logarithmic Return
  $r_t = \log P_t^C - \log P_{t-1}^C$
- HighLow
  $HighLow_t = \frac{P_t^H - P_t^L}{P_t^L}$
- Volatility
  $RV_t = \sqrt{\sum_{i=1}^{T} r_i^2}$
- Logarithmic Volume
  $LogVolume_t = \log\left(Volume_t + 1\right)$

Model Specifications

- Naïve Zero
  A baseline predictor, always predicting 0 as each day's return. Since all data is normalized, it corresponds to AR(0), estimated on training period.
- LSTM
  Long Short-Term Memory Neural Network, a go-to model for time series forecasting (Fischer & Krauss, 2018; Greff et al., 2017).
- CNN
  An architecture mainly designed for image processing, uses special Convolution layers to extract spatial information form multidimensional inputs (Hoseinzade & Haratizadeh, 2018; Mehtab & Sen, 2021).
- CNN-LSTM
  A hybrid architecture: CNN pre-processes inputs, extracting a new set of features, which are sequentially processed by LSTM (Livieris et al., 2020; Lu et al., 2020).

Modelling Technique

Multivariate rolling-window one-step-ahead regression forecasting.
T = 10
The data is reshaped into input from of X, then normalized, i.e. demeaned and divided by standard deviation.
Metric and Loss Function is MSE

$$MSE = \frac{1}{N}\sum(y_t - \hat{y}_t)^2$$

$$X_t = \begin{bmatrix} y_t & x_t^{(1)} & \dots & x_t^{(n)} \\ y_{t-1} & x_{t-1}^{(1)} & \dots & x_{t-1}^{(n)} \\ y_{t-2} & x_{t-2}^{(1)} & \dots & x_{t-2}^{(n)} \\ \vdots & \vdots & \ddots & \vdots \\ y_{t-T} & x_{t-T}^{(1)} & \dots & x_{t-T}^{(n)} \end{bmatrix}$$

$$\hat{y}_{t+1} = f(X_t)$$

Modelling Technique

1. Walk-Forward Cross-Validation
   1.1 12-week long test period is split into 12 parts
   1.2 All models are trained and validated on 1-year long data preceding each of test periods
   1.3 Best hyperparameters are chosen for each of 3 architectures
   1.4 Different models are compared by averaging test MSE
2. The models are re-trained on reduced set of features, iteratively *excluding* each of variable groups.
3. One of the models is interpreted with SHAP.

SHAP

SHapley Additive exPlanations (Lundberg & Lee, 2017)

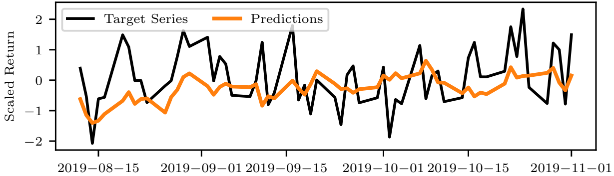
- Based on Shapely Values from Game Theory
- Locally expand output as linear sum of "attributions"
- Helps interpret Black-Box model predictions
- Global attributions are obtained by aggregating across different dimensions

$$\begin{bmatrix} \phi_t & \phi_t^{(1)} & \dots & \phi_t^{(n)} \\ \phi_{t-1} & \phi_{t-1}^{(1)} & \dots & \phi_{t-1}^{(n)} \\ \phi_{t-2} & \phi_{t-2}^{(1)} & \dots & \phi_{t-2}^{(n)} \\ \vdots & \vdots & \ddots & \vdots \\ \phi_{t-T} & \phi_{t-T}^{(1)} & \dots & \phi_{t-T}^{(n)} \end{bmatrix}$$

$$\hat{y}_{t+1} = \sum_{v=0}^{n} \sum_{l=-T}^{0} \phi_{t-l}^{(v)}$$

Empirical Results

Model Performance



Model	MSE
Naïve Zero	0.8793
LSTM	1.0643
CNN	**0.873**
CNN-LSTM	0.8834

Table: Model Performance

Conclusion:

- Models fail to outperform naïve predictor
- Still, full set of features yields better results
- This means there is important information in macroeconomic variables

Model Performance

Model	All features	Sectors	Shares	Curr.	Comm.	Foreign	Bonds
LSTM	**1.0643**	1.3561	1.4998	1.4968	1.6165	1.7486	1.1959
CNN	**0.873**	1.0061	0.951	0.9916	0.9182	0.9642	1.0135
CNN-LSTM	**0.8834**	0.9428	1.1223	1.1447	1.045	0.9582	1.0581

Table: MSE, *excluding* features

Feature Attribution

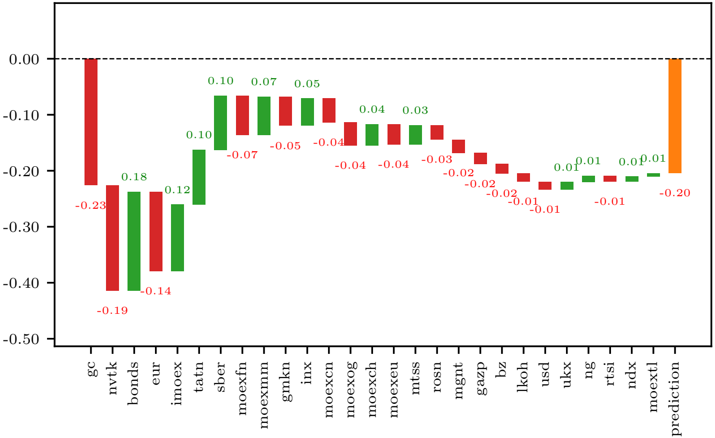
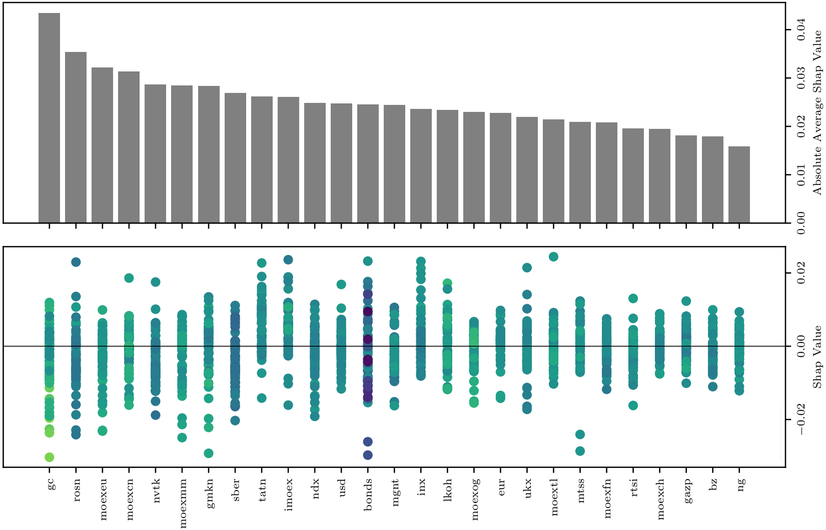


Figure: Local explanations for 12-08-2019

Feature Attribution



Feature Attribution

Surprising importance of Gold

- 2019 was a splendid year for gold prices
- Investors might have used it as hedge against geopolitical risks
- Or gold prices correlated with general news sentiment

Relative unimportance of Oil & Gas

- Korhonen and Peresetsky, 2016; Robert D. Gay, 2008
- Earlier studies do not find oil prices driving Russian stock market
- It advocates for Russian economy being mature and diversified

Conclusion

Do fundamental macroeconomic factors improve forecasts of the Russian stock market?

1. We fail to come up with a good forecasting model
2. Nevertheless, we find that all included variables indeed improve forecasts
3. We were able to extract a non-obvious relationship between IMOEX and Gold prices in August-November of 2019, as well as other interpretable conclusions.

References I

- Fischer, T., & Krauss, C. (2018). Deep learning with long short-term memory networks for financial market predictions. *European Journal of Operational Research*, *270*(2), 654–669.
- Greff, K., Srivastava, R. K., Koutník, J., Steunebrink, B. R., & Schmidhuber, J. (2017). LSTM: A Search Space Odyssey [Conference Name: IEEE Transactions on Neural Networks and Learning Systems]. *IEEE Transactions on Neural Networks and Learning Systems*, *28*(10), 2222–2232.
- Hoseinzade, E., & Haratizadeh, S. (2018). *CNNPred: CNN-based stock market prediction using several data sources* (tech. rep. arXiv:1810.08923) [arXiv:1810.08923 [cs, q-fin, stat] type: article]. arXiv.
- Korhonen, I., & Peresetsky, A. (2016). What Influences Stock Market Behavior in Russia and Other Emerging Countries? *Emerging Markets Finance and Trade*, *52*(5), 1210–1225.

References II

Livieris, I. E., Pintelas, E., & Pintelas, P. (2020). A CNN-LSTM model for gold price time-series forecasting. *Neural Computing and Applications*, *32*(23), 17351–17360.

Lu, W., Li, J., Li, Y., Sun, A., & Wang, J. (2020). A CNN-LSTM-Based Model to Forecast Stock Prices. *Complexity*, *2020*, e6622927.

Lundberg, S. M., & Lee, S.-I. (2017). A Unified Approach to Interpreting Model Predictions. *Advances in Neural Information Processing Systems*, *30*.

Mehtab, S., & Sen, J. (2021). *Stock Price Prediction Using Convolutional Neural Networks on a Multivariate Timeseries* (tech. rep.) [arXiv:2001.09769 [cs, q-fin]].

Robert D. Gay, J. (2008). Effect Of Macroeconomic Variables On Stock Market Returns For Four Emerging Economies: Brazil, Russia, India, And China. *International Business & Economics Research Journal (IBER)*, *7*(3).

Model Architectures

```
LSTM(
   (lstm): LSTM(101, 16, num_layers=1)
   (act): ReLU()
   (fc): Linear(in_features=320, out_features=1, bias=True)
)
```

Model Architectures

```
CNN(
  (cnn): Sequential(
    (0): Conv1d(101, 16, kernel_size=(1,), stride=(1,))
    (1): ReLU()
    (2): Conv1d(16, 32, kernel_size=(3,), stride=(1,))
    (3): ReLU()
    (4): MaxPool1d(kernel_size=2, stride=2, padding=0, dilation=1)
    (5): Conv1d(32, 64, kernel_size=(3,), stride=(1,))
    (6): ReLU()
    (7): MaxPool1d(kernel_size=2, stride=2, padding=0, dilation=1)
  )
  (fc): Linear(in_features=64, out_features=1, bias=True)
)
```

Model Architectures

```
CNNLSTM(
  (cnn): Sequential(
    (0): Conv1d(101, 256, kernel_size=(2,), stride=(1,))
    (1): ReLU()
    (2): Conv1d(256, 512, kernel_size=(2,), stride=(1,))
    (3): ReLU()
    (4): MaxPool1d(kernel_size=2, stride=2, padding=0, dilation=1)
  )
  (lstm): LSTM(512, 32, num_layers=2, dropout=0.1)
  (fc): Linear(in_features=128, out_features=1, bias=True)
)
```