HIGHER SCHOOL OF ECONOMICS
NATIONAL RESEARCH UNIVERSITY

RESEARCH PAPER:

Stock Price Prediction Using Machine Learning

Authors:

Economics faculty students:

BEC181

Grigoryan Mikhail Oganesovich

Romanenko Aleksandra Andreevna

Scientific advisor:

Mamedli Mariam Oktaevna

Moscow, 2021

Contents

1 Introduction

One of the most challenging machine learning challenges today is applying models to evaluate financial indices or stock prices. Due to the development of machine learning methods, approximations to accurate prediction are emerging. Moreover, modern models, unlike standard econometric methods, are capable of capturing nonlinear patterns and complex patterns in the input data. The issue of forecasting accuracy is of interest not only to private investors, but also to large companies and large institutional investors as well. The Russian financial market is developing more and more actively every year, therefore, the search for an optimal model for predicting the main indices of the Russian stock market is an important applied problem.

In our work, we strive to build a model that will predict well the next day's stock price. To predict the course, we use an ensemble of different models, trend prediction, and work with news. In the process of selecting the best ensemble, we identify which methods of processing and using information are the best and announce why this is so.

As the main indicator, the values of the Moscow Exchange index were taken, which allows obtaining the knowledge necessary for making important investment decisions. In addition, this index is used as a benchmark for building the absolute majority of Russian index mutual funds. Therefore, predicting the values of the underlying index provides a number of opportunities in the investment field. In addition, a better model can be applied to predict stock prices.

2 Literature Review

In recent years, there has been a significant increase in using machine learning and deep learning in stock price prediction. Currently existing literature can be divided into categories based both on the methods of price prediction and the values taken into account for the model. Exploring methods for stock price prediction, several areas can be distinguished: fundamental analysis, technical analysis and time series analysis.

The first one is aimed at studying the fundamental indicators that influence the change in the value of the index. It relies on company performance, global trends and analytic reports. This type of analysis is used for long-term forecasting and is useful for investors who are not looking for instant profit. Fundamental analysis, as a rule, does not take into account short-term changes in the value of an asset and takes into account the general trend of price behavior.

On the contrary, it is not common to use fundamental indicators of the company's activity in technical analysis. In this method the previous dynamics of changes in the value of the index is analysed. It is based on the idea that the price already reflects the entire situation in the market, and therefore, based on one price, it is possible to make

predictions about subsequent changes. This method is used by investors whose goal is to make a profit in the short term. It gives more weight to recent changes rather than long-term dynamics of the asset's value.

The third method includes many models based on the specifics of time series. Such models are subdivided into linear and non-linear, the first of which includes the classical econometric models ARMA, ARIMA and others. Machine learning and deep learning methods refer to non-linear time series learning models. They allow to take into account the hidden dynamics of changes in value, recognizing patterns like a human brain. With the development of this field of machine learning, there are more and more such models and their variations every year, the most famous of them are ARCH and GARCH methods, convolutional, recursive neural networks (CNN, RNN) and Long Short Term Memory (LSTM).

Researchers in this field use a variety of time series forecasting methods. A number of works take the series directly as a basis and work with prediction based on identifying trend components, seasonality and other characteristics of the series. Others use derived variables such as the difference between the maximum and minimum values over a period, a moving average and standard deviation over several days and / or weeks.

Less commonly, extraneous predictors are used to predict a time series. So, wind speed is used to predict temperature, and news and macroeconomic indicators are used to predict the value of the financial index. This area seems to us the most attractive and least explored at the moment, but basic analysis and forecasting of time series still remains the leading area of our research.

Since simple linear models are not a good way to predict stock prices due to the fact that they do not account for hidden price patterns, recent works in this area are aimed at creating hybrid models that combine the advantages of linear ARIMA and non-linear learning methods. [9] in their work try to use a hybrid model. With the development of neural networks, it became increasingly clear that it is important to capture non-linear dependencies. Therefore, in their work they combine the linear integrated autoregressive model (ARIMA) which dominated in the last century and the support vector machine (SVM). Their work is based on the assumption that combining two models that work well separately will capture both linear and non-linear parameters. The hybrid model is supposed to be an alternative to single models, which allows to improve the quality of prediction. However, the results showed that the quality can be improved only marginally. Thus, the theoretical basis of the work of the early twenty-first century showed that hybrid models that capture nonlinear features should perform well, but there is still a huge field for working with such models.

Another actively developing area of stock price prediction is the use of machine learning methods. Since these methods make it possible to capture nonlinear patterns

in the data, they are actively used both independently and in hybrid models in combination with autoregressive methods. Recent researches have shown that machine learning methods are good at modeling the dynamics of stock prices. In general, methods are classified into single baseline models and ensemble models that combine several teaching methods. The authors of most works use logistic regression, support vector machines and the method of k-nearest neighbors. [1, 10, 4, 6, 3]. Logistic regression is a basic classification model that predicts the direction of price change for the next day. A more advanced application of the model makes it possible to obtain a probabilistic parameter of price change, thereby significantly improving the prediction result. SVM, in turn, allows you to solve the classification problem by constructing a hyperplane that separates data based on their belonging to a certain class. However, a study of the above methods based on data from the Kuala Lumpur Stock Exchange showed that both methods alone do not provide good results in predicting prices for the next day, the quality ranged from 50 to 65%. At the same time, it was found that the SVM shows itself better than the LR and is on a par with the ensemble models. This suggests that a promising area for research is the use of hybrid or deep learning models that are better at capturing. [5].

Returning to ensemble models, we note that they include both models of machine learning algorithms based on the composition of several methods, and some neural networks. At the same time, the ANN basic neural network does not belong to ensemble models, since only a few hidden fully connected layers with nonlinear activation functions are used. Random forest is a very popular model for predicting the dynamics of stock prices. [13] The RF algorithm assumes the use of an ensemble of trees, which uses all possible partitions of the original vectors in the course of repeated recursive processes, which at the output gives the best partition used for binary classification. This method has been found to improve the prediction accuracy in time series models. [13] However, studies have also shown that RF does not win in quality over SVM in predicting financial indicators. However, studies have also shown that RF does not win in quality over SVM in predicting financial indicators. The quality of this method on the Kuala Lumpur Stock Exchange data also did not exceed 70%. [5] This also indicates that the RF is not suitable as a basic model for research.

The most progressive area at the moment is the use of deep learning methods. As mentioned earlier, hybrid models that are capable of capturing nonlinear patterns in data are especially widespread due to their greater efficiency. These models include various types of neural networks that, like the human brain, are able to find patterns that are not described by linear models. With the growth in the use of deep learning due to its good analytical results for the financial market, the variety of prediction methods has also expanded. The earliest works use simple neural networks consisting of fully connected layers with activation functions (ReLU, Sigmoid). Using neural

networks in the earliest works, Zhang (2003) used a hybrid model of ARIMA and a neural network, proving its effectiveness in three unrelated areas: weather data related to sunspots, the data about Canadyan lynx and an exchange rate of british pound to USD. He showed that the hybrid model outperforms the single model on the training dataset by more than 30 points on the MSE metric and by more than 60 points on the test data. [17]

More sophisticated neural network models that involve skip connections or convolutional layers have been explored or used in recent studies. Sauda and Shakya used several varieties of RNN in their work to predict the prices of shares of the two most popular and important commercial banks listed on Nepalese equities. They use Vanilla RNN (VRNN), Long Short-term Memory (LSTM), and Gated Recurrent Unit (GRU). The forecasting results showed that LSTM and GRU, as a rule, outperform VRNN, which is logical, because VRNN uses the most basic model. At the same time, in some cases, VRNN bypasses LSTM, since it takes into account fewer parameters and processes a more optimal amount of data. [14] This fact is confirmed by both Weiss and Golberg in their work on the accuracy of language recognition. In addition, studies have shown that processing too much data in retrospective analysis reduces the quality of forecasting for LSTM and GRU. The optimal size of the time horizon must be determined on the basis of metrics and depending on the specifics of the data. However, the general fact is that the change in the value of shares is usually not based on events that occurred more than 5 years ago, and therefore a longer time horizon causes an overfitting of the model. [16] Mehar, in one of the most recent papers, compared the prediction quality of a neural network with an ensemble random forest algorithm using data from Nike, JP Morgan and Co., Johnson Johnson and Pfizer Inc companies. In their study, the authors created a number of new variables based on the open, close, lagged values and mean and deviation in a sliding window of 7, 14 and 21 days. On both PMCE and MAPE metrics, the neural network outperformed the random forest for almost all indices. Moreover, the backlog did not exceed 2 percentage points. [8] Another interesting example of building a hybrid model is a study by the authors Rezaei and Faaljou, who, describing the advantages of a convolutional neural network, use it as part of a series decomposition and LSTM model. [12] CNN, being a neural network with sequential convolutional layers that allows you to catch even small patterns in the data, becomes a good addition to conventional models. If the data first passes through convolutional layers and max pooling, and then is used in the LSTM, then the quality is significantly improved compared to the basic LSTM. In this case, the authors used raw values of the indices, without additional predictors.

Since in our work we aim at using news as a predictor in a model with predicting the value of stocks, an especially interesting area for us is the study of the influence of news data on the quality of forecasting. It is worth noting that this area became

most widespread with the development of neural networks. Vargas and Lima use both news and technical indicators to forecast the S P500. At the same time, as news, the authors take the headlines of news data directly related to the companies included in the monitored index. In their research, the authors propose a number of hybrid models and try different specifications. In the field of news processing, they apply both word-based and whole-sentence embeddings, and for deep learning they use RNN, CNN and specification, combining the advantages of both networks. As a result of the use of various embeddings and complete sets of models, the authors found out that a hybrid neural network that combines the advantages of CNN and RNN manifests itself best. [15] This is justified by the fact that the convolutional network better models the patterns associated with the events of a particular day, and the RNN also takes into account the general contextual dynamics of the stock price movement. In addition, the best quality is shown by models in which embeddings of whole sentences were used rather than individual words. This fact may indicate that sparse words do not give a clear understanding of the event, while the full context of the sentence better reflects the color of the message. At the same time, this result depends on the specifics of the data used and is not a general truth. The authors also noticed that embedding event descriptions improves the quality of the model, but our current goal is to create a model that can identify important events for analysis from regular news headlines. the analysis showed that the addition of technical indicators has a positive rather than negative effect on the quality of the model.

Not only news, but also other sources are used as textual data reflecting the market situation. In addition to news media, Mao and Bollen use data from index-related tweets, search queries, and investor sentiment surveys in their financial market forecasting research. [7] In our opinion, such data are more difficult to aggregate for frequent updating of forecasts, but they are of important research interest. As an assessment of investor sentiment, the authors use the Investor Intelligence (II) and Daily Sentiment Index (DSI) indices. They also build their own indices based on the processing of data from tweets and news headlines, it is interesting that the indices built by the authors do not show significant correlation between themselves and with data from investor surveys. Despite the fact that the authors of the article used fairly simple linear models, their results suggest that expanding the types of data sources on investor sentiment in general can improve the quality of predictions. An important fact for future research is that user tweets tend to outpace search query data, anticipating the sentiment recorded by search engines.

Despite a significant amount of work addressing machine learning and deep learning predicting stock prices, this area continues to be understudied and is of keen interest for a number of reasons. First, the ability to predict financial performance challenges the efficient market hypothesis. In addition, the idea of predicting changes in financial

indicators based on open data attracts not only investors, but also large companies as a potential source of profit. But despite such an abundance of teaching methods and data sources, the entire body of works is often united by one rule - the results obtained by the authors are reproducible solely with the specification they use and on their data. Changing one of the parameters or the predicted index often leads to an inevitable deterioration in the quality of predictions, which makes existing studies not universal and very narrowly applicable.

3 Data

To study the forecasting quality of various machine learning methods, we chose the Moscow Exchange Index (MCX: IMOEX) as the main financial indicator. This index is a market capitalization-weighted stock index that tracks the 50 most liquid stocks of Russian companies. This number may vary, for example, in 2019, 41 organizations were included in IMOEX. It is important that the types of activities of issuers whose securities are included in the monitored index represent the main sectors of the economy represented on the Moscow Exchange. As a rule, about 50% of the index is made up of the energy resources' companies ("Gazprom" and "Lukoil" have the largest weight), followed by financial organizations, in particular, about 15% is accounted for by "Sberbank". The closing price is used for calculations. Calculations are made from 10:00 to 18:30 Moscow time, their frequency is once a second. Mathematically, the index is calculated as follows:

$$Index_T = Index_{T-1} \cdot \frac{\sum_{i=1}^{M}(P_{iT}^t \cdot Q_{iT} \cdot FF_{iT} \cdot W_{iT})}{\sum_{i=1}^{M}(P_{i(T-1)} \cdot Q_{iT} \cdot FF_{iT} \cdot W_{iT})}$$

where

- $Index_T$ - index value at time T
- M - the total number of securities of emitents included in the calculation of the index
- P_{iT}^t - ith paper price calculated at time t for day T
- Q_{iT} - volume of ith paper emission and in pieces
- FF_{iT} - free-float ratio of ith paper
- W_{iT} - the weight of ith paper

And the weights of emitents in the index are determined as follows:

$$w_i = \frac{Cap_i}{\sum_{i=1}^{k} Cap_i}$$

where

- Cap_i - capitalization of emitent i
- w_i - share of the capitalization of issuer i in the total capitalization of issuers included in the tracked index

This index allows us to track the state of the market in general. Since its dynamics is formed on the basis of the dynamics of the largest companies represented in it, this index should reflect the reaction of investors and issuers to the largest events that are reflected in the news. The periods for training and validating the model are presented below. For forecasting, the last year is used, which is very logical and convenient in the available data. The reason why September 2013 is taken as the starting point will be explained

Full data	Training	Test
17.09.2013 - 18.09.2021	17.09.2013 - 31.12.2020	01.01.2021 - 18.09.2021

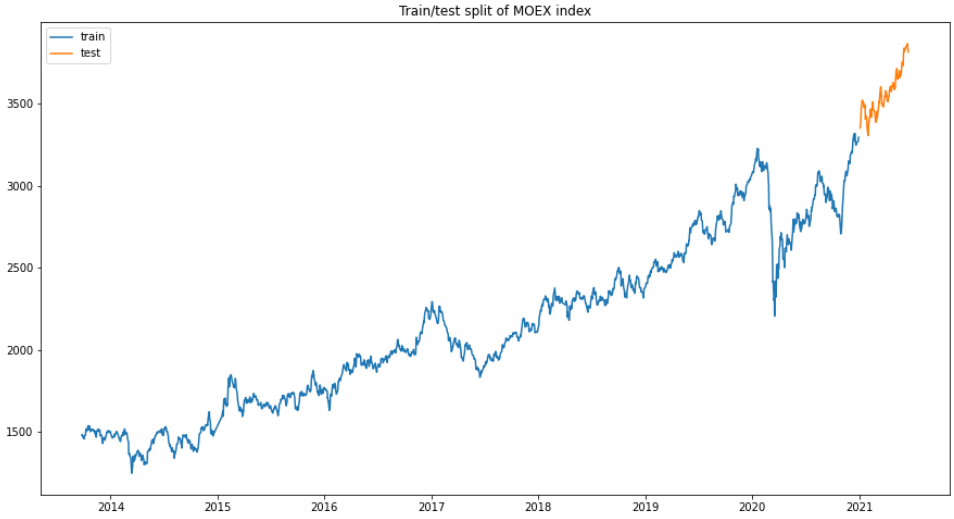


Figure 1: Training and validation time series components

Training models on only one financial index usually leads to their retraining. Therefore, we use additional indicators that reflect the dynamics characteristic of the Russian market in a different direction. So for a narrower analysis, we take the blue-chip index (MCX: MOEXBC). It corresponds to the shares of the top one and a half dozen of the most liquid and capitalized companies. This index has a 20% limit on the share weight of one issuer and is calculated once per second from 10:00 to 18:50. This index belongs to a narrower segment of the market, since it includes three times fewer companies than the first. At the same time, the specificity of the Russian market is such that the first 15 companies in terms of capitalization set the overall dynamics of both indices.

The broad market share index (MCX: MOEXBMI) was taken as the third analyzed index. This is the broadest indicator that includes 100 securities selected by liquidity criteria, free-float ratio and market capitalization. The base for calculating the Broad Market Index is used to form the sectoral indices of the Moscow Exchange. All indices are balanced once a quarter, and a detailed calculation method is presented on the Moscow Exchange website.

As technical financial indicators that are used in a number of models as predictors, we use several time series with macroeconomic data. Among these is the key rate of the Central Bank. It plays the role of a signal to investors and informs them about the monetary policy of the bank. In Russia, the mega-regulator, which became the Central Bank, appeared only in 2013, therefore the data on the key rate is relevant since September 2013. In addition, we use the dollar / ruble rate and the oil price (Brent) as predictors. Both of these indicators are important for the Russian economy and often have a strong influence on the dynamics of the share price of the largest companies.

Table 1: Index Correlation

	MOEX	MOEXBC	MOEXBMI	Brent	USD/RUB
MOEX	1.000000	0.996620	0.999838	0.233506	0.538797
MOEXBC	0.996620	1.000000	0.997176	0.248590	0.526176
MOEXBMI	0.999838	0.997176	1.000000	0.232258	0.533291
Brent	0.233506	0.248590	0.232258	1.000000	-0.401561
USD/RUB	0.538797	0.526176	0.533291	-0.401561	1.000000

As we can see from the correlation table, the leading indices of the Moscow Birdie are quite strongly correlated with each other. This is due to the fact that the Russian market is highly capitalized. A huge share belongs to the top ten companies. Therefore, MOEX index will be the leading in our study, while the rest will act as a check for the absence of model overfitting.

To track events in the economy, we use headlines from the news portal Lenta.ru. This source was chosen for the reasons that it collected news available for parsing for a fairly long period of time. In our research, we use archived news from the economics section. On average, the site published about 10 posts on economic topics per day. Less news was published on weekends and holidays, however, the exchange does not work on these days, and therefore a decrease in the number of news feeds does not have a big impact on the quality of forecasting.

Then, we processed the news using various embeddings, since the model cannot perceive news in raw. We tried TF-IDF Vectorizer, Word2Vec, FastText and GloVE. Then we built linear regressions in order to assess the influence of the resulting vectors on the value of the index. The TF-IDF vectorizer proved to be the best, which can be

explained by the longer vector length than in other embeddings. It is noteworthy that the quality of Lasso regression was much higher than the quality of conventional linear regression, that is, many words turned out to be insignificant and rather interfered with the estimation of the regression.

Therefore, we chose the Word2Vec embedding as the final one, since it translates phrases that are close in meaning to vectors that are close in Euclidean space. We also built a lasso regression in order to remove the most insignificant components of the vectors, and then applied principal component analysis to dimensionality reduction (the main advantage of the principal component method is that it leaves as much information as possible on the minimum dimension).

4 Quality Metrics

To compare the models with each other, we have chosen several quality metrics that are best interpreted on our data. Metrics which will be calculated are RMSE, MAE and MAPE. This metric is the root of the square of the error. It is easy to interpret since it is calculated in the same units as the original values (as opposed to MSE). It also operates with smaller values in absolute value, which can be useful for optimizing the problem. It is known that MSE-based errors are minimized using the mean, while MAE-based errors are minimized by the medians. Therefore, the RMSE metric allows you to effectively track discontinuous data in some cases. It prevents forecasts from approaching 0 in case of a large number of outliers and is a good complement for MAE.

$$RMSE = \sqrt{\frac{1}{m}\sum_{i=1}^{m}|a_i - y_i|^2}$$

where y - vector of true values and a - model response vector, m - number of observations

This estimate is usually applied for time series, the actual values of which are significantly greater than 1. For example, estimates of the energy consumption forecast error in almost all articles are given as MAPE values. It is a dimensionless coefficient with a very simple interpretation. It can be measured in fractions or percentages. A MAPE result of 10% indicates that the error was 10% of the actual values.

$$MAPE = \frac{1}{m}\sum_{i=1}^{m}\frac{|y_i - a_i|}{|y_i|} \cdot 100\%$$

where y - vector of true values and a - model response vector, m - number of observations

In MAE, the error is calculated as the average of the absolute differences between

the target values and the predictions. MAE is a linear estimate, which means that all individual differences are weighted by the same average. This distinguishes it from the RMSE metric. What's important about this metric is that it punishes huge errors better than MSE does. Thus, this metric is not as sensitive to outliers as the root mean square error. MAE is convenient to use in our case, as it reflects the fact that a 100$ error is much worse than a 50$ error.

$$MAE = \frac{1}{m}\sum_{i=1}^{m}|a_i - y_i|$$

where y - vector of true values and a - model response vector, m - number of observations

5 Methodology

In our work, we aim to find the best model for predicting stock prices and identify the advantages and disadvantages of different forecasting approaches. We used both standard econometric forecasting methods and machine and deep customary models. In the following sections, the algorithm for using the models, their theoretical justification, the results of their application in predicting the financial index and the conclusions drawn will be described.

5.1 Econometrics Models

To begin with, we decided to work with econometric models. The main purpose of this section is to prove that linear econometric models are not capable of capturing non-linear patterns in data. As a result, their quality when trained on a financial index dataset should be extremely poor.

5.1.1 ARIMA

The most common in recent researches is the ARIMA model. It is an extension of the ARMA model, in which, in addition to AR and MA and the component, differences of a certain order are taken. Mathematically, ARIMA (p, d, q) is represented as follows:

$$\Delta^d X_t = c + \sum_{t=1}^{p}\alpha_i \Delta^d X_{t-i} + \sum_{j=1}^{q}\beta_{t-j} + \varepsilon_t,$$

where c, α, β are model parameters, Δ^d - time series difference operator of order d and ε is a stationary time series. As mentioned earlier, ARIMA performs poorly on volatile non-linear financial data. This is confirmed by the results in Table 2.

5.1.2 Hindman-Khandakar algorithm

In order to automate the routine procedure for selecting parameters in the ARIMA model, there is a Hindman-Khandakar algorithm. Generally, the algorithm is an iterative procedure that, by changing the model parameters p, d, q, and, minimizes the value of the corrected Akaike information criterion. The Akaike information criterion and the corrected Akaike information criterion for ARIMA are characterized by the following formulas:

$$AIC = -2log(L) + 2(p + q + k + 1)$$

$$AIC_c = AIC + \frac{2(p + q + k + 1)(p + q + k + 2}{T - p - q - k - 2},$$

where L is the maximized value of the likelihood function of the model; $k = 1$ if the constant according to the model $c \neq 0$ and $k = 1$ if the opposite is true. This information criterion has been developed and is used to select the best of several statistical models. The results of the application of this algorithm are also described in the Table 2 and are under the AutoARIMA label, since this method bears this name in most statistical packages. As can be seen, it did not significantly improve the prediction quality after the iterative procedure for selecting the parameters. This is further evidence that such linear models do not capture the required patterns in our data.

5.1.3 ETS model

The ETS models are based on the exponential smoothing technique. Exponential smoothing looks like this mathematically:

$$\hat{y}_{(T+1|T)} = \alpha \sum_{i=0}^{\infty} (1 - \alpha)^i y_{t-i},$$

where y_t is the value of the variable in moment t and $y_{T+1|T}$ is the forecast for period $T + 1$ and α is a parameter.

It lies in the fact that with an increase in the time horizon, all the variables included in the model decrease their weight. The advantage of this model is that despite the data transformation, the historical dynamics of the index is not lost. This method will generally perform best when there is a strong trend and seasonality in the data. This is inherent in the name itself, implying the decomposition into error, trend and seasonality in the model.

There is no clearly defined trend in our data, and one cannot speak of a pronounced seasonality. Moreover, it is still a linear model, and therefore it is not able to capture complex patterns in the data. As expected, the model results are not superior to other methods used. Moreover, we can say that among the models used, this is the worst method in the analysis of financial time series. The results of applying the method

with the measurement of basic metrics are presented in the Table 2.

5.1.4 AutoETS

The next method used was the automatic selection of the ETS. This is a special algorithm used in most statistical packages. It automatically adjusts the model parameters, iteratively comparing the performance of each assembly. The number of combinations can exceed tens of thousands, depending on the number of lags used, including seasonal ones, and Fourier pairs. The entire space of combinations is divided into certain segments, which become more complicated during the operation of the algorithm. The most effective segment is selected from these segments and a sample is formed on its basis. As a result, the model is trained on the best parameters.

This algorithm also helps us prove that even with an effective selection of the parameters of the ETS model, it remains ineffective in predicting financial data. The prediction results in the estimated metrics are in the table, you can see that they are not significantly improved relative to the basic ETS model.

5.1.5 Naive Model

The simplest model to implement was the naive model. She makes predictions based on the last observed result. This model does not explain patterns in the data, but only tries to predict based on the latest historical data. For the estimation we use a simple model based only on the latest data and does not analyze seasonal parameters. Its results are shown in the Table 2. Despite its simplicity, it does not lag far behind other models, which shows that even more complex econometric models capture the natural

Table 2: Econometrics models

height	Naive		ARIMA		AutoARIMA		ETS		AutoETS	
	real	log	real	log	real	log	real	log	real	log
RMSE	45596.1	45596.1	54852.4	81434.8	54810.7	80483.1	238967	281588	46617.5	46810.1
MAE	172.689	172.689	194.578	243.301	194.501	241.755	1524.38	1651.94	175.228	175.657
MAPE	6.00229	6.00229	6.60726	7.98924	6.60553	7.94655	34.4956	36.273	6.14469	6.16584

5.2 Machine Learning Methods

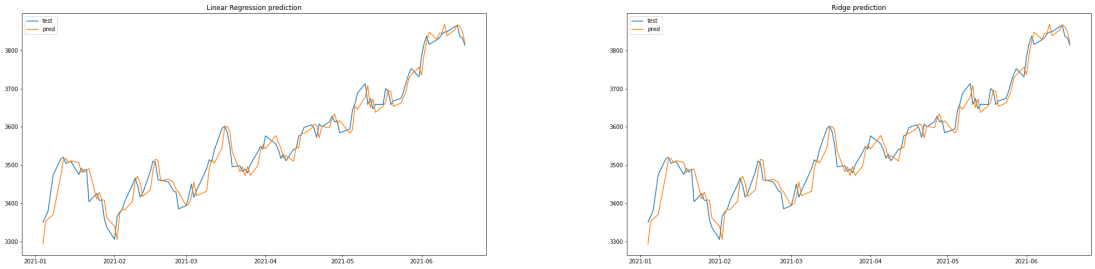
Machine learning is often used for time series forecasting, as it provides different methods of which some are good at forecasting and some are not. In this section we aim to consider general models for fitting and forecasting time series, using MOEX index. In this section we will use Python library "sklearn", since this library implements

many machine learning methods and metrics necessary for evaluating fitted values and predictions. [11]

5.2.1 Linear Models

Linear Models are pretty popular, as they are quite simple in understanding, and provide some useful information for the user. Time series usually autoregressive, we can say for sure that the value of the index today strongly depends on the value of the index yesterday, so we are gonna to use it, that is why we have some lagged variables in the dataset. Also, we suppose that all the dependencies between index values and lagged variables are linear since the index has upward piece-wise linear trend. For all models, except for linear regression, we chose the hyperparameters by minimizing mean squared error on the validation part of the dataset.

Firstly, we tried to fit Ordinary Least Squares Linear Regression, and this model gives some pretty predictions: mean average error is equal to 22.4, or 0.63 %. We can say that predictions are quite accurate, and we can improve them by adding regularization. Lasso Regression (Regression with L1-regularization) demonstrates some progress in improving prediction. Ridge Regression (Regression with L2-regularization) shows slightly worse results compared to Lasso, we can interpret this so that our dataset is slightly noisy (since the improvement is not very large), and and Lasso regression nullifies coefficients in front of noisy variables. Finally, we tried to fit ElasticNet regression, in which both L1- and L2-regularization are implemented, and this model turned out to be the best in the class of ordinary linear models, so both types of regularization are quite useful in forecasting time series.

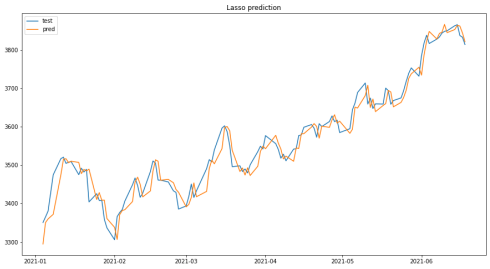


(a) Linear Regression prediction

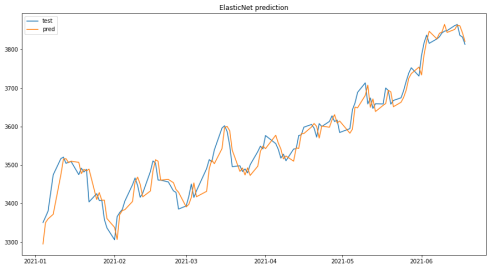
(b) Ridge prediction

5.2.2 Boosting Algorithms

Boosting algorithms are one of the most common models for machine learning, since they are fast enough and use simple rules for training, which allows approximating not only linear dependencies. Therefore, many people use these models for regression analysis, in particular, for predicting time series. Actually, that's why we will also try



(c) Lasso prediction



(d) ElasticNet prediction

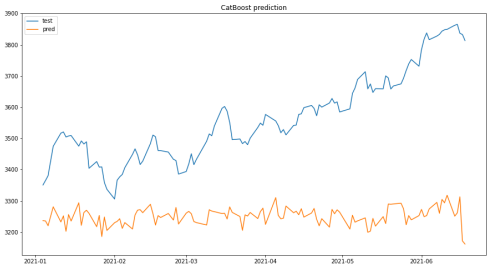
Figure 3: Linear models' predictions

popular boosting algorithms in order to predict the value of the MOEX index. We will start with CatBoost Regressor, provided by Yandex. This model does not show the best results, since in our dataset almost all variables are numeric, and simple rules are not very applicable to analysis, and even more so to predicting an index. Nevertheless, we can get some useful information from the model, for example, consider the importance of features:

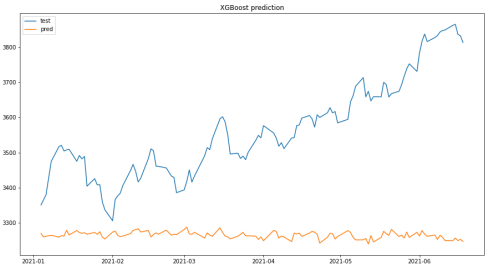
	feature importance	feature names
0	87.03	mAR1
6	9.02	rubHigh_AR1
2	3.01	stavka_AR2
1	0.13	stavka_AR1
5	0.11	stavka_AR5
3	0.08	stavka_AR3
17	0.07	brentHigh_AR2
9	0.04	rubHigh_AR4
12	0.04	rubDelta_AR2
7	0.04	rubHigh_AR2
20	0.04	brentHigh_AR5
11	0.03	rubDelta_AR1
46	0.03	w2v_PC1_AR5
10	0.03	rubHigh_AR5
16	0.02	brentHigh_AR1

As we can see, the value of the MOEX index yesterday is the most important regressor in our model, followed by yesterday's value of the ruble exchange rate and the day before yesterday's value of the interest rate. Most likely we can assume that the market reacts to a change in the key rate not the next day, but only in a day. It is also noteworthy that among the 15 most important regressors, only one refers to processed news, which indicates either that the news is very noisy, or we used not optimal processing and embedding of news.

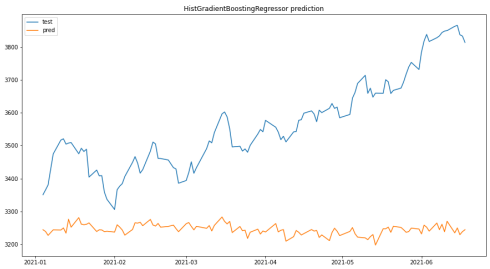
Other boosting algorithms show approximately the same result, their predictions look more like the prediction of Naive Forecaster with the strategy of predicting the last observed value.



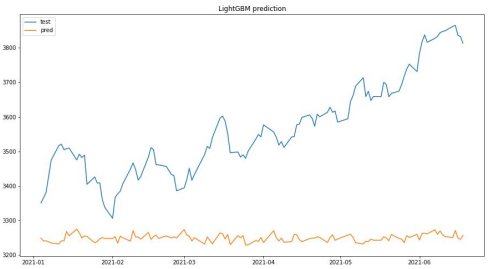
(a) CatBoost prediction



(b) XGBoost prediction



(c) HistGradientBoostingRegressor prediction



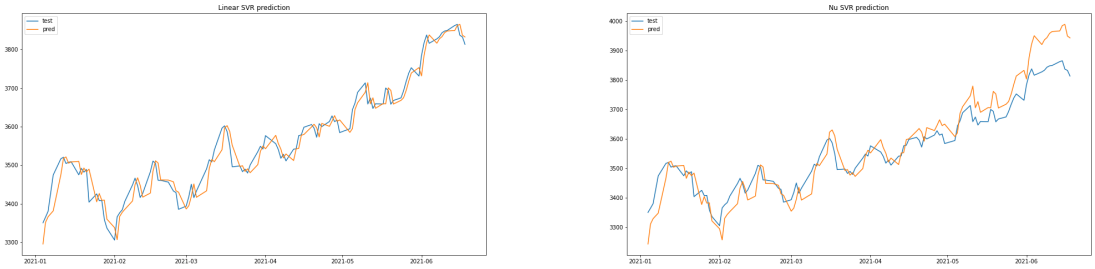
(d) LightGBM prediction

Figure 5: Boosting models' predictions

5.2.3 Support Vector Machine Regression

SVMs are quite popular for time series forecasting, since they may have linear or polynomial kernels, and also they take into account nonlinear relationships during training. Despite the fact that many time series have linear trends and dependences on exogenous variables and lagged values, the time series are often full of non-linearity and irregularity. SVM is quite a compromise model when it comes to choosing between linearity and non-linearity in time series. The ability of SVM to solve non-linear regression estimation problems makes Support Vector Machines successful also in time series forecasting.

We fitted two models from module "svm" of "sklearn": Linear SVR and Nu SVR. Linear SVR has linear kernel, so it is used more for fitting data with linear correlations, while Nu SVR also has non-linear epsilon-support rule, that makes this model more flexible for fitting non-linear data. As a result, Linear SVR was much better, as our index mostly linearly depends on lagged variables and exogenous variables.



(a) Linear SVR prediction (b) Nu SVR prediction

Figure 6: SVM's predictions

5.2.4 More Advanced Models

Gaussian process is the Bayesian approach to solve regression estimation problem, which has some advantages, compared to usual linear regression models. Firstly, we assume the linear regression equation $Y = X\beta + \varepsilon$, then we specify a prior distribution of the coefficients, $p(\beta)$ (the most common is Gaussian distribution). Then the Bayes' Rule is applied to get the aposterior distribution:

$$p(\beta \mid y, X) = \frac{p(y \mid X, \beta) \cdot p(\beta)}{p(y \mid X)}$$

$$posterior = \frac{\text{likelihood} \cdot \text{prior}}{\text{marginal likelihood}}$$

To predict new value by new information, x^*, the predictive distribution is calculated:

$$p(y^* \mid x^*, y, X) = \int_\beta p(y^* \mid x^*, \beta) \cdot p(\beta \mid y, X)\, d\beta$$

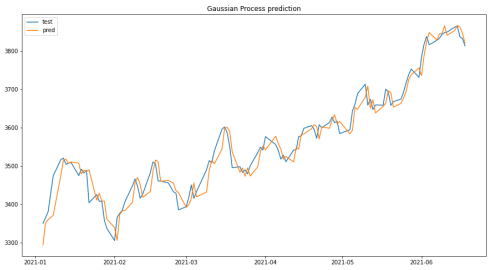
We used gaussian process from sklearn to estimate regression model, and this model performs fairly well given that we are taking a completely different approach to estimating regression coefficients.

Next, we evaluated the regression using SGDRegressor: linear regression model fitted by minimizing stochastic gradient descent. This model proved to be the worst in the class of linear models, since random observations are used for training, and they are correlated with each other. Also from the linear models, we decided to use Least Angle Regression (as knows as LARS) and Passive Aggressive Regressor.

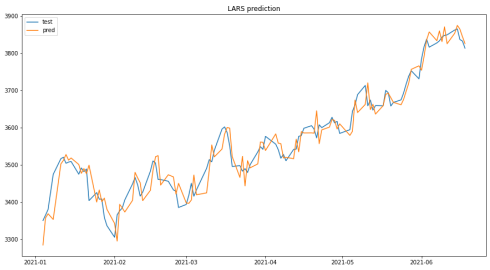
LARS is similar to stepwise regression. At each iteration, it finds the most correlated feature with the MOEX index at period t, and proceeds in a direction of this feature. When there are multiple features having equal correlation, it proceeds in a direction equiangular between the features. This method is practically used when number of regressors is greater than number of observations, but LARS is quite sensitive to noises

in data. This can explain (at least we can assume it can) that the quality of the model is lower than that of ordinary least squared linear regression.

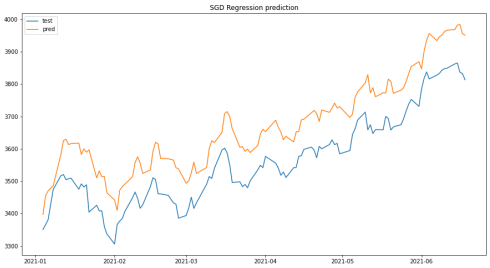
Passive Aggressive regression called so because this model is passive when predictions are correct (no changes in model) and aggressive when prediction are incorrect (than there are some changes in model estimation). It works in similar way as MultiLayer Perceptron, but with regularization parameter and without any learning rate. MultiLayer Perceptron is one of the basic neural networks used for fitting the table data. As we can see in the following mlmet, MLP shows quite more impressive results than Passive Aggressive Regressor.



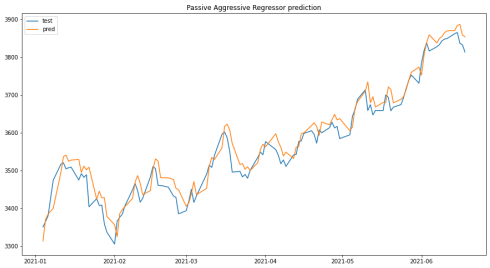
(a) Gaussian Process prediction



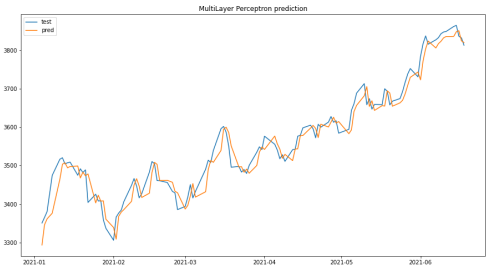
(b) LARS prediction



(c) SGD Regression prediction



(d) Passive Aggressive Regressor prediction



(e) MultiLayer Perceptron prediction

Figure 9: Advanced models' predictions

Table 3: Models' results

Model	RMSE	MAE	MAPE	R^2
Linear Regression	826.4928	22.3954	0.6326	0.9576
Lasso(0.01)	810.3288	22.0751	0.6237	0.9584
Ridge(0.01)	826.3557	22.3931	0.6325	0.9576
ElasticNet	808.9630	22.0539	0.6231	0.9585
CatBoost	117931.1006	314.9963	8.7026	-5.0480
XGBoost	110898.3647	300.5963	8.2883	-4.6873
LightGBM	119242.1553	316.5629	8.7425	-5.1152
HistGradientBoostingRegressor	122968.1938	320.3325	8.8438	-5.3063
Linear SVR	781.6121	21.6378	0.6116	0.9599
Nu SVR	3157.0671	43.6965	1.2096	0.8381
Gaussian Process	816.4521	22.2184	0.6277	0.9581
SGD Regression	12199.4501	106.7263	2.9962	0.3744
LARS	1123.1052	26.4503	0.7487	0.9424
Passive Aggressive Regressor	2344.3457	41.9079	1.1744	0.8798
MultiLayer Perceptron	844.4028	23.1558	0.6529	0.9567

In the bottom line, we see that the best models turned out to be Linear SVR, ElasticNet and Gaussian Process. This can be explained by the fact that many dependencies between the variables in our dataset are rather linear than non-linear, and these models assume either nonlinear features (like Linear SVR and Gaussian Process) or enhanced regularization (ElasticNet).

5.3 Deep Learning Methods

Machine learning methods are used quite often in time series forecasting and using for solutions to other types of problems, as they are most often well interpreted and simple enough to use. The relative simplicity of machine learning methods is both an advantage and a disadvantage of machine learning methods, since simple models do not predict well after being trained on a sufficiently large data set. They can also show poor results if the model specification is incorrectly selected, or noisy variables are included. Problems also arise due to the multicollinearity of some regressors and possibly unaccounted for significant factors of the model. That's why neural networks are used to solve more advanced problems, in particular for time series forecasting.

In our paper we will train three general neural networks for time series forecasting from pytorch library: LSTM (Long Short-Term Memory), RNN (Recurrent Neural Network) and GRU (Gated Recurrent Unit). Their main feature is that when training on the layer t, the model also uses inputs and outputs from past layers, giving them some weight (the closer the layer is to the current one, the greater the weight of its outputs). This allows models to better capture changes in the time series and, as a result, better predict it.

Initially, neural networks were used to classify images and detect objects on them, but eventually neural networks have become a universal toolkit for solving many problems in applied problems.

To predict our series, we first normalized all data with using sklearn MinMaxScaler (range from -1 to 1), and fitted it on a training set (thus, information from the test sample did not leak into the training sample, and therefore into the model). Next, we created custom dataloaders in pytorch to load into our models. And with this dataloaders we can train neural networks.

Using indices similar in composition, described in the data section, we obtained comparable results. Therefore, we do not provide a detailed interpretation of the metrics for other indicators in order to avoid cluttering the work. Note that additional indices were primarily used to check the absence of overfitting, loss plots and detailed descriptions of procedures accomplish this task and reduce the need to present the results of the same models on similar data.

5.3.1 RNN

RNN is recurrent neural network, this is a special architecture of neural networks that takes into account the output of not only the previous layer, but also the layer before that. Due to this, this neural network can capture changes in the behavior of the time series and work better with autoregressive data.

Pytorch RNN is the Elmar RNN with hyperbolic tangent used as non-linearity. Hyperboluc tangent as activation function works quite well with normalized data, since the function is quite sensitive on input $|x| \leq 1$. Elmar RNN is the advanced form of the basic fully recurrent neural network, this form of RNN works faster and more stable on data with rather high variance.

Firstly, we decided to test the robustness of the models by creating a validation set and looking at how the loss functions change in both the training and validation sets. We trained a neural network with a validation sample for 1000 epochs, and we got that it does not overfit much, since with a fall in the train losses, the validation losses also fall.

Then we trained the neural network on a full training set to predict the test. Despite the fact that the model performed well on both the training and validation sets, it did not perform well on the test set, as can be seen from the following graph:

5.3.2 LSTM

LSTM neural network or Long Short-term Memory is a neural network, which is an extension of a recurrent neural network with the ability to learn long-term connections. Since this neural network can perceive and memorize dependencies for a

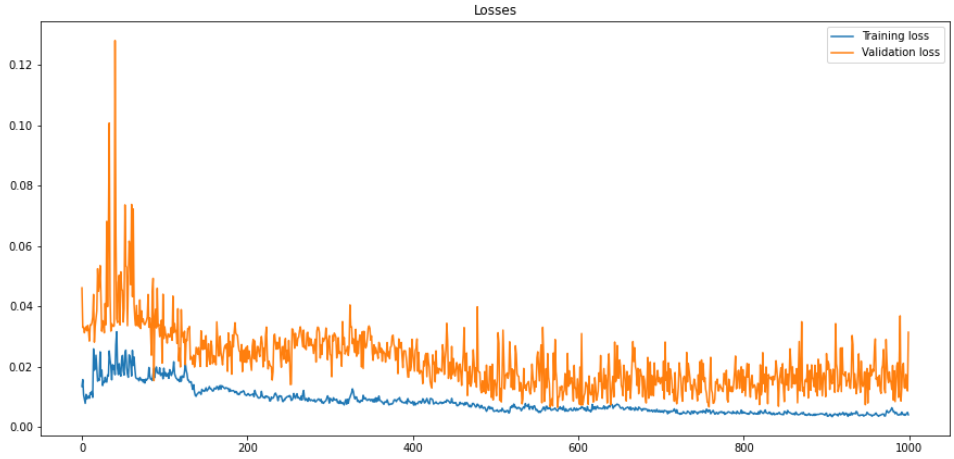


Figure 10: RNN validation losses

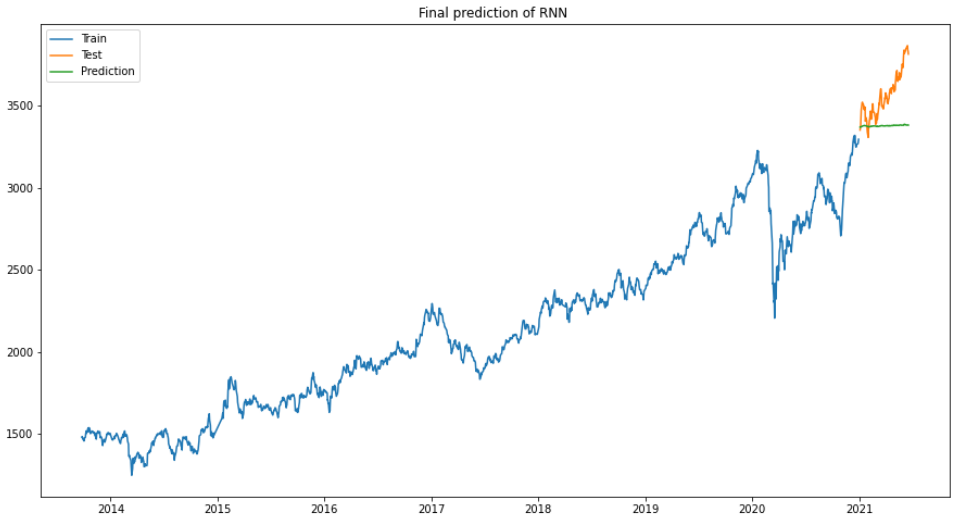


Figure 11: RNN final predictions

sufficiently long period of time, it does not require complex training. LSTM shows itself well in forecasting time series, since this neural network allows you to well learn multidimensional patterns and take them into account when forecasting.

LSTM, like other neural networks, is a chain of state cells.The most important components of the network are the input gare, the forget gate and the output gate, their formulas are as follows:

$$i_G = \sigma(w_i[h_{t-1}, x_t] + b_i)$$

$$o_G = \sigma(w_o[h_{t-1}, x_t] + b_o)$$

$$f_G = \sigma(w_f[h_{t-1}, x_t] + b_f),$$

and σ is a sigmoid function, w is the vector of weights, x is an input at a certain moment, h_{t-1} is an output of a previous LSTM cell and b is for the bias. The cell state is described by following equations:

$$\widetilde{c}_t = \tanh(w_c[h_{t-1}, x_t] + b_c)$$

$$c_t = f_t \cdot c_{t-1} + i_t \cdot \widetilde{c}_t$$

$$h_t = o_t \cdot \tanh(c_t)$$

and c_t is a cell state memory at the moment t.

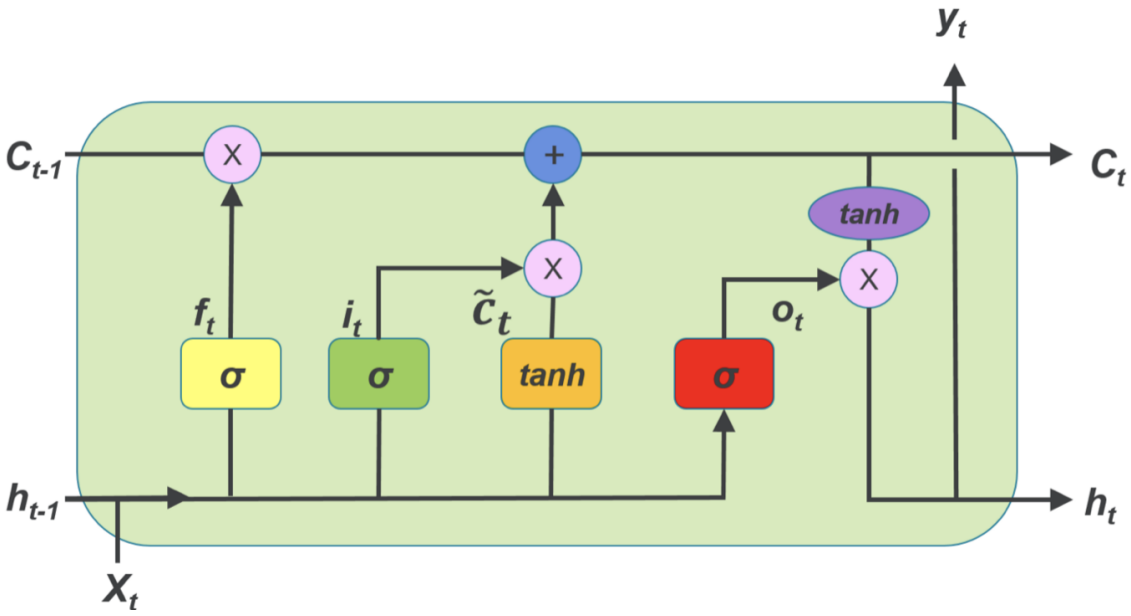


Figure 12: LSTM cell scheme

LSTM caught non-linear patterns quite well, it is noticeable that losses are steadily decreasing both on the train and on the validation. In addition, you can notice that the model is not retrained, which is also seen from the following graph.

However, in terms of predictions, the model slightly outperformed previous specifications. It is noticeable that the prediction quality is very low. Despite the fact that the model is able to capture long-term relationships and learns with a sufficient information volume of predictors, the predictions are not accurate enough.

5.3.3 GRU

Gated Recurrent Unit is the modification of RNN, was firstly proposed by Cho et. al. [2] at 2014. Gated Recurrent Unit is representable in the form of two fully recurrent neural networks, where the first is the decoder, and the second one is the encoder. Initially, this model was used for natural language processing: some phrase was first encoded and then decoded into another phrase. Thus, the model searched

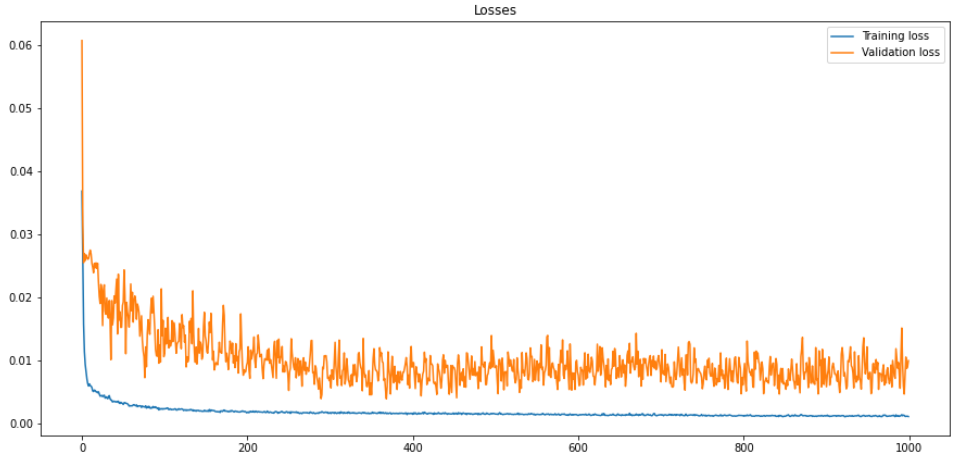


Figure 13: LSTM validation losses

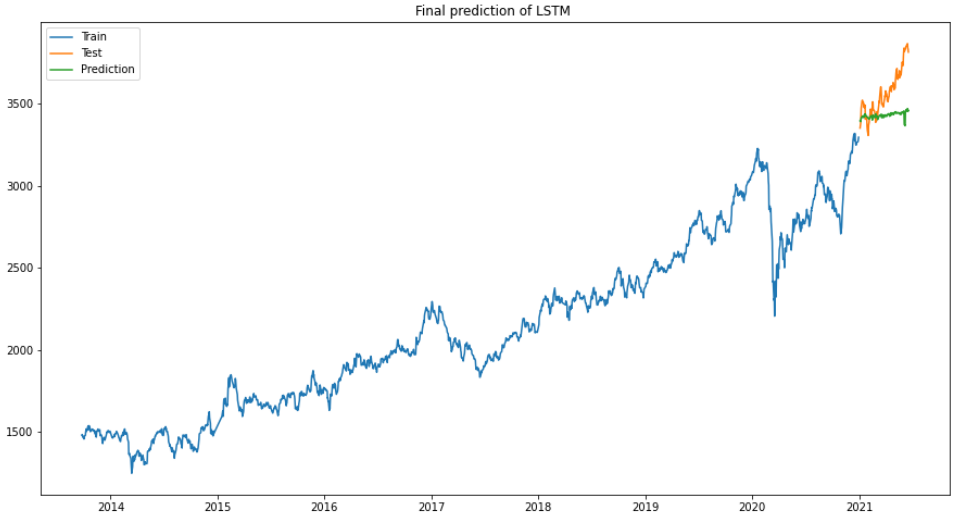


Figure 14: LSTM final prediction

for phrases that were close in meaning. Also, such a structure allows you to better analyze the data, since when training a decoder, it learns to remove noises from data. Therefore, this neural network architecture is used for time series forecasting.

For this model, we also checked how steadily it learns, and the graph with losses during training shows that training this neural network leads to a decrease in losses on the validation set, that is, overtraining is not observed, as it can be seen in the graph:

Like other models, this model did not give very accurate forecasts, but they turned out to be slightly better, as they caught the upward trend, and almost caught the desired slope. The predictions were not as volatile as the real ones.

Contrary to what was expected, neural networks performed worse than many machine learning methods. This can be explained by the fact that we had quite a few

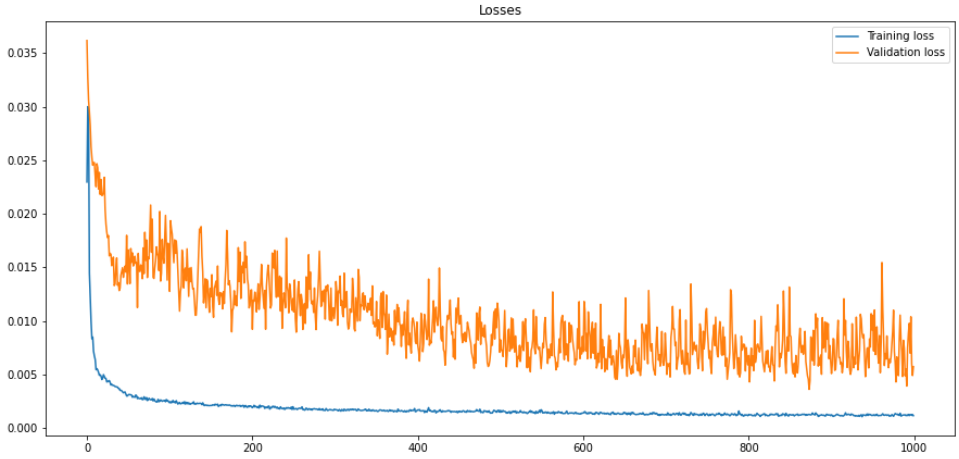


Figure 15: GRU validation losses

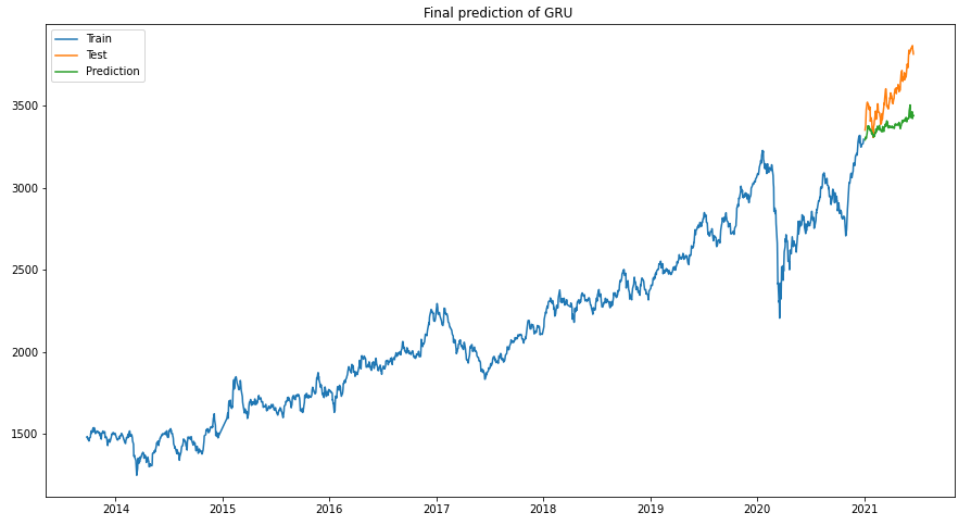


Figure 16: GRU final prediction

data, so neural networks were more likely to try to adapt to the data than to look for patterns, and also not the best choice of the scaler: the test sample was most often underestimated, and did not go beyond 1, although almost the entire test sample the sample was larger than the maximum value of the train. For a better result, either you need to use a different type of scaler, or collect more data, and try to prevent overfitting by introducing regularization.

6 Conclusion

We tried different models for time series forecasting, and got different results for different types of models, due to the peculiarities of their construction. ARIMA models

can work only with stationary rows, that is, with rows that have no trend and seasonality, and whose characteristics do not change over time. If the series is not stationary, then it is reduced to a stationary form using differentiation (normal or seasonal). In our case, the differentiated series rather resembles white noise, the increments are quite difficult to simulate using ARIMA, which makes this model extremely unsuitable. ETS models can work with non-stationary series, but they break them down into error, trend, and seasonality, which is obviously not suitable for our data, since there is no seasonality in the data, and one trend cannot explain the change in the index values. It is impossible to add regressors to the classic ETS models, but you can do it yourself using the code on the PyStan, but this does not show much improvement. Naive models works bad on stock time series, as price of stocks of indices change change almost all the time.

Among the models in machine learning, linear models, SVM models and the Gaussian process, which is the implementation of the Bayesian approach in regression estimation, have proven themselves well. This is because we assembled a dataset from lagged dependent variable variables, exogenous variables and their lags as well. You can also notice that the dependent variable depends on the regressors linearly, so the linear models performed well enough. The best model turned out to be SVM, which shows the importance of taking into account non-linearity, even though the linear models performed well. Non-linearity make it possible to better approximate the time series. Actually, the main idea of using neural networks in forecasting time series consists in capturing non-linearity. A certain architecture of neural networks, recurrent neural networks, do not allow approximating the time series exactly using sigmoid functions, but also takes into account autoregressive motives, since it takes into account the outputs of the previous layers in each layer .Despite the fact that in theory neural networks should perform better than machine learning methods, in practice it turned out that linear models turned out to be much better. This can be explained by incorrect data preprocessing, since normalization by maximum and minimum gives the neural network some idea that the time series does not go beyond 1 in absolute value, so the predictions are unreliable. It is worth reconsidering the choice of the scaler.

To summarize, we evaluated many different models that allow predicting future values of the time series with or without exogenous variables, and tried to compare the quality of these models with each other to find out the advantages and disadvantages of each model. Despite the fact that neural networks could be trained differently, GRU showed quite a good result in catching trends and small fluctuations, so it is possible to develop in this direction further: expand the data set, change preprocessing, change model parameters or make changes to the architecture of the neural network. We also found out that the MOEX index is significantly influenced by the ruble exchange rate and the oil price, which is quite expected, since this affects the expectations of

investors and can change the economic situation. We also found using the CatBoost model that the rate change most significantly affects the index on the second day after the change, but not immediately, and news sources turn out to be too noisy even after embedding and regularization. Perhaps it is worth using more advanced methods of news embedding, taking already pretrained models (specifically on economic news), changing the methods of regularization. In general, the analysis of the MOEX index using machine and deep learning is quite possible, which we have shown in this paper.

References

Michel Ballings et al. "Evaluating multiple classifiers for stock price direction prediction". In: *Expert Systems with Applications* 42 (May 2015). DOI: 10.1016/j.eswa.2015.05.013.

Kyunghyun Cho et al. *Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation*. 2014. arXiv: 1406.1078 [cs.CL].

Rajashree Dash and Pradipta Kishore Dash. "A hybrid stock trading framework integrating technical analysis with machine learning techniques". In: *The Journal of Finance and Data Science* 2.1 (2016), pp. 42–57. ISSN: 2405-9188. DOI: https://doi.org/10.1016/j.jfds.2016.03.002. URL: https://www.sciencedirect.com/science/article/pii/S2405918815300179.

Yu-Pei Huang and Meng-Feng Yen. "A new perspective of performance comparison among machine learning algorithms for financial distress prediction". In: *Applied Soft Computing* 83 (2019), p. 105663. ISSN: 1568-4946. DOI: https://doi.org/10.1016/j.asoc.2019.105663. URL: https://www.sciencedirect.com/science/article/pii/S1568494619304430.

Mohd Sabri Ismail et al. "Predicting next day direction of stock price movement using machine learning methods with persistent homology: Evidence from Kuala Lumpur Stock Exchange". In: *Applied Soft Computing* 93 (2020), p. 106422. ISSN: 1568-4946. DOI: https://doi.org/10.1016/j.asoc.2020.106422. URL: https://www.sciencedirect.com/science/article/pii/S1568494620303628.

Dennys C. A. Mallqui and R. Fernandes. "Predicting the direction, maximum, minimum and closing prices of daily Bitcoin exchange rate using machine learning techniques". In: *Appl. Soft Comput.* 75 (2019), pp. 596–606.

Huina Mao, Scott Counts, and Johan Bollen. *Predicting Financial Markets: Comparing Survey, News, Twitter and Search Engine Data*. 2011. arXiv: 1112.1051 [q-fin.ST].

Sidra Mehtab, Jaydip Sen, and Abhishek Dutta. *Stock Price Prediction Using Machine Learning and LSTM-Based Deep Learning Models*. 2020. arXiv: 2009.10819 [q-fin.ST].

Ping-Feng Pai and Chih-Sheng Lin. "A hybrid ARIMA and support vector machines model in stock price forecasting". In: *Omega* 33.6 (2005), pp. 497–505. ISSN: 0305-0483. DOI: https://doi.org/10.1016/j.omega.2004.07.024. URL: https://www.sciencedirect.com/science/article/pii/S0305048304001082.

Jigar Patel et al. “Predicting stock and stock price index movement using Trend Deterministic Data Preparation and machine learning techniques”. In: *Expert Systems with Applications* 42.1 (2015), pp. 259–268. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2014.07.040. URL: https://www.sciencedirect.com/science/article/pii/S0957417414004473.

F. Pedregosa et al. “Scikit-learn: Machine Learning in Python”. In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

Hadi Rezaei, Hamidreza Faaljou, and Gholamreza Mansourfar. “Stock price prediction using deep learning and frequency decomposition”. In: *Expert Systems with Applications* 169 (2021), p. 114332. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2020.114332. URL: https://www.sciencedirect.com/science/article/pii/S0957417420310228.

Matheus Henrique Dal Molin Ribeiro and L. Coelho. “Ensemble approach based on bagging, boosting and stacking for short-term prediction in agribusiness time series”. In: *Appl. Soft Comput.* 86 (2020).

Arjun Singh Saud and Subarna Shakya. “Analysis of look back period for stock price prediction with RNN variants: A case study on banking sector of NEPSE”. In: *Procedia Computer Science* 167 (2020). International Conference on Computational Intelligence and Data Science, pp. 788–798. ISSN: 1877-0509. DOI: https://doi.org/10.1016/j.procs.2020.03.419. URL: https://www.sciencedirect.com/science/article/pii/S1877050920308851.

Manuel R. Vargas, Beatriz S. L. P. de Lima, and Alexandre G. Evsukoff. “Deep learning for stock market prediction from financial news articles”. In: *2017 IEEE International Conference on Computational Intelligence and Virtual Environments for Measurement Systems and Applications (CIVEMSA)*. 2017, pp. 60–65. DOI: 10.1109/CIVEMSA.2017.7995302.

Gail Weiss, Yoav Goldberg, and Eran Yahav. *On the Practical Computational Power of Finite Precision RNNs for Language Recognition*. 2018. arXiv: 1805.04908 [cs.LG].

G.Peter Zhang. “Time series forecasting using a hybrid ARIMA and neural network model”. In: *Neurocomputing* 50 (2003), pp. 159–175. ISSN: 0925-2312. DOI: https://doi.org/10.1016/S0925-2312(01)00702-0. URL: https://www.sciencedirect.com/science/article/pii/S0925231201007020.