UNIVERSITY OF TARTU
Institute of Computer Science
Computer Science Curriculum

Anders Nõu

Predicting stock return and volatility with machine learning and econometric models — a comparative case study of the Baltic stock market

Bachelor's Thesis (9 ECTS)

Supervisor: Rajesh Sharma, PhD
Supervisor: Mustafa Hakan Eratalay, PhD
Supervisor: Darya Lapitskaya, MA

Tartu 2021

Predicting stock return and volatility with machine learning and econometric models — a comparative case study of the Baltic stock market

Abstract:

Predicting the stock market is a widely researched area of study that is a challenging task. The nature of the problem lies in correctly forecasting the direction and the magnitude of the stock market movement. The severity of the problem exists due to the stock market being impacted by a multitude of factors. There are numerous ways to analyse the stock market and make appropriate investment decisions, but it is challenging to decide the best approach. Here we show which approach is more effective in predicting the returns and volatility of the Baltic stock market: the machine learning or econometric approach. There is a low amount of research on using machine learning or econometric models to predict the Baltic stock market. However, there are no comparative researches that offer a fair comparison between the different approaches for the Baltic stock market. Regarding results, the lowest symmetric mean absolute percentage error for the support vector regression model is 61.90%, and for the autoregressive moving average model, it is 165.43%. The lowest symmetric mean absolute percentage error for GARCH is 51.05%, and for the GARCH-ANN model, it is 61.65%. Overall, the machine learning models outperform the econometric models in most of the evaluated metrics. However, the econometric models' results are comparable to the machine learning models' results in most cases.

Keywords:
machine learning, neural networks, autoregressive moving average, generalized autoregressive conditional heteroskedasticity

CERCS: P170

Aktsia tootluse ja volatiilsuse ennustamine masinõppe ja ökonomeetriliste mudelitega – Balti aktsiaturu võrdlev juhtumianalüüs

Lühikokkuvõte:

Aktsiaturu tootluse ja volatiilsuse ennustamist on laialdaselt uuritud, ent see on osutunud keeruliseks ülesandeks. Probleemi keerukus on tingitud sellest, et aktsiaturgu mõjutavad paljud tegurid. Aktsiaturu analüüsimiseks ja asjakohaste investeerimisotsuste langetamiseks on mitmeid viise, kuid parima lähenemise valimine on keeruline. Siinkohal näitame, milline lähenemisviis on Balti aktsiaturu tootluse ja volatiilsuse ennustamisel efektiivsem: masinõppe või ökonomeetriline lähenemine. Masinõppe ja ökonomeetriliste mudelite kasutamist Balti aktsiaturu tootluse ja volatiilsuse ennustamiseks on uuritud väga vähe. Balti aktsiaturu ennustamise erinevate lähenemisviiside kohta puuduvad ka võrdlevad uuringud, mille alusel saaks võrrelda erinevate meetodite efektiivsust. Töö tulemusena saavutas tugivektori regressioonimudel madalaima sümmeetrilise keskmise absoluutprotsendivea 61,90% ja autoregressiivse libiseva keskmise mudeli puhul tuli

tulemuseks 165,43%. GARCHi mudeli madalaim sümmeetriline keskmine absoluutne protsendiviga on 51,05%, GARCH-ANN mudeli puhul 61,65%. Üldiselt edestavad masinõppemudelid enamiku käsitletud mõõdikute puhul ökonomeetrilisi mudeleid, ent siiski on ökonomeetriliste mudelite tulemused enamikul juhtudel võrreldavad masinõppemudelite tulemustega.

Võtmesõnad:
masinõpe, neurovõrk, autoregressiivne libisev keskmine, üldistatud autoregressiivne tinglik heteroskedastsus

CERCS: P170

Contents

1 Introduction

Investing is an excellent way to grow assets. The first stock exchange opened in the Baltic region after the collapse of the Soviet Union was the Vilnius stock exchange in 1993 [1]. After that, the Riga stock exchange opened in the same year [1]. The Tallinn stock exchange opened in 1995 [1]. The three stock exchanges joined the OMX group during 2000-2004 to form the Nasdaq Baltic [1].

Among Estonians, the interest in investing has grown significantly in recent years, probably due to the popularisation of investment and the favourable trading conditions of banks [2]. More extensive exchanges such as the New York Stock Exchange, the Japan Exchange Group, and the London Stock Exchange use different techniques: automated trading, quantitative analysis, and the use of other econometric models [3, 4]. The research on currently used methods in the Baltic stock market did not show whether investors and traders use automated trading transactions or machine learnings models. Unlike larger markets, there are very few attempts to apply machine learning models in research in the Baltic market. Thus, using machine learning models in the Baltic Stock Exchange is a fresh idea that researchers have not widely applied to the Baltic stock market. Grigoryan [5] has studied the use of neural networks on the example of the Tallink Group – he observed the price of the Tallink Group between March 12, 2012, to December 30, 2014. He evaluated the prediction performance with the mean squared error. Furthermore, in the case of 150 samples, the prediction results mean squared error was 0.0011703, and in the case of 200 samples, the mean squared error was 0.0034567. Lastly, Grigoryan found that the artificial neural network method is effective and efficient in forecasting stock prices [5].

As of April 22, 2021, the main list of the Nasdaq Baltic includes 15 issuers of the Tallinn Stock Exchange, three issuers of the Riga Stock Exchange, and 13 issuers of the Vilnius Stock Exchange – a total of 32 companies [6]. The Baltic secondary list contains 28 companies [6]. Over the past year, the Baltic stock exchange has grown enormously – while in 2019, the total turnover of the stock exchange was approximately 260 million [7], in 2020, the total turnover was approximately 444 million [7]. The OMX Baltic Benchmark index consists of the twenty most significant and most-traded listed companies on the stock exchange [8]. The composition of the index is reviewed by Nasdaq Baltic twice a year [8].

The aim is to give investors (retail investors and financial institutions) and researchers a comparison of the possibilities of using different approaches for the time series analysis of predicting the stock return and volatility of the Baltic stock market. The baltic stock market is studied very little with regards to machine learning and financial models. Grigoryan [5] used the artificial neural network model to predict the daily stock price of the Tallink Group AS (TAL1T). Ercan [9] used the artificial neural network model to OMX predict the Baltic Benchmark GI (OMXBBGI) index value. The study's uniqueness comes from implementing machine learning and financial models and comparing

said models on the Baltic stock market index. The research contributes by offering a foundation to forecasting the Baltic stock market by providing a fair comparison between different approaches and offering future research possibilities to expand the research.

The approach is to apply machine learning, artificial neural networks and econometric models to the Baltic stock exchange. Specifically, the goal is to predict the OMX Baltic Benchmark index's performance by comparing machine learning and artificial neural network models' stock return prediction results with the Autoregressive–moving-average (ARMA) model's prediction results. The second goal is to compare the generalised autoregressive conditional heteroskedasticity (GARCH) model's forecasted volatility with the hybrid model of the generalised autoregressive conditional heteroskedasticity artificial neural network (GARCH-ANN) volatility prediction results.

The dataset used is the OMX Baltic Benchmark Price Index intraday data from 04.09.2001 to 01.03.2021.

Overall, the machine learning models performed better than the ARMA model in the return prediction. As for the volatility prediction, when comparing the GARCH and GARCH-ANN models, the better approach depends on the favoured evaluation metric.

The thesis is organised as follows. The first chapter briefly describes the problem, the goal, and the motivation behind the problem. The second chapter provides an overview of the previous literature – machine learning models used in the past, the markets these models have been used on, and the different approaches of the previous works. The third chapter explains the definitions of return and volatility. The fourth chapter provides an overview of the data used, the time frame, characteristics, processing, and characteristics used. The fifth chapter describes the methodology, i.e., the programming of machine learning models and econometric models. The sixth chapter summarises the results of the different models and compares the machine learning and econometric models. The seventh chapter contains a summary, which describes whether it is reasonable to use machine learning models on the Baltic Stock Exchange instead of econometric models and provides recommendations for further improvements.

2 Related Work

In this chapter, the related works are discussed. The various methods are discussed in Section 2.1. In Section 2.2, stock markets that are used as research targets, are discussed. Section 2.3 describes research that has handled the stock market prediction as a classification problem. Section 2.4 discusses previous works that have taken the stock market prediction as a regression problem.

2.1 Methods

Various methods are used to predict the stock market. Regarding machine learning, many learning methods are used by researchers. For example, the support vector machine is a widely used machine learning algorithm in the time-series forecasts [10–14]. Logistic regression and random forest models are also extensively used in the research [13–15]. Though many kinds of research have implemented the numerous machine learning models, there are even more experimentations with artificial neural networks (ANN), and their various versions [12, 15–18]. Several econometric models are also extensively used to predict the stock price and volatility, like the autoregressive–moving-average and generalised autoregressive conditional heteroscedasticity models [19], [20].

2.2 Stock markets

Since the United States stock market is the largest exchange globally with the combined market cap of Nasdaq US and New York Stock Exchange of over 30 trillion [21], it is one of the most popular research target markets. There are also many kinds of research on other markets like the Japan Stock Exchange, Shanghai Stock Exchange, and India's National Stock Exchange. Few works have also researched the Baltic stock market [5, 9]

2.3 Classification

One approach to the stock market price or return prediction problem is to handle it as a classification problem. Random forest and support vector machine models were used to predict the direction of the stock price of significant companies Samsung, Apple and General Electric, and the Tokyo Stock Exchange index Nikkei 225 [10, 22]. Multiple machine learning models like the kernel factory, AdaBoost, K-nearest neighbours, logistic regression, random forest, and support vector machine were compared on the same dataset to predict the direction of the stock price on the European market [10–12, 14–16, 22]. Among those models, the random forest and support vector machine models were the most accurate [13]. Choudhry and Garg have also researched a hybrid model to predict the price of India Stock Exchange companies Tata Consultancy Services, Infosys, and Reliance Industries [11]. Specifically, a hybrid model of genetic algorithm and support

vector machine. Choudhry and Garg [11] stated that the hybrid model performed better than the standalone support vector machine model. Long short-term memory (LSTM) neural networks, multi-filters neural network (MFNN), computational efficient functional link artificial neural network (CEFLANN), and artificial neural networks (ANN) are used on numerous stock exchanges [12, 15–18].

2.4 Regression

Another approach to the problem is handling it as a regression problem – instead of predicting the direction of the stock price, the exact price or return is predicted. Researches are also widely using the support vector method with the regression problem. In the year 2000, the support vector regression (SVR) model was used to predict the price of IBM, Yahoo, and America Online [23]. A fusion of machine learning algorithms and neural networks like a combination of support vector regression and neural networks, a combination of support vector regression and random forest, and a combination of support vector regression and support vector regression were used on the Bombay Stock exchange [14]. The research [14] concluded that the two-stage hybrid models perform better than the single-stage prediction models. Like the classification problem, researchers use artificial neural networks with the regression problem of stock price or return prediction. Huisu, Jang and Jaewook Lee used Bayesian neural networks to predict the Bitcoin price [24]. Ticknor used a Bayesian regularised artificial neural network to predict the one-day future stock price of individual companies Microsoft and Goldman Sachs [25]. Rounaghi and Zadeh [19] used the autoregressive–moving-average (ARMA) model to forecast the yearly and monthly stock returns of the S&P 500 and the London Stock Exchange. They found that the ARMA model can predict medium or long horizons for the specified stock exchanges. For evaluating the model's performance, Rounaghi and Zadeh used mean absolute error, mean absolute percentage error, median absolute percentage error, symmetric median absolute percentage error and the mean absolute scaled error [19]. Although neural networks are used more widely and might generally perform better than machine learning models, it is still worth researching the different machine learning models' possibilities on the Baltic stock market.

Moreover, the thesis will also focus on predicting the stock return's volatility by composing a hybrid model of the GARCH and ANN model and comparing the results with the standard GARCH model. Oberholzer and Venter [20] applied the GARCH model to the South African stock exchange (JSE). Their dataset consisted of 3326 daily closing values. Oberholzer and Venter used the Akaike information criteria and Schwarz information criteria to evaluate the performance of the GARCH model [20].

3 Background

3.1 Return

The return is the change in the price of an asset, investment, or project over time, represented in the price change or percentage change. A positive return expresses profit, and a negative return represents a loss. In this work, the stock return is the target that the ARMA and machine learning models predict in this study. Figure 1 shows the intraday return of the OMX Baltic Benchmark Price Index from 04.12.2006 to 01.03.2021.

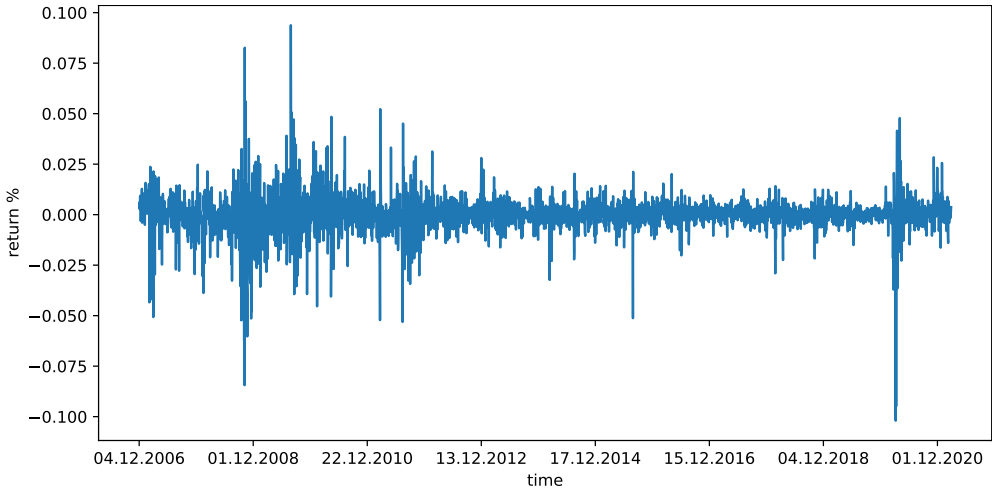

Figure 1. OMX Baltic Benchmark Price Index return

Formula 1 shows the return calculation for an observed period where the Final Value is the final value of the observed period. The Initial Value is the initial value of the observed period.

$$R = \frac{FinalValue - InitialValue}{InitialValue} \tag{1}$$

3.2 Volatility

Volatility is a statistical measure of the dispersion of returns for a given security or index. The higher the volatility, the riskier the security because the price is less predictable. Volatility is measured as either the standard deviation or variance between returns from the same security or market index [26]. There are several ways to find or calculate the volatility of a stock or index.

One option to find the volatility of a security is through a volatility index for a specific market. For example, the VIX index tracks the volatility based on S&P 500 index options. However, an index that tracks the volatility of the Nasdaq Baltic market does not exist.

Another option is to calculate the realised volatility of the index, which is the sum of squared returns of a period. Calculating the daily realised volatility requires intraday data. However, daily intraday data is not available for OMX Baltic Benchmark Index. Therefore an alternative approach is needed. There are several alternative formulas to calculate the estimated volatility. One of them is the Garman and Klass approach [27]. Formula 2 shows the Garman and Klass formula where $H_{it}, L_{it}, O_{it}, C_{it}$ are high, low, open and close values of a security respectively. In the formula, i represents the series, and t represents the time.

$$\begin{aligned}\widetilde{\sigma}_{it}^2 &= 0.511(\log H_{it} - \log L_{it})^2 \\ &- 0.019[(\log C_{it} - \log O_{it})(\log H_{it} + \log L_{it} - 2\log O_{it}) \\ &- 2(\log H_{it} - \log O_{it})(\log L_{it} - \log O_{it})] - 0.383(\log C_{it} - \log O_{it})^2\end{aligned} \tag{2}$$

4 Data

The main goal of choosing the data set is to have as many entries as possible. One possibility is to choose several companies and extract the complete stock information of all those companies. An alternative to choosing individual companies is to choose an index.

Since the Baltic stock market is not very volatile, choosing individual stocks might result in time frames that are too stable regarding the stock returns. Instead, the OMX Baltic Benchmark (OMXBB) index is chosen because it contains the Baltic market's largest and most traded companies. An alternative to the OMX Baltic Benchmark Gross Index is the OMX Baltic 10 index (OMXB10), a tradable index of the ten most actively traded stocks on the Nasdaq Baltic Market [8]. The second alternative to the OMX Baltic Benchmark index is the OMX Baltic All-Share index (OMXB), which follows all the stocks, except when a single shareholder controls at least 90% of the outstanding shares, on the Main and Secondary lists of the Baltic stock market [8].

There are also various index types to choose from - Gross Index (GI), Price Index (PI), Capped Index (CAP). The Gross Index type reflects the stocks' gross return, meaning that the index also tracks the dividends that the stock pays in addition to the tracking price. The Price Index type displays the changes in stock prices [8]. A Capped Index has an upper limit to any single security weight, and if a stock exceeds the limit, its weight will be reduced to the cap [8]. The capped index has two versions: the Price Index (OMXBBCAPPI) and the Gross Index (OMXBBCAPGI) version. The Price Index and Gross Index types are available for all three indexes - OMX Baltic Benchmark, OMX Baltic 10, OMX Baltic All-Share. The Capped Index type is only available for the OMX Baltic Benchmark index. [8, 28]

Due to the availability of the data, the OMX Baltic Benchmark index's variation of Price Index (OMXBBPI) is chosen for this research.

4.1 Data collection

There are several ways to collect data on the stock markets. Because the Baltic stock market is not large, various market data can be challenging to find. One option to collect data is through an API. Many API's do not offer international data with a free subscription. Instead, a paid subscription plan is ordinarily needed to access international data. Thus, gathering data through an API is not a favourable option for this thesis.

An alternative to API-s is the Google Finance functionality in the Google Sheets, which fetches current or historical securities information from Google Finance [29]. Although Google Finance offers an extensive list of attributes about larger stock markets, it has limited available attributes for the Baltic stock market. Google Finance offers the following attributes for the Baltic stock market: open, high, low, close, volume. However, Google Finance does not have data about the OMX Baltic Benchmark volume. Also,

Google Finance offers the Baltic stock market data from 2010. Table 1 gives an example of the Google Finance data of OMX Baltic Benchmark Price Index.

Table 1. Example of Google Finance OMXBBPI data

Date	Open	High	Low	Close	Volume
10/1/2010 16:00:00	329.96	331.25	328.61	330.42	0
10/4/2010 16:00:00	330.33	330.35	327.99	329.16	0
10/5/2010 16:00:00	329.15	330.26	328.6	329.99	0
10/6/2010 16:00:00	329.73	332.92	329.73	332.51	0
10/7/2010 16:00:00	331.8	334.49	331.1	334.33	0

Furthermore, the official Nasdaq Baltic website offers data which contains the following fields for the individual stocks: Date, Average price, Open price, High price, Low price, Close price, Last price, Adjustment factor, Last price adjusted, Change %, Bid, Ask, Trades, Volume, Turnover, Currency, ISIN. According to the index's composition, one solution to gather the index's data is to calculate the index's close, open, high, low values through individual stocks. The historical composition of the OMX Baltic Benchmark index is publicly available through the official market news about the composition changes from the Nasdaq Baltic. However, the early 2000's composition changes news are not public through the Nasdaq Baltic. Therefore, the whole historical composition of the index can not be backtracked through the official market news. The mid-2000's market news does not contain the whole composition of the index, but only the changes. Since the late 2000s, the composition change news also includes the complete list of constituents. Table 2 gives an overview of the Nasdaq Baltic website's individual stock data.

Table 2. Example of Tallink Group's stock data from the Nasdaq Baltic official website [30]

Date	Average price	Open price	High price	Low price	Close price
30.03.2021	0.67170468	0.676	0.678	0.668	0.678
29.03.2021	0.68016006	0.674	0.698	0.67	0.67
26.03.2021	0.67135625	0.678	0.68	0.666	0.676
25.03.2021	0.67453044	0.676	0.676	0.672	0.676
24.03.2021	0.67579414	0.68	0.68	0.672	0.678

Last price	Adjustment	Last price adjusted	Change %	Bid	Ask
0.67	1	0.67	-1.18	0.67	0.67
0.678	1	0.678	1.19	0.678	0.678
0.67	1	0.67	-0.89	0.67	0.67
0.676	1	0.676	0	0.674	0.676
0.676	1	0.676	-0.29	0.676	0.676

Trades	Volume	Turnover	Currency	ISIN
312	203340	136584.43	EUR	EE3100004466
374	328271	223276.822	EUR	EE3100004466
295	176664	118604.48	EUR	EE3100004466
116	46147	31127.556	EUR	EE3100004466
193	80432	54355.474	EUR	EE3100004466

The official Nasdaq Baltic website also offers the complete history of the indexes. The index data offered by the official Nasdaq Baltic website contains the following columns: Ticker/Index, Name, Date, Value, Change %. However, the official website does not offer the close, open, high, low values needed for the volatility formula. Therefore, Nasdaq Baltic does not publicly offer a suitable source of data needed for this research. Table 3 shows an example of data from the official website.

Table 3. Example data of OMXBBPI from the Nasdaq Baltic official website [31]

Ticker / Index	Name30	Date	Value	Change
OMXBBGI	OMX_Baltic_Benchmark_GI	03.01.2000	109.17	9.17
OMXBBGI	OMX_Baltic_Benchmark_GI	04.01.2000	108.26	8.26
OMXBBGI	OMX_Baltic_Benchmark_GI	05.01.2000	104.98	4.98
OMXBBGI	OMX_Baltic_Benchmark_GI	06.01.2000	105.16	5.16
OMXBBGI	OMX_Baltic_Benchmark_GI	07.01.2000	106.43	6.43

Finally, there are also alternative sources that offer various data of the financial

markets. However, not all of them have data of the smaller markets. One such source that offers information about the Baltic stock market is the Investing website [32]. However, the data source mentioned above only offers two variations of the OMX Baltic Benchmark index – OMX Baltic Benchmark Price Index and OMX Baltic Benchmark Capped Price Index. The Investing web portal [32] has close, open, high, low and change % intraday values of the OMX Baltic Benchmark index from 04.09.2001. However, the open, high, and low values are identical to the closing value until 04.12.2006. However, the exact values of the columns as mentioned earlier differ from each other from that date. Table 4 gives an overview of the OMX Baltic Benchmark Price Index data from the Investing website [32].

Table 4. Example of OMXBBPI data from the Investing web portal [32]

Date	Price	Open	High	Low	Change %
10-Sep-01	107.73	107.73	107.73	107.73	-1.44%
07-Sep-01	109.3	109.3	109.3	109.3	-1.28%
06-Sep-01	110.72	110.72	110.72	110.72	-1.07%
05-Sep-01	111.92	111.92	111.92	111.92	-0.37%
04-Sep-01	112.34	112.34	112.34	112.34	0.39%

Due to the number of entries and the available columns to calculate the volatility, the OMX Baltic Benchmark Price Index data from the Investing web portal [32] is chosen for this research.

4.2 Data processing

The chosen Baltic Benchmark Price Index data contains 4935 entries, which start from 04.09.2001 until 01.03.2021. Since the open, low, high values are all equal to the close value from the beginning of the data set until 04.12.2006, the entries can not be used because the correct high, low, open, close values are needed to calculate the volatility. Therefore, the entries before 04.12.2006 will not be used to achieve identical data sets for both comparisons: ARMA and Machine learning comparison and GARCH and GARCH-ANN comparison. A total of 1374 entries are clipped in the process, and a total of 3561 entries are left in the dataset.

The dataset includes the Change % column, which is equal to the daily return. However, the Change % column values have a precision of 4 decimal places in the scientific format. For a more precise value, a new return column is included in the data set. The new column will include the daily returns with a precision of 5 decimal places. Table 5 shows the comparison of the new return values and the Change % values:

Table 5. Comparison of Return and Change % columns

Date	Change %	Return
23.02.2021	0.0019	0.00193
24.02.2021	-0.0002	0.00023
25.02.2021	-0.0011	0.00107
26.02.2021	0.0007	0.00068
01.03.2021	0.0037	0.00372

4.3 Data characteristics

The data set contains 3561 entries and spans between the dates 04.12.2006 and 01.03.2021. The value of the index was 579.21 on 04.12.2006, and the value in 01.03.2021 was 518.03, meaning that the OMX Baltic Benchmark Price Index has not yet reached the value before the global financial crisis of 2007-2008. Figure 2 shows the value of the OMX Baltic Benchmark Price Index between 04.12.2006 and 01.03.2021.

Figure 2. OMX Baltic Benchmark Price Index value

The arithmetic average of the return is -0.000006 (-0.0006%), and the geometric average of return is 0.002999 (0.299%). The median of return is 0.0003 (0.03%). In the observed period, the most significant negative return is -0.10202 (-10.20%), and the most significant return is 0.093790 (9.3790%). There are more positive return days than negative return days in the observed period – 52.60% of the returns are positive, and 47.40% of the returns are negative. The largest volatility cluster occurs in 2007-2009, which was caused by the global financial crisis. Another volatility cluster occurred in 2010-2011, and the latest cluster can be seen in 2020-2021, which was caused by the COVID-19 pandemic.

4.4 Features

Features are a vital component for econometric and machine learning models. They are the foundation on what the predictions are made. Generally, only one feature is used for the ARMA and GARCH models. In this case, the return is used as the feature for the models. Although, additional exogenous variables can be included for both models. However, exogenous variables will not be included in the econometric models in this thesis.

For machine learning, several features are often used for fitting the models. Several features are considered for the machine learning models. First, we consider n number of previous returns, where n is the number of previous days of the data record. We will consider a minimum of 1 and a maximum of 30 for the value of n. Secondly, high, low, close and open values are considered as features. The high and low values are the financial instruments' highest and lowest price of a specific period respectively. The open value is the instruments' price at the start of the trading day. The close value is the instruments' price at the end of the trading day.

5 Methodology

In this section, we first introduce the time series sliding window method (Section 5.1). Then we introduce the econometric models that are researched (Section 5.2). Finally, we discuss the random forest, support vector machine and k-nearest neighbours machine learning methods (Section 5.3).

5.1 Sliding window method

The usual procedure for machine learning involves shuffling the data and then splitting the dataset to train and test sets. However, when dealing with a time series problem, the dataset can not be split because the order of observations is essential. Therefore, to perform cross-validation methods when dealing with a time series problem, the sliding window method can be used [33].

There are at least two variations of the sliding window techniques used for the time series problems [33]. One variation is the nested method, where after each iteration, the test set is added to the training size, and then a new test size is a set following the previous test set. However, this method will not be used in this thesis.

The second variation of the sliding window method is when the training and test sets are of constant size throughout the cross-validation process. After each iteration, the test data is added to the training data, and a chunk of the size of the test data is removed from the start of the training set. A visualisation of the process is seen in Figure 3.

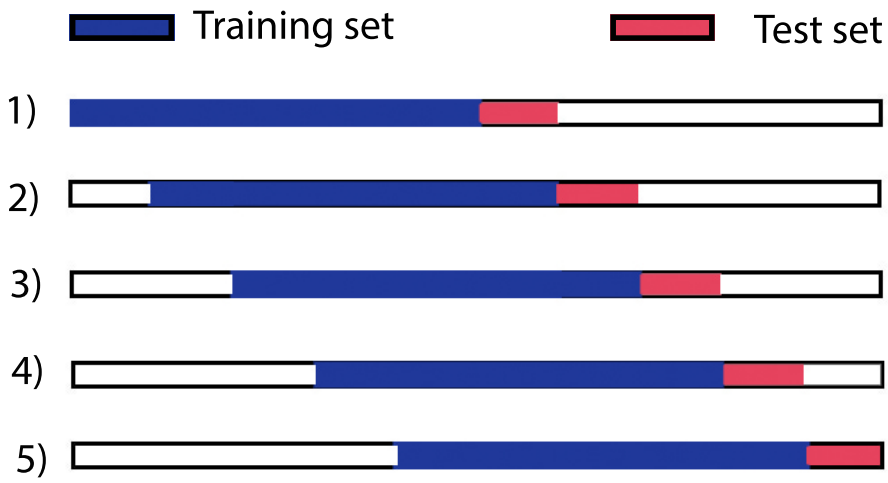

Figure 3. Sliding window method

The sliding window technique also shows whether it is more beneficial to use either the whole dataset or a smaller amount of data to forecast the future values.

The sliding window method is carried out for all the models – ARMA, machine learning models, GARCH and GARCH-ANN. The training size of the window is 1000, 500, 365, 300 or 100. The test size is 5 or 10 to simulate short-term forecasting.

5.2 Econometric models

5.2.1 ARMA

The autoregressive moving average (ARMA) model is widely used in statistical analysis of time series that was introduced in 1951 by Whittle[34]. The model is a combination of the autoregressive (AR) and moving average (MA) model.

The ARMA(p,q) model has two parameters: the order p of the AR(p) model and the order q of the MA(q) model. The orders of p and q represent the number of immediate previous values that are used to predict the value of the present time [35, 36].

A requirement for the ARMA model is that the observed time series has to be stationary. Stationary time series is a series with properties that are not dependent on the time the series is observed [37]. The augmented Dickey-Fuller test is a method to determine whether the observed time series is stationary or not. If the series is not stationary, then a phase of detrending is needed to be carried out to obtain a stationary time series [38]. According to the Augmented Dickey-Fuller test, the observed OMX Baltic Benchmark Price Index return data is determined to be stationary. Thus no further action is required regarding the stationarity of the time series. The python statsmodels library's function adfuller is used to test for stationarity [39, 40].

For the ARMA model, optimal p and q have to be chosen. There are several methods to do so. The most optimal and widespread method is to calculate the autocorrelation and partial autocorrelation for the time series. The partial autocorrelation determines the optimal order of p, and the autocorrelation determines the order of q. The respective plots for partial autocorrelation and autocorrelation lags are in Figure 4 and Figure 5 respectively. The figures show the autocorrelation or partial autocorrelation for each displayed lag.

For partial autocorrelation and autocorrelation, there is a sharp drop after one lag. Thus, the most optimal orders of p and q are one according to the plots. However, in both plots, a considerable autocorrelation can be observed for lags between 2 and 14, with some exceptions. In both cases, the correlation drops off, with some exceptions of small spikes for some lags. Pmdarima library for Python offers a function called auto_arima to determine the optimal orders of p and q for the ARMA model [41]. The pmdarima's auto_arima function determined that the best model is ARMA(3,3) based on the default information criterion Akaike information criterion.

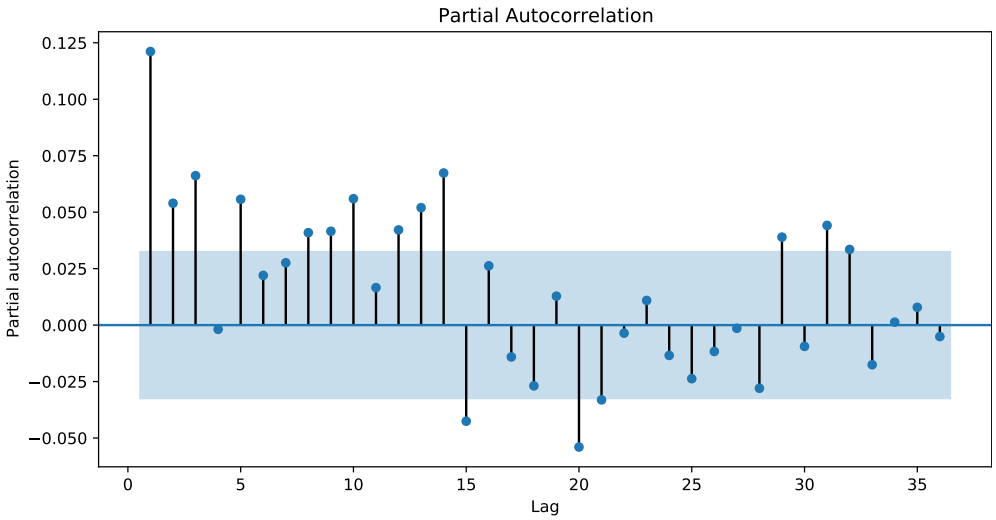

Figure 4. Partial autocorrelation of OMXBBPI time series

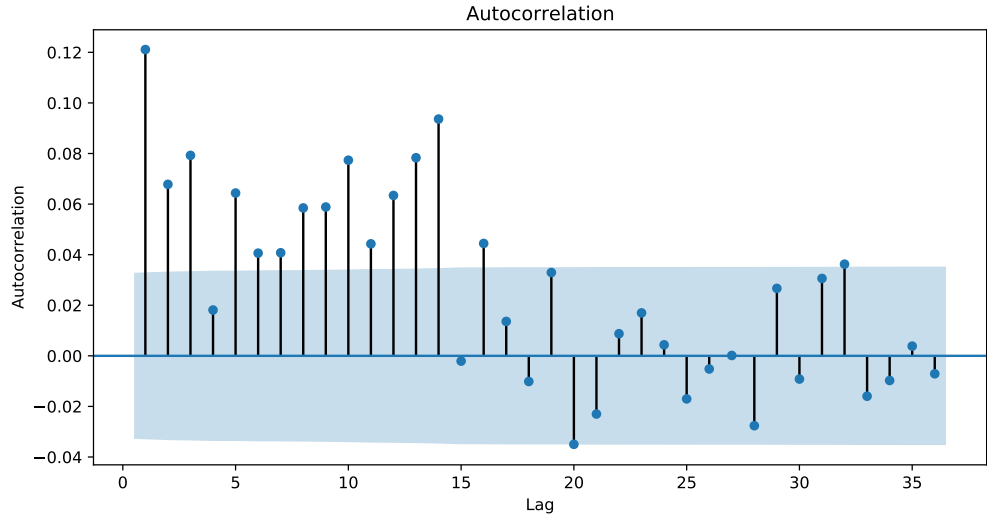

Figure 5. Autocorrelation of OMXBBPI time series

Since the partial autocorrelation and autocorrelation method and auto_arima function offered models with different orders of p and q (ARMA(1,1) and ARMA(3,3) respectively), it is also reasonable to test different sets of orders of p and q. Thus, a test of different sets of p and q is carried out. For a reasonable amount of tests, values of p and q are a minimum of 1 and a maximum of 5 each, which means that a total of 25 tests is carried out regarding orders of p and q.

After the best orders of p and q are determined, the best performing model is compared with the best performing machine learning models. Furthermore, the outperforming

ARMA model is applied for the sliding window method. The sliding window method results of ARMA are then compared with the machine learning results of the sliding window.

5.2.2 GARCH

The autoregressive conditional heteroskedasticity (ARCH) model is a statistical model used in time series forecasting [42]. The ARCH model describes the variance of the error [42]. Engle first introduced the model in 1982 [42]. Bollerslev generalised the ARCH model in 1986 to allow for past conditional variances [43]. Thus, became the generalised autoregressive conditionally heteroscedastic (GARCH) model.

The GARCH(p,q) has two parameters: the orders of p and q. Similar to the ARMA(p,q) parameters described in Section 5.2.1.

For the GARCH model, similar tests to the ARMA model (described in Section 5.2.1) are carried out to have a fair comparison. First, the optimal orders of p and q are tested. The minimum value of p and q is one, and the maximum is five. Thus, a total of 25 tests are carried out to determine the optimal orders of p and q for the GARCH(p,q) model. The most efficient GARCH(p,q) model is also used for the GARCH-ANN hybrid model.

After determining the optimal orders of p and q, the corresponding GARCH model is used to implement the sliding window method. Various training sizes and test sizes are tested in the sliding window test. Then, the optimal training size and test size is determined.

Finally, the GARCH model is compared with the GARCH-ANN model. First, the most optimal GARCH(p,q) model is compared with the best performing GARCH-ANN model with a 70%/30% split and then compared in the sliding window method. The 70%/30% split is used for the GARCH model because testing the different split sizes determined that the 70%/30% split size is the most optimal.

5.3 Machine learning models

This thesis includes implementing three different machine learning procedures to predict the return of the Baltic stock market. The random forest, support vector machine (SVM) and K-nearest neighbours (KNN) methods are used. The performance of the models is measured by five metrics: MAE, MAPE, sMAPE, MSE and RMSE. The machine learning library scikit-learn for Python is used to create the addressed machine learning models.

For all three learning methods (random forest, support vector regression, k-nearest neighbours), multiple models are created, and a set of tests are carried out. In many cases, the dataset is first shuffled and then split into training and datasets. Although, shuffling can not be performed in a time series problem because the order of the inputs is essential. A 70%/30% training and test split is used in all cases because testing different split sizes concluded that the 70%/30% split yields the best results. Afterwards, fitting the model is done, and then the fitted model is used to predict the return. The predictions are then compared with the actual values, and the errors are calculated.

The overall testing and optimisation procedure consists of three parts. First, the optimal features are chosen for each model. Two sets of features are tested:

1) n number of previous returns

2) n number of previous returns, and high, low, close and open values

Afterwards, the best performing feature set is chosen for each model. Secondly, parameter optimisation is carried out to maximise the effectiveness of the models. Scikit-learn's GridSearchCV class is used for the parameter optimisation [44]. Thirdly, the sliding window method is implemented for each model to measure the effects of adjusting the size of the training size and the range of the prediction size. Finally, the machine learning methods are compared to determine the best model in the standard method and the most efficient model in the sliding window method. Afterwards, the most efficient machine learning model is compared with the ARMA model.

5.3.1 Random forest

Random forest is an algorithm proposed by Breiman in 2001 [45]. The algorithm consists of creating several trees that each cast a vote for the most popular class. Essentially, the random forest consists of several decision trees that use randomly selected inputs. Figure 6 shows an example of several decision trees (that the random forest consists of), where the red leaf node for each tree shows the classification of each decision tree.

The supervised learning method is renowned because of its' accuracy and ability to handle large datasets [46]. Also, the learning method proves to be reliable in a variety of prediction problems [46]. A random forest classifier can be used for the classifier problems, and a random forest regressor can be used for the regression problems. In this thesis, the regressor version is used. Scikit-learn for Python offers sklearn modules'

RandomForestRegressor class with a moderate range of modifiable parameters [47].

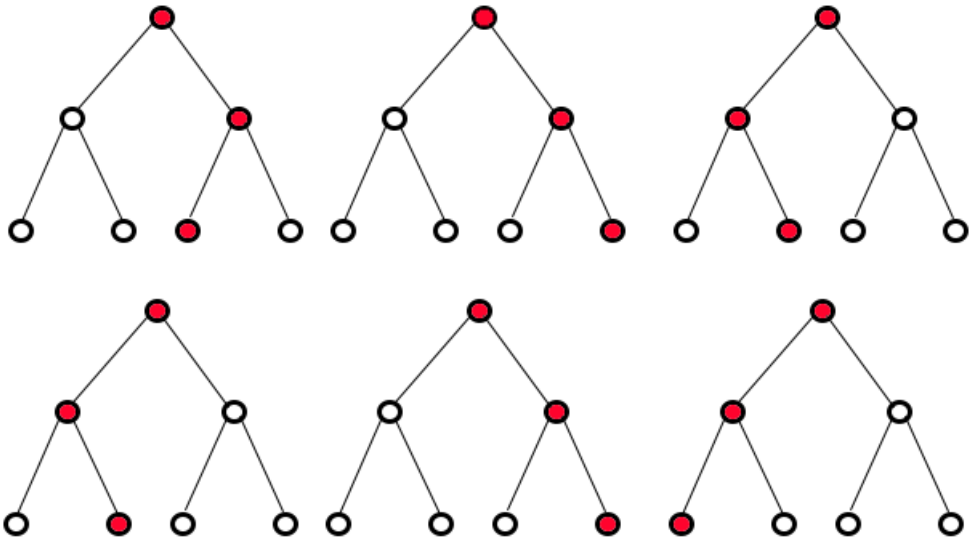

Figure 6. Random forest learning method decision trees

5.3.2 Support vector regression

Support vector regression (SVR) is a supervised learning model based on the Vapnik-Chervonenkis theory [48]. SVR is used for the regression problem; meanwhile, the support vector machine (SVM) is the analogue version of the SVR technique used for classification problems. The general idea of the SVM is to find a hyperplane in a multidimensional space that separates the classification targets [49]. For a human, it is usually easy to separate the targets in a one-dimensional or two-dimensional space when dealing with somewhat clearly separated targets [49]. However, as the targets are more complexly spread, then it is harder for a human to draw a separating hyperplane, especially when the separator exists in a high-dimensional space. This is where the support vector technique comes to aid.

5.3.3 K-nearest neighbours

The K-nearest neighbours (KNN) is an algorithm that predicts the target value based on the k nearest neighbours of the observed features [50, 51]. The prediction is decided on a majority vote, which is similar to the random forest decision tree forecasting explained in Section 5.3.1. The KNN method can be applied for classification, and regression problems [51, 52]. The KNN is described as a simple and intuitive but low-efficiency learning method [50, 52]. The effectiveness of the model is also heavily dependent on

an optimal value of k. The scikit-learn library's KNeighborsRegressor class is used to construct a KNN regressor model in this research [53]. The default value of k in the KNeighborsRegressor class is five [53].

5.4 GARCH-ANN

GARCH-ANN is a hybrid model composed of the GARCH model (described in Section 5.2.2 and an artificial neural network (ANN) model. The general idea of this hybrid model is the following: first, the GARCH model is created and fitted. Secondly, the predictions of the previously constructed GARCH model is used to fit the ANN model. Finally, the fitted ANN model is used to predict the target value, and the target is then evaluated with the actual value.

The Keras library for Python is used to apply the TensorFlow ANN implementation [54]. The Keras library is used to simplify the TensorFlow implementation process [55].

As for the specific ANN implementation model, first, several ANN structures are tested. The model's structure optimisation consists of modifying the number of layers and the dimensionality of the layer's output space [56]. The Dropout layer, which assists in avoiding overfitting, is also included in some cases [57]. For ANN structure testing, the default optimiser is used (RMSprop [58]), and no layer activation functions are used.

The activation layer functions and optimisers are cross-tested to find the best combination of the layer activation functions and the optimiser for the ANN model. The Keras library offers nine layer activation functions: relu, sigmoid, softmax, softplus, softsign, tanh, selu, elu, exponential [59]. The Keras library offers eight optimisers: SGD, RMSprop, Adam, Adadelta, Adagrad, Adamax, Nadam and Ftrl [60]. The cross-testing of optimisers and layer activation function consists of a total of 72 tests.

Finally, the best performing ANN structure, layer activation function and optimiser combination are used for the sliding window method.

5.5 Standardised residuals

Standardised residuals show the strength of the difference between the observed value and the actual value [61]. The goal of standardised residuals is to compare the general effectiveness between using the econometric models and using the machine learning models. The closer to zero the standardised residuals is, the greater the strength. The absolute values of standardised residuals are taken. The mean absolute value is then calculated to compare the strength of the econometric models and the machine learning models.

Two sets of standardised residuals are calculated. Formula 3 is used to calculate the standardised residuals for econometric models.

$$\frac{\text{ARMA forecast error}}{\sqrt{\text{GARCH volatility forecast}}} \tag{3}$$

Formula 4 shows how to calculate the standardised residuals for machine learning models.

$$\frac{\text{Machine learning forecast error}}{\sqrt{\text{GARCH-ANN volatility forecast}}} \tag{4}$$

Standardised residual is calculated for the standard method and the sliding window method for machine learning and econometric models. Regarding sliding window method parameters, a training size of 1000 and a test size of 5 is chosen. The standardised residuals are calculated for econometric models and all three machine learning methods: random forest, SVR, and KNN. The mean absolute of standardised residuals is calculated to compare the different approaches and methods.

6 Evaluation

In this chapter, we first discuss various evaluation metrics (Section 6.1) and then present results of the evaluation (Section 6.2). In Section 5.5, the standardised residuals results for econometric and machine learning models are discussed.

6.1 Metrics

In this section, the evaluation metrics that are used to compare the performance of the models, are described. A total of five metrics are used: mean absolute error (Section 6.1.1), mean absolute percentage error (Section 6.1.2), symmetric mean absolute percentage error (Section 6.1.3), mean squared error (Section 6.1.4) and the root mean square error (Section 6.1.5).

6.1.1 Mean absolute error

Mean absolute error (MAE) is a statistical metric used to measure a model's performance [62]. MAE is also often compared with root mean square error and determined to be a better metric than the last-mentioned [62, 63]. Essentially, MAE is an arithmetic average of the absolute errors. Formula 5 shows the calculation of MAE where y_i is the prediction value, x_i is the actual value, and n denotes the number of errors [62].

$$\frac{1}{n}\sum_{i=1}^{n}|y_i - x_i| \tag{5}$$

6.1.2 Mean absolute percentage error

Mean absolute percentage error (MAPE) is a widely used metric in different fields like finance, electricity consumption, and retail [64]. MAPE offers an interpretation of the relative error, which helps to evaluate the results better. Also, due to MAPE construing a relative error, the results can be compared with models created for other markets.

One shortcoming of MAPE is that the metric's maximum range is infinite [65] – this means that significant individual outlying errors can lead to a massive average error. However, the most significant shortcoming of MAPE is considered to be the vulnerability to zero values [65]. When the actual value is zero, then the absolute percentage error is undefined due to division by zero [65]. There are many propositions to solve the issue [65]. One of the propositions is to exclude the zero values [65]. However, excluding the zero values might cause a misrepresentation of the data. Another problem of MAPE is that the errors above the actual value lead to a more significant absolute percentage error than an error under the actual value [66]. The symmetric mean absolute percentage fixes the problem [66, 67].

Formula 6 shows the calculation of MAPE where y_i is the prediction value, x_i is the actual value, and n denotes the number of errors [65].

$$\frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - x_i}{y_i}\right| \tag{6}$$

6.1.3 Symmetric mean absolute percentage error

Armstrong [67] first introduced the symmetric mean absolute percentage error (sMAPE) as shown in Formula 7 where the F_t is the forecasted value, A_t is the actual value, and n is the is the number of observed errors [67].

$$1/n\sum_{t=1}^{n}\frac{|F_t - A_t|}{(A_t + F_t)/2} \tag{7}$$

The advantage of sMAPE is that its' value is constrained. The bounds of the sMAPE is [0, 200%]. The maximum value of the range is 200% because of the division by two in the denominator. Also, the over predictions do not yield higher errors than the under predictions, unlike the MAPE. However, Flores [68] proposed another version of sMAPE in 1986 to change the range of the sMAPE to [0, 100%]. Formula 8 shows the sMAPE version proposed by Flores where the F_t is the forecasted value, A_t is the actual value, and n is the number of observed errors [68]. Unlike in Armstrong's proposed sMAPE version, the division by two is not used in the denominator, which is the cause of the reduced range.

$$1/n\sum_{t=1}^{n}\frac{|F_t - A_t|}{(A_t + F_t)} \tag{8}$$

One disadvanatage of Armstrong's proposed definition is however, that the percentage error can be undefined if $A_t + F_t = 0$, due to division by zero. In addition, there is a possibility of division by zero if $|A_t|+|F_t|= 0$. Another disadvantage is that the error can be negative if $A_t + F_t < 0$.

Armstrong's proposed definition is used to calculate the sMAPE in this thesis.

6.1.4 Mean squared error

The mean squared error is an evaluation metric that presents the quantitative level of error or similarity between two values [69]. The range of MSE is [0, ∞]. There are several advantages and disadvantages of the MSE metric. The first advantage of MSE is the simplicity – it is easy and cheap to calculate [69]. Also, the MSE has several properties that offer a clear and direct interpretation of the metric [69]. The first property is the

symmetric property of equality where $(x_i - y_i)^2 = (y_i - x_i)^2$, which shows that over-predicted and under-predicted forecasts have the same weight [69]. Another property is the identity property where $(x_i - y_i)^2 = 0$ if x = y shows that if both observed values x and y are equal, then the error is 0 [69]. MSE also preserves the unit of measurement, which is in some cases desired and useful.

On the contrary, in some cases, the original unit of measurement can offer a vague interpretation of the measurement if there is not a clear understanding of the data set. Another disadvantage is the squared error's vulnerability to outliers [70]. Outliers can cause high error values, which in return might skew the mean of the squared errors [70]. High error values are especially problematic when there is a low number of observed errors – then the outliers lead to a massive mean error.

Formula 9 shows the calculation of the MSE where x is the actual value, and y is the predicted value [69].

$$\frac{1}{N}\sum_{i=1}^{n}(x_i - y_i)^2 \tag{9}$$

6.1.5 Root mean square error

Root mean square error (RMSE) is the square root of the MSE metric. Formula 10 shows the calculation of RMSE where x is the actual value, and y is the predicted value.

$$\sqrt{\frac{1}{N}\sum_{i=1}^{n}(x_i - y_i)^2} \tag{10}$$

The scale of the RMSE is [0, ∞] and is presented in the same unit of measurement as the observed values of x and y in the formula. RMSE is often compared with MAE [62, 63]. The RMSE penalizes larger errors more than the smaller errors [62]. Thus, it depends on the nature of the problem, whether it is reasonable to evaluate the results with the RMSE metric.

6.2 Results

For testing and comparing the models, the ARMA and machine learning models are compared. The GARCH and GARCH-ANN models are compared on their own. Finally, the standardised residuals of ARMA-Machine learning and GARCH-GARCHANN are compared.

Regarding ARMA, the model results are compared with different orders of p and q. The ARMA model is then applied with the sliding window method to determine optimal training size and test size for the sliding window method with ARMA.

For machine learning, first, the optimal set of features is chosen. Then, the models' parameters are optimised and compared with the default parameter models. Furthermore, each machine learning model is applied for the sliding window method. Furthermore, the best performing machine learning method is determined regarding the standard method and the sliding window method.

Finally, the ARMA and machine learning models are compared in the standard and sliding window method to determine which model is more efficient in predicting the OMX Baltic Benchmark Price Index return.

Regarding GARCH, first, the different orders of p and q are tested to determine the optimal orders for the GARCH model. Secondly, the GARCH model is applied with the sliding window method with the optimal p and q determined from the first test. Finally, the optimal training size and test size is elected from the sliding window test.

For GARCH-ANN, the best performing GARCH model's (which is determined in the GARCH testing) forecasts are used as the neural network model input to determine the optimal neural network structure, optimiser, layer activation function and epoch size. Secondly, the optimal neural network structure and parameters are determined to construct a GARCH-ANN model for the sliding window method. Finally, the best training size and test size of the sliding window method is chosen.

After the best performing GARCH and GARCH-ANN models are determined, the models are first compared in the tests performed on the complete data set, and then finally in the sliding window method. Finally, based on the comparisons, it is determined which model is more efficient in predicting the volatility of the Baltic stock market.

6.3 ARMA

For ARMA, to find out the optimal orders of p and q for AR(p) and MA(q), a series of tests are carried out. A total of 25 tests with different sets of p and q are performed. The minimum of both p and q is 1. Otherwise, if p is 0, the model would be a MA(q) model and vice versa. Table 27 gives an overview of the aforementioned tests.

Although the autocorrelation and partial autocorrelation plots indicated the best model to be ARMA(1,1) and the auto_arima function determined that the ARMA(3,3) is the best model, neither ARMA(1,1) or ARMA(3,3) did outperform the other models in any of the five metrics. On the contrary, in regards to sMAPE, the ARMA(4,5) model outperformed the others. ARMA(2,1) model exceeded others in regards to the MAPE metric. Among the MAE results, ARMA(4,4) is the best. ARMA(3,2) model dominated in the MSE and RMSE metric. The best performing models among different metrics and the models stated to be the best performers by the correlation plots and auto_arima are displayed in Table 6.

Table 6. ARMA best performing models and the models stated to be best

p	q	MAE	MAPE	sMAPE	MSE	RMSE
1	1	0.003876	126.187%	167.512%	0.00005198	0.0072
2	1	0.003871	124.312%	168.777%	0.00005161	0.0072
3	2	0.003866	125.766%	165.329%	0.00005112	0.0071
3	3	0.003879	137.051%	161.123%	0.00005211	0.0072
4	4	0.003848	128.991%	162.887%	0.00005128	0.0072
4	5	0.003905	167.689%	155.925%	0.0000549	0.0074

Based on the MAPE metric, the ARMA(2,1) model outdid the other ARMA models. Thus, the ARMA(2,1) is chosen for the sliding window method. When considering MAPE, ARMA(2,1) is the most effective with the sliding method when the training size is 1000, and the test size is 5. The model achieved an average result of MAPE 119.845%. ARMA(2,1) exceeded the previous method that used the whole data set by approximately 5% in regards to MAPE. On the contrary, the sMAPE result of the sliding window method is inferior by about 4%. The model is also impressive when the training size is 1000, and the test size is 5, with the MAPE of 122.683% and sMAPE of 171.973%. Also, when the training size is 1000, the MSE is almost equal to the MSE of the standard method. However, with a smaller training size, the MSE results of the sliding window are generally worse. The average RMSE of the sliding window method exceeded the RMSE of the standard method in terms of benefit. The standard method's MAE is lower than the sliding window method's MAE – 0.003876 against the lowest MAE of the sliding window method of 0.004323.

In conclusion, the method's efficiency is highly based on the metric that the methods are evaluated with. When MAPE or RMSE is preferred, then it is more effective to use the sliding window method. On the other hand, when MAE or sMAPE is preferred for evaluation, the standard method is more favourable.

Table 7. ARMA(2,1) sliding window results

Training size	Test size	MAE	MAPE	sMAPE	MSE	RMSE
1000	5	0.004323	122.683%	171.973%	0.000052	0.005268
500	5	0.005277	132.907%	170.189%	0.000077	0.006416
365	5	0.005718	131.661%	165.430%	0.000101	0.006930
300	5	0.005699	144.047%	162.609%	0.000097	0.006887
100	5	0.005774	156.280%	159.968%	0.000096	0.006991
1000	10	0.004326	119.845%	172.934%	0.000052	0.005523
500	10	0.005292	130.805%	171.219%	0.000078	0.006741
365	10	0.005672	126.463%	166.578%	0.000094	0.007195
300	10	0.005753	145.816%	164.120%	0.000099	0.007241
100	10	0.005817	157.229%	160.082%	0.000098	0.007348

6.4 Machine learning

For different machine learning models, identical data sets and features are used to display a fair comparison between them. First, the optimal features are evaluated for each machine learning method. Subsequently, the best set of features is chosen. Then, for each method, the models with default parameters and optimised parameters are compared. Afterwards, the best set of features and best set of model parameters is chosen to create a sliding window method. The best training size and test size is determined for each model. In the end, the best model is determined to be compared with the best ARMA model in terms of the standard method and the sliding window method.

6.4.1 Random forest

Random forest's first model optimisation is the testing of the features. Initially, the model is tested with n number of previous returns, where the minimum value of n is one, and the maximum n is 30. Table 29 gives an overview of the number of previous returns feature testing. As the number of previous returns as features grow, the MAE, MAPE, MSE and RMSE decrease. On the other hand, as the number of previous returns increase, the sMAPE value increases. An immense difference in MAPE can be seen between the results of 1 previous return and 30 previous returns – 469.225% and 151.448%, respectively. As shown on Figure 7, 8, 10, the MAE, MAPE and MSE have exponential decay.

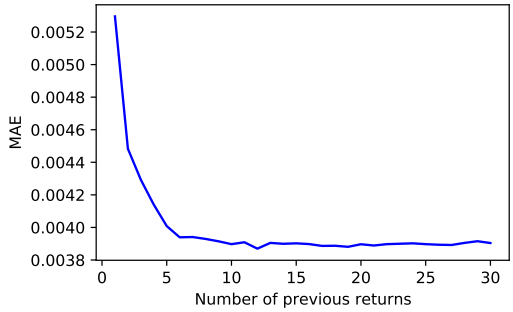

Figure 7. RF nr of previous returns MAE results

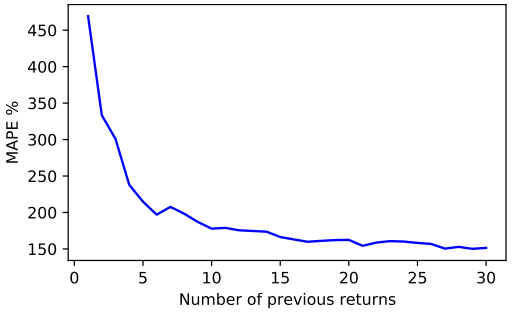

Figure 8. RF nr of previous returns MAPE results

Figure 9. RF nr of previous returns sMAPE results

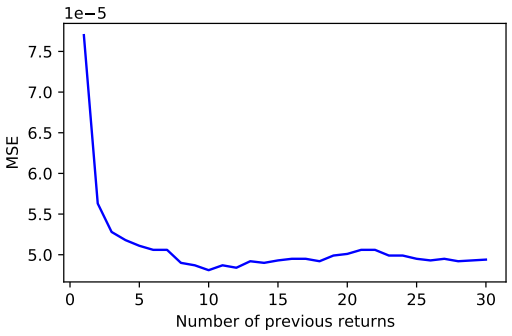

Figure 10. RF nr of previous returns MSE results

The second set of model feature tests also included the high, low, open and close values and the n number of previous returns. Table 30 gives a numerical overview of the results. Similarly to the previous feature tests, the MAE and MAPE also have an exponential decay as seen on Figures 11, 12. However, unlike in the previous set of tests, MSE initially has an exponential decay but has a spike between 15 and 25 number of returns as seen in Figure 14. Figure 13 shows an analogous growth of sMAPE to the previous tests.

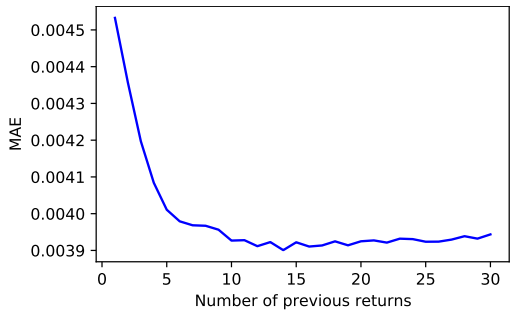

Figure 11. RF all features MAE results

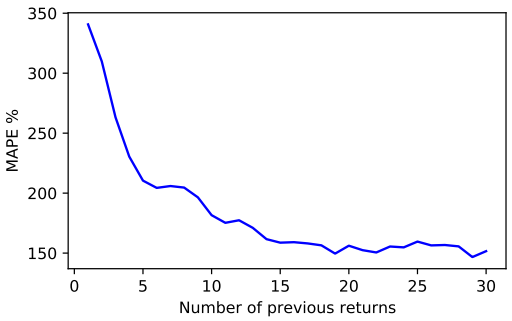

Figure 12. RF all features MAPE results

Figure 13. RF all features sMAPE results

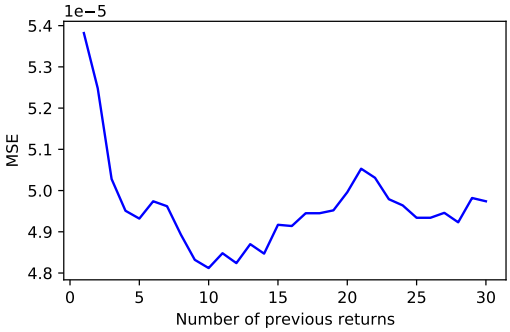

Figure 14. RF all features MSE results

To conclude the feature testing, the number of previous returns has an extensive impact on the results. The MAE, MAPE and MSE metric values decrease substantially as the number of previous returns grows. On the contrary, the value of sMAPE increases as the number of previous returns rises. However, as the MAE, MAPE and MSE values decreased exponentially, it is more reasonable to choose a high number of previous returns if the metrics are valued equally. If the sMAPE is considered more substantial, it is more feasible to favour a low number of previous returns. Regarding high, low, open and close features, if the low number of previous returns is preferred, it is more efficient to include the extra features. If a high number of previous returns is chosen, then the high, low, open and close values do not appear to be meaningful because the results are similar. For further random forest model optimisation and tests, both options of 1 previous return and 30 previous returns are studied.

Further optimisation of the random forest model includes the model's parameter optimisation. Table 8 and 9 shows the difference of using default parameters and optimised parameters for the model. With one previous returns, there is a massive increase in effectiveness regarding MAE and MAPE. MSE and RMSE values are fairly similar. However, the model's effectiveness suffered concerning sMAPE. With 30 previous returns, the outcome is similar to the only one previous return example, except for MAE and MAPE not shrinking as much. In conclusion, the optimisation technique did not favour the sMAPE metric but is very beneficial with MAE and MAPE metrics.

Table 8. Random forest results with default parameters

N prev. returns	MAE	MAPE	sMAPE	MSE	RMSE
1	0.004532	340.681%	142.120%	0.000053	0.0073
30	0.003943	151.616%	162.270%	0.000049	0.0071

Table 9. Random forest results with optimised parameters

N prev. returns	MAE	MAPE	sMAPE	MSE	RMSE
1	0.003888	114.516%	170.743%	0.000054	0.0073
30	0.003872	109.421%	182.709%	0.000051	0.0071

The sliding window method is performed with two different sets of features – 30 previous returns, high, low, close, open values, and one previous return, high, low, close, open values. The overview of the results are seen in Table 11 and 10. With both sets of features, the metric value patterns are quite identical. A large set of training data set is beneficial for the MAE, MAPE, MSE and RMSE metrics in both cases. However, the metrics mentioned above values increased as the training size increased. On the contrary, a small training size is favourable for the sMAPE. The test size did not make a significant impact on the results for either set of features. The results did not differ significantly between the two tests when the training size is at its highest value. The sliding window method results align with the results of the feature testing and parameter optimisation testing. Thirty previous returns as features yielded better outcome than one previous return with MAE, MAPE, MSE, RMSE, and one previous return yielded better results regarding sMAPE.

Table 10. Random forest sliding window results with 1 previous return, high, low, close and open values as features

Training size	Test size	MAE	MAPE	sMAPE	MSE	RMSE
1000	5	0.0042292	136.707%	158.989%	0.0000503	0.0054251
500	5	0.0051478	171.140%	147.707%	0.0000797	0.0063650
365	5	0.0056279	183.895%	143.967%	0.0001036	0.0069238
300	5	0.0056305	195.862%	141.303%	0.0001015	0.0069124
100	5	0.0057843	229.950%	136.660%	0.0001015	0.0070492
1000	10	0.0042292	136.708%	158.990%	0.0000503	0.0054251
500	10	0.0052239	177.779%	147.516%	0.0000816	0.0067373
365	10	0.0056816	195.423%	144.304%	0.0000995	0.0072991
300	10	0.0057086	195.899%	140.693%	0.0001041	0.0073129
100	10	0.0058286	228.130%	136.958%	0.0001032	0.0074374

Table 11. Random forest sliding window results with 30 previous return, high, low, close and open values as features

Training size	Test size	MAE	MAPE	sMAPE	MSE	RMSE
1000	5	0.004331981	136.487%	165.986%	0.0000526	0.0052887
500	5	0.005235231	160.208%	160.844%	0.0000773	0.0064205
365	5	0.005786871	172.378%	158.141%	0.0000994	0.0070673
300	5	0.005771790	182.862%	155.889%	0.0000980	0.0070461
100	5	0.005938070	209.415%	150.579%	0.0000992	0.0072083
1000	10	0.004367577	137.181%	165.987%	0.0000543	0.0055787
500	10	0.005272594	162.311%	161.058%	0.0000789	0.0067350
365	10	0.005809639	167.753%	158.240%	0.0000989	0.0074285
300	10	0.005823662	185.551%	156.302%	0.0000996	0.0073638
100	10	0.005968445	208.415%	150.319%	0.0001008	0.0075318

6.4.2 Support vector regression

For SVR models, the same set of tests are carried out as for the random forest models. Table 31, 32 give an overview of the values of the feature testing with SVR default parameters. As seen on Figures 16, 18, 20, 29, 31, as the number of previous returns increases gradually, the errors increase. Thus, the optimal number of returns for SVR with all the features is 1. , for the feature set including only the previous returns, the errors start relatively small and then spike up between 2 and 5 previous returns. However, after the spike, the errors decrease and are somewhat stable until 30 previous returns. Contrary to the random forest models, using high, low, open, close features does not yield better results than without them. Also, the sMAPE value is positively correlated with other metrics, unlike in the random forest model, where the sMAPE metric is negatively correlated with other metrics. In conclusion, in both cases, it is optimal to use only one previous return. However, using only previous returns is more efficient for the SVR model.

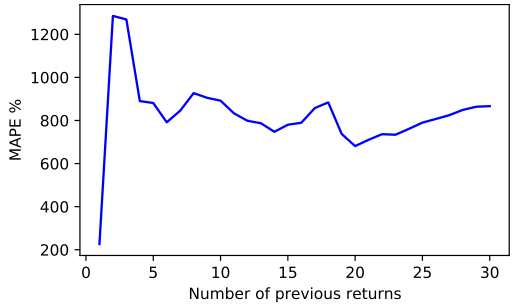

Figure 15. SVR nr of previous returns MAPE results

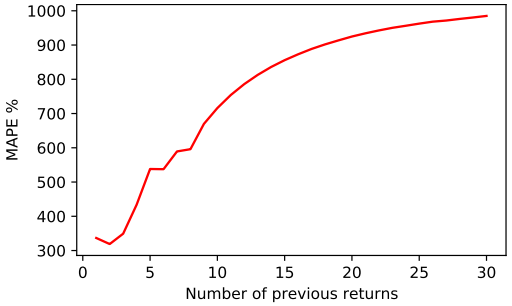

Figure 16. SVR all features MAPE results

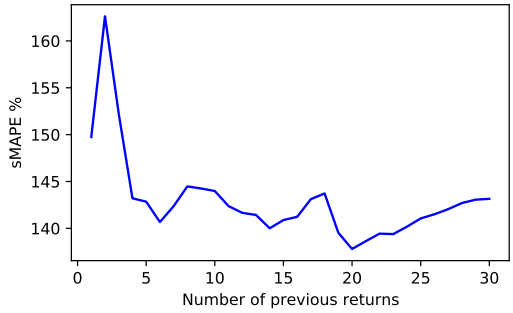

Figure 17. SVR nr of previous returns sMAPE results

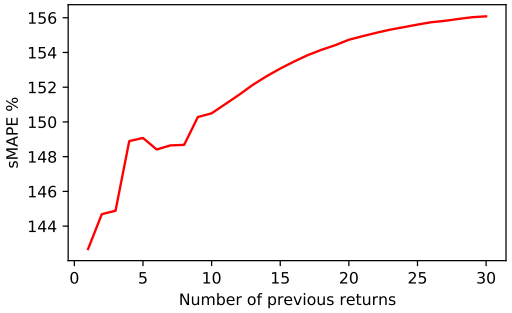

Figure 18. SVR all features sMAPE results

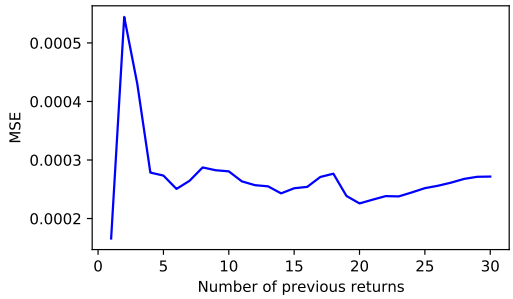

Figure 19. SVR nr of previous returns MSE results

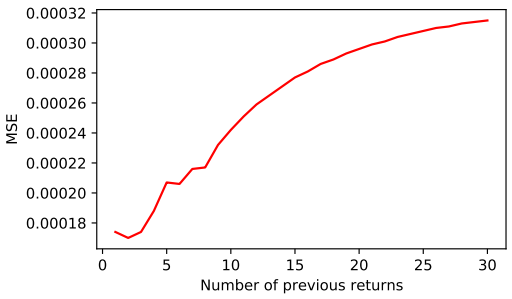

Figure 20. SVR all features MSE results

According to the SVR feature testing, the optimal number of previous returns is 1. Two sets of cases are compared similarly to the random forest for a fair comparison between the models. As seen in Table 12 and 13, the optimised parameters greatly improved the model's MAE, MAPE, MSE and RMSE metrics. A very significant improvement occurred with 30 previous returns – the MAPE value reduced from 866.428% to 105.440%. Also, with one previous returns, the value of MAPE reduced a bit over two times. With optimised parameters, the SVR model proved to be more effective with 30 previous returns regarding MAE, MAPE, MSE and RMSE. However, like the random

forest, the SVR's sMAPE greatly increased with the optimised variation of the model. To conclude, the parameter optimisation proved to offer significant improvements over the default parameters. Also, it is reasonable to test the sliding window method with both sets of parameters.

Table 12. SVR results with default parameters

N prev. returns	MAE	MAPE	sMAPE	MSE	RMSE
1	0.0071281	226.278%	149.743%	0.0001659	0.0129
30	0.0123163	866.428%	143.149%	0.0002717	0.0165

Table 13. SVR results with optimised parameters

N prev. returns	MAE	MAPE	sMAPE	MSE	RMSE
1	0.006642	112.357%	172.399%	0.000156	0.0125
30	0.006687	105.440%	185.201%	0.000157	0.0125

The sliding method is implemented with two sets of features for SVR: 1 previous return and 30 previous returns. Unlike random forest, the high, low, open and close values are not used for the SVR model with the sliding window implementation.

The sliding window implementation results with one previous return as features can be seen in Table 14. For the most part, the results are of the same order of magnitude. For example, the difference between the maximum value and minimum value of sMAPE is 6%. The only outlier in the results is the test with a training size of 100 and test size of 10, where the MAPE, MSE and RMSE are vastly different from other tests. However, the SVR model seems to be a bit more effective with a smaller test size of 5 than with a higher test size of 10. Overview of SVR model sliding window implementation with 30 previous returns can be seen in Table 15. The results between 1 previous return and 30 previous return do not differentiate much. The same metric growth and decay patterns are also identical. For example, the MAE grows as the training size decreases. Also, a training size of 365 yielded the best results in regards to MAPE. Also, sMAPE is the lowest with a training size of 365, with the result of training size of 365 being the close second. In conclusion, the optimal training and test size depend heavily on the favoured metric. However, a training size of 365 with a test size of 5 seems to be the most optimal due to its' good results across various metrics.

Table 14. SVR sliding window results with 1 previous return

Training size	Test size	MAE	MAPE	sMAPE	MSE	RMSE
1000	5	0.001809	134.904%	66.605%	0.000013	0.002295
500	5	0.001813	127.716%	62.403%	0.000013	0.002309
365	5	0.001887	121.033%	61.806%	0.000015	0.002398
300	5	0.001897	126.677%	61.290%	0.000015	0.002402
100	5	0.002062	140.286%	63.849%	0.000017	0.002566
1000	10	0.001813	133.369%	67.359%	0.000013	0.002459
500	10	0.001878	132.227%	62.436%	0.000015	0.002539
365	10	0.001921	123.939%	62.683%	0.000014	0.002555
300	10	0.002064	141.501%	62.529%	0.000020	0.002738
100	10	0.002505	220.737%	64.919%	0.000095	0.003334

Table 15. SVR sliding window results with 30 previous return

Training size	Test size	MAE	MAPE	sMAPE	MSE	RMSE
1000	5	0.001824	133.317%	66.993%	0.000013	0.002321
500	5	0.001839	127.577%	63.749%	0.000013	0.002324
365	5	0.001865	123.760%	61.886%	0.000013	0.002367
300	5	0.001894	127.980%	62.082%	0.000014	0.002389
100	5	0.002279	140.768%	63.960%	0.000059	0.002816
1000	10	0.001817	128.120%	66.935%	0.000013	0.002472
500	10	0.001853	127.928%	63.806%	0.000013	0.002492
365	10	0.001924	127.943%	63.265%	0.000013	0.002568
300	10	0.002000	129.264%	63.327%	0.000016	0.002663
100	10	0.002344	146.169%	65.194%	0.000032	0.003071

6.4.3 K-nearest neighbours

For KNN, the feature testing overview results can be seen in Tables 33, 34. When the feature set included only the n number of previous returns, the MAPE, MAE and MSE varied moderately in terms of magnitude. However, the sMAPE is relatively stable. There seems to be no correlation between the number of returns and the results as seen on Figures 22, 24, 26, 33, 35.

When the feature set also included high, low, open, close values in addition to previous returns, the results are relatively stable. However, as the number of previous

returns increased, generally, the errors also increased. The most optimal number of previous returns is 1.

In general, including all the features yielded better results in almost all the metrics, except the MAPE. However, the test results, including all the features, are relatively stable. In conclusion, all the observed features are included in the further tests for the KNN model due to better results.

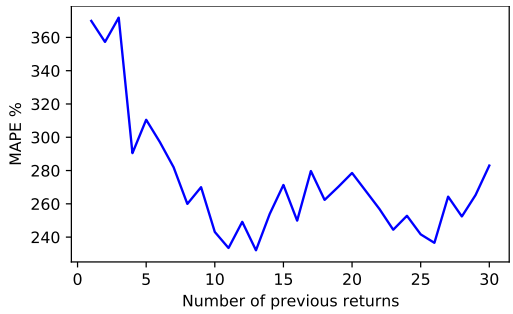

Figure 21. KNN n nr of previous returns MAPE results

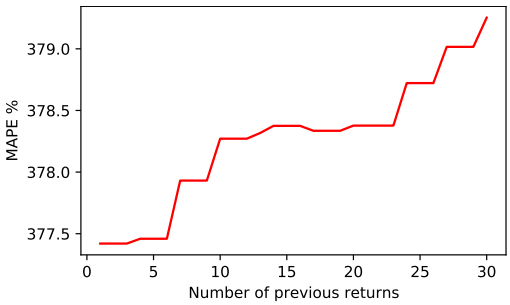

Figure 22. KNN all features MAPE results

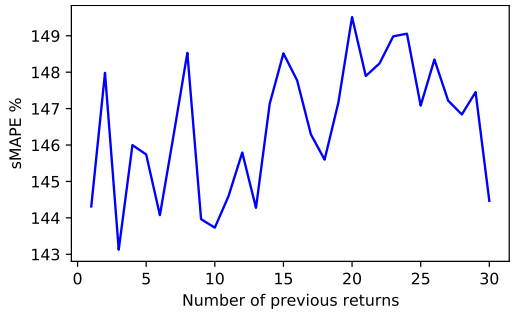

Figure 23. KNN n nr of previous returns sMAPE results

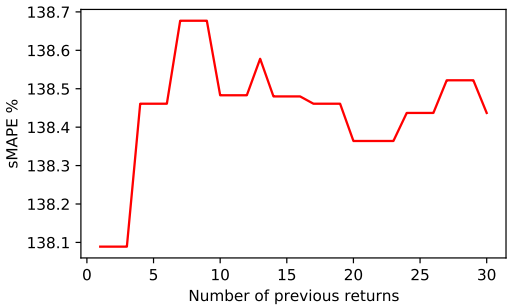

Figure 24. KNN all features sMAPE results

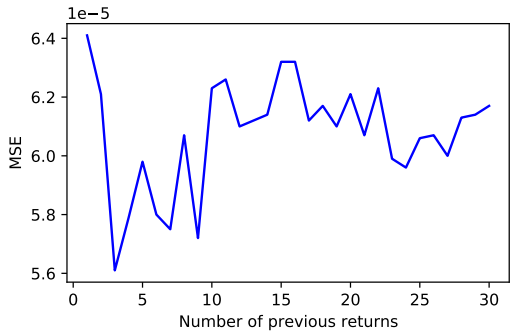

Figure 25. KNN n nr of previous returns MSE results

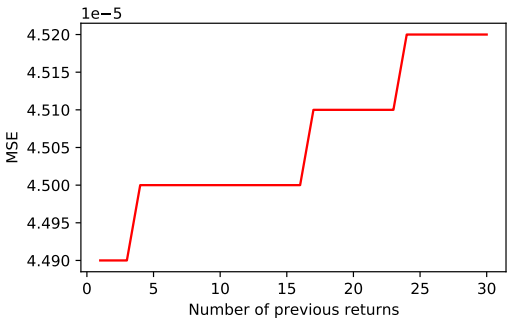

Figure 26. KNN all features MSE results

Parameter optimisation is carried out with n previous returns, high, low, open and close values as features. However, the optimal number of n previous returns is one, according to the feature testing tests where n = 1 and n = 30 are carried out for fair comparison with other machine learning models. The parameter optimisation concluded that the optimal k for KNN is three.

For a set of features where n = 1, only the MAPE metric is moderately improved with the optimisation – from 138% to 129%. The MSE and RMSE values also decreased but by a meagre margin, as seen in Table 16. MAE and MAPE values increased. For a set of features where n = 30, the model also improved only regarding sMAPE. The other metrics' values increased with the optimised model.

In conclusion, the optimised KNN model improved only regarding sMAPE and deteriorated with regards to other metrics. The optimisation is not as successful for the KNN model as it is with random forest and SVR.

Table 16. KNN results with default parameters

N prev. returns	MAE	MAPE	sMAPE	MSE	RMSE
1	0.0043319	377.420%	138.089%	0.0000449	0.0067
30	0.0043565	379.253%	138.437%	0.0000452	0.0067

Table 17. KNN results with optimised parameters

N prev. returns	MAE	MAPE	sMAPE	MSE	RMSE
1	0.004337	406.823%	129.109%	0.000044	0.0066
30	0.004514	420.124%	129.911%	0.000047	0.0069

The sliding window method is performed with two sets of features:

1) 1 previous return, high, low, open, close values 2) 30 previous returns, high, low, open, close values

The results between the two sets are almost equal as seen in Tables 18 and 19. However, the second set, where n = 30, had slightly better results. Overall, a higher training size is more favourable for the KNN model. The errors gradually increased as the size of the training size decreased for the set of results. The test size had a minor impact on the results – all the metrics except MAPE are greater, with a test size of 10. Although, the difference is minor.

In conclusion, the KNN performed better with n = 30 and a more extensive training size and a smaller test size of 5.

Table 18. KNN sliding window results with 1 previous return

Training size	Test size	MAE	MAPE	sMAPE	MSE	RMSE
1000	5	0.003074	201.410%	100.715%	0.000031	0.003796
500	5	0.004186	237.488%	106.728%	0.000062	0.005206
365	5	0.004818	237.170%	109.437%	0.000090	0.005949
300	5	0.004857	235.477%	109.942%	0.000087	0.005994
100	5	0.005236	254.503%	115.289%	0.000095	0.006391
1000	10	0.003126	207.538%	101.042%	0.000032	0.004066
500	10	0.004409	250.347%	108.955%	0.000069	0.005701
365	10	0.004946	240.381%	111.768%	0.000086	0.006381
300	10	0.005101	241.137%	112.827%	0.000094	0.006530
100	10	0.005432	252.849%	117.991%	0.000099	0.006909

Table 19. KNN sliding window results with 30 previous return

Training size	Test size	MAE	MAPE	sMAPE	MSE	RMSE
1000	5	0.003058	198.870%	100.521%	0.000031	0.003791
500	5	0.004060	235.659%	107.105%	0.000057	0.005035
365	5	0.004800	229.944%	109.217%	0.000088	0.005906
300	5	0.004873	229.376%	110.175%	0.000087	0.005976
100	5	0.005309	251.451%	115.660%	0.000096	0.006441
1000	10	0.003116	196.041%	101.485%	0.000033	0.004042
500	10	0.004283	242.706%	108.508%	0.000064	0.005539
365	10	0.004997	246.391%	111.847%	0.000089	0.006404
300	10	0.005055	228.944%	112.435%	0.000091	0.006496
100	10	0.005499	250.767%	118.310%	0.000098	0.006990

6.5 ARMA and machine learning comparison

The ARMA model is compared with the machine learning model in two different methods. The first method is the method of taking the complete training and testing on the whole data set. The second method is the sliding window method. The ARMA(2,1) is chosen for both comparisons. Although the ARMA(3,2) model outperformed ARMA(2,1) in MAE, sMAPE, MSE and RMSE metrics, the error difference is minimal. Generally, the performance of ARMA is not significantly affected by the orders of p and q as seen from Table 7. Therefore, any of the models specified in Table 7 would be sufficient to compare the machine learning models.

The best performing models are also chosen among the machine learning models. First, the standard method's best-performing machine learning model is chosen to compare with the ARMA model. Each machine learning model excelled in specific metrics. For example, the random forest has the lowest MAE. SVR has the lowest MAPE, but the random forest model's MAPE is not far off – 105.440% and 109.421%, respectively. However, KNN has the lowest sMAPE, MSE and RMSE values. All in all, the optimal machine learning model to be compared to ARMA heavily depends on the favoured metric. However, to give a complete overview of all the metrics, each metric's best performing models are compared with the respective ARMA metric value.

The comparison of the models is displayed in Table 20. The best result of each metric is coloured grey. As seen, each model excels in a specific metric. The ARMA(2,1) model outperforms the machine learning models in MAE, with the random forest being a close second. The SVR and random forest models are the most optimal if MAPE is the favoured metric. KNN is the best model to achieve the lowest sMAPE. Random forest exceeds others in regards to MSE and RMSE. In conclusion, there is not a single model that outshines others in all regards. On the contrary, the appropriate model should be chosen according to the most favoured metric. However, if one should choose to have an all-around model, the random forest would have great results in all metrics, except sMAPE, where the KNN model excelled.

Table 20. Best performing ARMA and Machine learning model comparison

Model / Metric	**MAE**	**MAPE**	**sMAPE**	**MSE**	**RMSE**
ARMA(2,1)	0.003871	124.312%	168.777%	0.00000516	0.0072
RF	0.003872	109.421%	182.709%	0.00005055	0.0071
SVR	0.006687	105.440%	185.201%	0.000157	0.0125
KNN	0.004514	420.124%	129.911%	0.000047	0.0069

Secondly, the best performing models of machine learning and ARMA are compared in the sliding window method. For the comparison, only the best results yielding training size and test size values are chosen. The best values for each metric varied across three different parameter sets:

1) training size = 1000 & test size = 5
2) training size = 365 & test size = 5
3) training size = 1000 & test size = 10

The overview of the results is given in Table 21, where the lowest value of each metric is highlighted in grey. Overall, the best performing model in the sliding window method is the SVR model, which had the lowest value in a total of 4 metrics – MAE, sMAPE, MSE and RMSE. In terms of sMAPE, the SVR model exceeded the other models by at least 39%. Regarding MSE, the difference is more than twice as big. THE ARMA model

excelled with the MAPE metric with the lowest value of 119.84%. The machine learning model's MAPE is not far off - SVR with a MAPE of 123.76%.

In conclusion, the machine learning models exceeded the ARMA model in almost all of the metrics. Therefore, machine learning has great potential in predicting the Baltic stock market return when compared to the ARMA model.

Table 21. ARMA and machine learning results of sliding window method

	Training size	Test size	MAE	MAPE	sMAPE	MSE	RMSE
ARMA(2,1)	1000	5	0.004324	122.68	171.97	0.000052	0.005269
RF	1000	5	0.004332	136.487	165.986	0.000053	0.005289
SVR	1000	5	0.001824	133.317	66.993	0.000013	0.002321
KNN	1000	5	0.003058	198.870	100.521	0.000031	0.003791
ARMA(2,1)	365	5	0.005718	131.66	165.43	0.000102	0.006931
RF	365	5	0.005787	172.378	158.141	0.000099	0.007067
SVR	365	5	0.001865	123.760	61.886	0.000013	0.002367
KNN	365	5	0.004800	229.944	109.217	0.000088	0.005906
ARMA(2,1)	1000	10	0.004327	119.84	172.93	0.000052	0.005524
RF	1000	10	0.004368	137.181	165.987	0.000054	0.005579
SVR	1000	10	0.001817	128.120	66.935	0.000013	0.002472
KNN	1000	10	0.003116	196.041	101.485	0.000033	0.004042

6.6 GARCH

For the GARCH model, the same subset of tests is carried out for the ARMA model – testing the different values of p and q (from one to five for each parameter) and the sliding window method test.

Table 35 gives an overview of the GARCH p and q test results. The test consisted of 25 models, where p, q have a minimum value of one and a maximum value of five. Overall, the GARCH(1,1) outperformed the other variations of the GARCH models. The only exception where GARCH(1,1) did not outperform all the others is in MSE and RMSE metrics, where the GARCH(2,1) model excelled. Overall, the results varied significantly when the order of p and q is changed. Although, the MAPE, MAE, MSE, RMSE metrics fluctuated considerably more than the sMAPE metric. Also, GARCH(4,1) model results are very significant outliers. In conclusion, the GARCH(1,1) is the best performing model among GARCH models and is used for the sliding window method test.

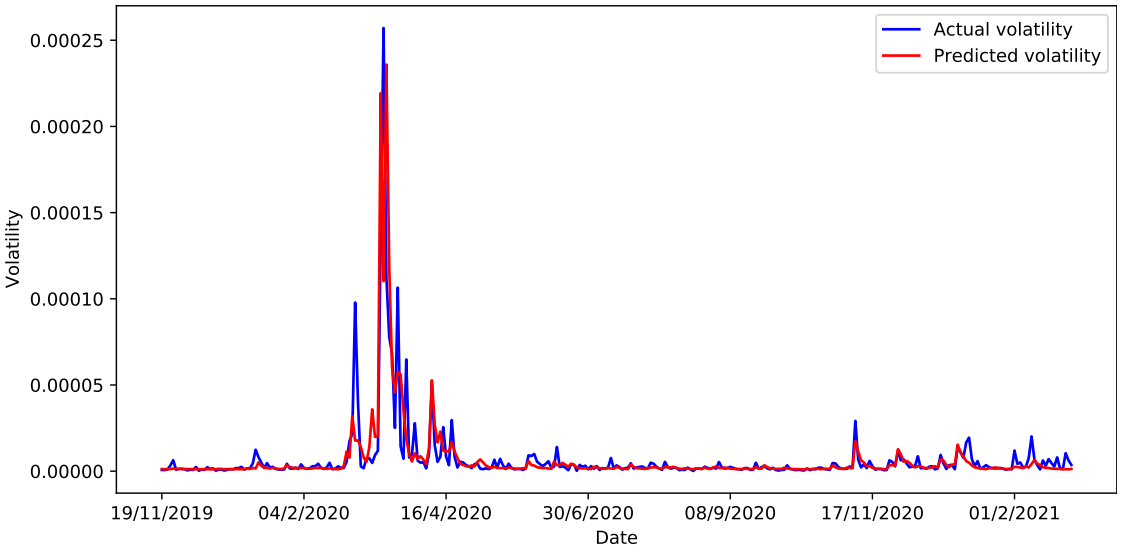

Figure 27. GARCH(1,1) volatility prediction

The sliding window method with the GARCH(1,1) model is executed with training sizes of 1000, 500, 365, 300, 100, and test sizes of 5 and 10 – a total of 10 tests. Table 22 gives an overview of the GARCH(1,1) sliding window results.

The MAE values are the lowest when the training size is 1000 – modifying the test size did not significantly impact the result. The MSE and RMSE values are also the lowest with a higher training size. However, MAPE is at its lowest with a smaller training size. However, the sMAPE values are all varying between about 60% and 67%. The difference between the lowest and highest sMAPE values does not exceed 8%. However, the best parameter set for sMAPE is a training size of 100 and a test size of 5 – this

shows that the recent history is more impactful in regards to MAPE, and long history is more significant in regards to MAE, sMAPE, MSE and RMSE.

Table 22. GARCH(1,1) sliding window results

Training size	Test size	MAE	MAPE	sMAPE	MSE	RMSE
1000	5	3.36E-06	108.772	62.30179	2.04E-11	4.11E-06
500	5	5.57E-06	120.217	60.15201	4.00E-10	6.31E-06
365	5	4.79E-06	115.560	59.88104	3.16E-10	5.42E-06
300	5	4.47E-06	110.575	60.89112	2.90E-10	5.10E-06
100	5	4.45E-06	97.607	66.09664	7.35E-10	6.00E-06
1000	10	3.53E-06	115.261	62.48914	2.32E-11	4.73E-06
500	10	6.17E-06	122.235	60.35259	8.60E-10	8.47E-06
365	10	4.79E-06	115.853	59.74211	3.14E-10	5.57E-06
300	10	4.49E-06	112.073	61.954	2.87E-10	5.65E-06
100	10	4.48E-06	95.786	66.95244	8.34E-10	7.34E-06

In conclusion, a higher training size is more favourable for MAE, MSE and RMSE, and a smaller training size is more favourable for MAPE. For sMAPE, however, the optimal training size is 365. The test size did not have a significant impact on the results.

6.7 GARCH-ANN

For the GARCH-ANN model, various structures of the ANN model is created for evaluation. Six different structures, which vary by the number of layers used and the dimensionality of the layer's output space, are represented in Table 23. It is important to note that the default optimiser RMSprop [58] is used, and no layer activation functions are used in the structure testing.

Table 23. GARCH-ANN structure testing

Structure	MAE	MAPE	sMAPE	MSE	RMSE
Structure 1	3.06E-06	58.258	66.791	2.87E-10	1.70E-05
Structure 2	3.56E-06	182.221	79.050	2.80E-10	1.67E-05
Structure 3	2.98E-06	63.332	61.652	2.86E-10	1.69E-05
Structure 4	3.13E-06	57.502	71.571	2.88E-10	1.70E-05
Structure 5	3.05E-06	58.574	65.960	2.87E-10	1.70E-05
Structure 6	3.02E-06	59.805	64.124	2.87E-10	1.69E-05

Overall, most of the structures performed similarly across various metrics. The only exception is Structure 2, which has significantly higher MAPE and slightly higher sMAPE. Structure 3 and structure 2 are almost identical, apart from the dropout layer that the structure 3 model has. The high error is most likely avoided in the structure 3 predictions thanks to the dropout layer. However, structures 1, 4, 5 and 6 (in addition to structure 2) did not have a dropout layer, and the results do not show a significant rise in errors. On the contrary, the MAPE is lower without a dropout layer. However, the sMAPE is slightly higher without the dropout layer than with. Adding more layers and increasing the dimensionality of the layer output did not significantly impact the results. Structure 6 had more layers than others, and the first layers also had higher dimensionality, but the results are still similar. Structure 6 is used to test the optimisers and layer activation functions due to the most outstanding results.

Tables 36-44 give an overview of the cross-validation tests of the optimisers and the layer activation functions. Cross-testing did not yield significantly better results. Only a few combinations have considerably higher errors. For example, using the ftrl optimiser led to very larger errors in some cases. The sgd optimiser is not presented because the results are not a number (NaN) values in all cases.

Overall, the sigmoid-adam combination has the best balance between MAPE and sMAPE. Even though many combinations yielded better MAPE or sMAPE, then the other metric (MAPE or sMAPE) had significantly higher errors. Structure 6 with the sigmoid activation function and adam optimiser is used with the sliding window method.

An overview of the GARCH-ANN sliding window method results is given in Table 22. The results indicate that a smaller test size of 5 is more effective than having a test size of 10 – this applies to all metrics. A smaller training size proved to yield better results regarding MAE, MAPE and sMAPE, than a larger training size. For example, with a training size of 1000 and a test size of 5, the MAPE is almost 30% higher than with a training size of 5. The difference in MAE is almost double. However, the MSE and RMSE metrics indicate that a higher training size is more beneficial than a smaller training size. The difference between MSE and RMSE is almost double between the tests with a training size of 1000 and a training size of 100.

In conclusion, if minimising MAE, MAPE or sMAPE is the main priority, then a smaller training size should be used. If the MSE or RMSE metric is the main priority, then a larger training size is preferred. In any case, a smaller test size should be chosen for better effectiveness.

Table 24. GARCH-ANN sliding window method

Training size	Test size	MAE	MAPE	sMAPE	MSE	RMSE
1000	5	2.64E-06	119.504	60.498	1.75E-10	3.76E-06
500	5	4.04E-06	101.804	55.645	2.47E-10	5.75E-06
365	5	5.02E-06	101.118	56.113	3.61E-10	6.98E-06
300	5	4.99E-06	98.454	56.751	3.50E-10	6.94E-06
100	5	5.07E-06	90.926	56.220	3.47E-10	7.01E-06
1000	10	1.24E-05	1074.851	144.060	4.08E-10	1.36E-05
500	10	1.30E-05	714.963	137.284	4.34E-10	1.47E-05
365	10	5.10E-06	95.357	56.486	3.63E-10	7.78E-06
300	10	5.22E-06	116.604	58.011	3.84E-10	8.06E-06
100	10	5.40E-06	105.967	59.054	3.78E-10	8.27E-06

6.8 GARCH and GARCH-ANN comparison

GARCH and GARCH-ANN models are compared with two different methods: the standard method and the sliding window method. The GARCH(1,1) is the best performing model among the GARCH models. The best performing GARCH-ANN model consisted of GARCH(1,1) model, the sigmoid layer activation function and adam algorithm as the optimiser. The results conclude that the GARCH(1,1) outperformed the GARCH-ANN model regarding sMAPE, MSE and RMSE. The GARCH-ANN model had lower MAE and MAPE.

The sliding window results are displayed on Table 22, 24. The results show that even though the difference is not large, the GARCH-ANN model outperformed the GARCH model in every metric in the sliding window technique. GARCH model only outperformed the GARCH-ANN model by having more consistent results: the GARCH-ANN model had very high errors in two cases. Meanwhile, the GARCH's results did not fluctuate as much.

In conclusion, when applying the sliding window method, the GARCH-ANN model yields better results. However, if the standard method of using the whole data set to train and test, then the GARCH model is more effective when the favoured evaluation metric is sMAPE, MSE or RMSE. However, if MAE or MAPE is preferred to use as the evaluation metric, then the GARCH-ANN model would be more effective.

6.9 Standardised residuals

The histograms of the standardised residuals for the standard method are displayed on Figures 36, 37, 38, 39. The histograms show that the econometric models had the most standardised residuals between zero and one – almost 300. Machine learning models

had a much lower number of standardised residuals between zero and one. Overall, the machine learning models had substantially higher standardised residuals than the econometric models' standardised residuals. Also, as seen in Figure 36, there are almost no standardised residuals above seven for econometric models. However, there are many higher standardised residual values for the machine learning models – The SVR model has the highest amount.

Table 25 displays the mean absolute standardised residuals for all methods. The econometric approach had the lowest mean – 1.8633. The SVR approach had the highest mean of the absolute standardised residuals of 5.4157.

Table 25. Mean absolute standardised residuals with the sliding window method

Method	Mean absolute standardised residuals
Econometric	1.8633
Random forest	3.2216
SVR	5.4157
KNN	3.5365

Figures 40, 41, 42, 43 display the histograms of the standardised residuals with the sliding window method. Compared to the standard method, the econometrics models have significantly more standardised residuals in the higher range of values. Contrary to the standard method, the SVR and KNN models have substantially better results. The SVR approach has a low number of high values of standardised residuals, as seen in Figure 42. The standardised residuals of the random forest did not improve significantly.

Table 26 show the mean absolute standardised residuals with the sliding window method. With the sliding window method, the SVR approach has the best result with a mean of 1.2236. The random forest method has the highest mean, and the econometric models have a mean of 2.6327, which is higher than the standard method.

In conclusion, the sliding window method is very beneficial for the machine learning models, but the mean of the standardised residuals increased significantly for the econometric models. Overall, the SVR model with the sliding window method had the lowest standardised residuals.

Table 26. Sliding window method mean absolute standardised residuals

Method	Mean absolute standardised residuals
Econometric	2.6327
Random forest	2.8816
SVR	1.2236
KNN	2.0968

7 Conclusion

The goal of the thesis was to predict the return of the Baltic stock market with machine learning and ARMA models, compare the results and determine which approach is more efficient. Another goal was to predict the volatility of the Baltic stock market with the GARCH and GARCH-ANN models, compare the results and determine which approach predicts the volatility better. The goal of the thesis was achieved. Several models were created, an extensive number of tests were carried out, and a fair comparison was performed between the different approaches. The overall effectiveness of the machine learning and financial models was then compared with the standard residuals. The research managed to contribute by offering a fair comparison between different approaches to predicting the Baltic stock market.

In terms of machine learning and ARMA comparison, the various machine learning models outperformed the ARMA in almost all of the metrics, except the MAPE metric in the sliding window method and the MAE metric when applying the standard method. Although, the difference between the machine learning's MAE and MAPE values and ARMA's MAE and MAPE values were not as significant. The machine learning models proved to be more effective overall. However, the results indicate that one machine learning model is not as effective in all areas. Instead, one should choose an appropriate machine learning model depending on the metric that the results are evaluated on.

Regarding GARCH and GARCH-ANN models, the GARCH model outperformed the GARCH-ANN model in sMAPE, MSE and RMSE when applying the standard method. Using the sliding window technique, the GARCH-ANN was superior in every metric but mostly with only a small margin. Overall, the volatility prediction errors were mostly lower when applying the standard method instead of the sliding window method.

Regarding standardised residuals, with the standard method, the econometrics models' mean of absolute standardised residuals was the lowest of all. However, with the sliding window method, the SVR model had a significantly lower mean than others.

Overall, the machine learning models random forest, SVR, KNN and GARCH-ANN performed better when applying the sliding window method. The econometric models ARMA and GARCH usually outperformed the machine learning models when using the standard method. All in all, the most effective approach to predict the return of the Nasdaq Baltic index is to use the machine learning models with the sliding window method. For volatility prediction, both GARCH and GARCH-ANN are viable options. However, when applying the sliding window technique, the GARCH-ANN model yields better results, and when using the standard method, the GARCH model provides lower errors.

Future work on stock market prediction can undoubtedly be very extensive. There are several ways to expand on the topic. The data set could be modified: for example, filtering the outliers to minimise the impact of the outliers in the data and research the prediction if the time-series is not as volatile. More machine learning models, financial

models and hybrid models could be implemented. More features could also be introduced — for example, technical indicators and financial statements. In addition, the sentiment analysis could also be used to determine the public's opinion about specific stocks or the market overall. The larger stock markets' data could also be introduced to see if there are consistent correlations between the markets.

References

[1] Jussi Nikkinen, Vanja Piljak, and Janne Äijö. "Baltic stock markets and the financial crisis of 2008–2009". In: *Research in International Business and Finance* 26.3 (2012), pp. 398–409.

[2] Andres Suimets. *Kas Praegu on Õige Aeg Investeerida Tallinna Börsile?* Sept. 2020. URL: https://kukkur.swedbank.ee/investeerimine/kas-praegu-on-oige-aeg-investeerida-tallinna-borsile (visited on 03/03/2021).

[3] Yun Li. *80% of the Stock Market Is Now on Autopilot*. June 2019. URL: https://www.cnbc.com/2019/06/28/80percent-of-the-stock-market-is-now-on-autopilot.html (visited on 02/26/2021).

[4] *Algorithmic Trading Market - Growth, Trends, Forecasts (2020 - 2025)*. URL: https://www.mordorintelligence.com/industry-reports/algorithmic-trading-market (visited on 01/30/2021).

[5] Hakob Grigoryan et al. "Stock market prediction using artificial neural networks. Case Study of TAL1T, Nasdaq OMX Baltic Stock". In: *Database Systems Journal* 6.2 (2015), pp. 14–23.

[6] *Shares*. URL: https://nasdaqbaltic.com/statistics/en/shares (visited on 04/22/2021).

[7] *Statistics*. URL: https://nasdaqbaltic.com/statistics/en/statistics (visited on 04/22/2021).

[8] *About Indexes*. URL: https://nasdaqbaltic.com/market-information/about-indexes/ (visited on 02/18/2021).

[9] Harun Ercan. "Baltic stock market prediction by using NARX". In: *2017 12th International Scientific and Technical Conference on Computer Sciences and Information Technologies (CSIT)*. Vol. 1. IEEE. 2017, pp. 464–467.

[10] Wei Huang, Yoshiteru Nakamori, and Shou-Yang Wang. "Forecasting stock market movement direction with support vector machine". In: *Computers & operations research* 32.10 (2005), pp. 2513–2522.

[11] Rohit Choudhry and Kumkum Garg. "A hybrid machine learning system for stock market forecasting". In: *World Academy of Science, Engineering and Technology* 39.3 (2008), pp. 315–318.

[12] Yakup Kara, Melek Acar Boyacioglu, and Ömer Kaan Baykan. "Predicting direction of stock price index movement using artificial neural networks and support vector machines: The sample of the Istanbul Stock Exchange". In: *Expert systems with Applications* 38.5 (2011), pp. 5311–5319.

[13] Michel Ballings et al. “Evaluating multiple classifiers for stock price direction prediction”. In: *Expert Systems with Applications* 42.20 (2015), pp. 7046–7056.

[14] Jigar Patel et al. “Predicting stock market index using fusion of machine learning techniques”. In: *Expert Systems with Applications* 42.4 (2015), pp. 2162–2172.

[15] Wen Long, Zhichen Lu, and Lingxiao Cui. “Deep learning-based feature engineering for stock price movement prediction”. In: *Knowledge-Based Systems* 164 (2019), pp. 163–173.

[16] David MQ Nelson, Adriano CM Pereira, and Renato A de Oliveira. “Stock market’s price movement prediction with LSTM neural networks”. In: *2017 International joint conference on neural networks (IJCNN)*. IEEE. 2017, pp. 1419–1426.

[17] Kai Chen, Yi Zhou, and Fangyan Dai. “A LSTM-based method for stock returns prediction: A case study of China stock market”. In: *2015 IEEE international conference on big data (big data)*. IEEE. 2015, pp. 2823–2824.

[18] 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.

[19] Mohammad Mahdi Rounaghi and Farzaneh Nassir Zadeh. “Investigation of market efficiency and Financial Stability between S&P 500 and London Stock Exchange: Monthly and yearly Forecasting of Time Series Stock Returns using ARMA model”. In: *Physica A: Statistical Mechanics and its Applications* 456 (2016), pp. 10–21. ISSN: 0378-4371. DOI: https://doi.org/10.1016/j.physa.2016.03.006. URL: https://www.sciencedirect.com/science/article/pii/S0378437116002776.

[20] Niel Oberholzer and Pierre Venter. “Univariate GARCH Models Applied to the JSE/FTSE Stock Indices”. In: *Procedia Economics and Finance* 24 (2015). International Conference on Applied Economics (ICOAE) 2015, 2-4 July 2015, Kazan, Russia, pp. 491–500. ISSN: 2212-5671. DOI: https://doi.org/10.1016/S2212-5671(15)00616-4. URL: https://www.sciencedirect.com/science/article/pii/S2212567115006164.

[21] *List of Stock Exchanges*. URL: https://en.wikipedia.org/wiki/List_of_stock_exchanges (visited on 03/15/2021).

[22] Luckyson Khaidem, Snehanshu Saha, and Sudeepa Roy Dey. “Predicting the direction of stock market prices using random forest”. In: *arXiv preprint arXiv:1605.00003* (2016).

[23] Theodore B Trafalis and Huseyin Ince. "Support vector machine for regression and applications to financial forecasting". In: *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks. IJCNN 2000. Neural Computing: New Challenges and Perspectives for the New Millennium*. Vol. 6. IEEE. 2000, pp. 348–353.

[24] Huisu Jang and Jaewook Lee. "An empirical study on modeling and prediction of bitcoin prices with bayesian neural networks based on blockchain information". In: *Ieee Access* 6 (2017), pp. 5427–5437.

[25] Jonathan L Ticknor. "A Bayesian regularized artificial neural network for stock market forecasting". In: *Expert Systems with Applications* 40.14 (2013), pp. 5501–5506.

[26] Justin Kuepper. *Volatility Definition*. Feb. 2021. URL: https://www.investopedia.com/terms/v/volatility.asp#:~:text=Volatility%20is%20a%20statistical%20measure,same%20security%20or%20market%20index. (visited on 03/05/2021).

[27] Elizaveta Lebedeva. "Spillovers between cryptocurrencies. Network map of cryptocurrencies". PhD thesis. Master's Thesis, University of Tartu, Tartu, Estonia, 2018.

[28] *Capped Index*. URL: https://www.investopedia.com/terms/c/capped-index.asp (visited on 02/18/2021).

[29] *GOOGLEFINANCE*. URL: https://www.investopedia.com/terms/c/capped-index.asp (visited on 03/25/2021).

[30] *Tallink Grupp*. URL: https://nasdaqbaltic.com/statistics/en/instrument/EE3100004466/trading (visited on 05/05/2021).

[31] *Baltic market indexes*. URL: https://nasdaqbaltic.com/statistics/en/charts (visited on 05/05/2021).

[32] *Investing*. URL: https://www.investing.com/ (visited on 04/23/2021).

[33] Or Herman-Saffar. *Time Based Cross Validation*. 2020. URL: https://towardsdatascience.com/time-based-cross-validation-d259b13d42b8 (visited on 04/20/2021).

[34] Peter Whittle. *Hypothesis testing in time series analysis*. Vol. 4. Almqvist & Wiksells boktr., 1951.

[35] *14.1 - Autoregressive Models*. URL: https://online.stat.psu.edu/stat501/lesson/14/14.1 (visited on 04/25/2021).

[36] *14.5.1 - ARIMA Models*. URL: https://online.stat.psu.edu/stat501/lesson/14/14.5/14.5.1 (visited on 04/25/2021).

[37] Shay Palachy. *Stationarity in time series analysis*. Apr. 2019. URL: https://towardsdatascience.com/stationarity-in-time-series-analysis-90c94f27322 (visited on 03/06/2021).

[38] Wu Ji and Keong Chan Chee. "Prediction of hourly solar radiation using a novel hybrid model of ARMA and TDNN". In: *Solar Energy* 85.5 (2011), pp. 808–817. ISSN: 0038-092X. DOI: https://doi.org/10.1016/j.solener.2011.01.013. URL: https://www.sciencedirect.com/science/article/pii/S0038092X11000259.

[39] Jonathan Taylor Josef Perktold Skipper Seabold. *Augmented Dickey-Fuller unit root test*. Feb. 2021. URL: https://www.statsmodels.org/stable/generated/statsmodels.tsa.stattools.adfuller.html (visited on 03/27/2021).

[40] Jonathan Taylor Josef Perktold Skipper Seabold. *Source code for statsmodels.tsa.stattools*. Feb. 2021. URL: https://www.statsmodels.org/stable/_modules/statsmodels/tsa/stattools.html#adfuller (visited on 03/27/2021).

[41] Taylor G Smith. *pmdarima.arima.auto_arima*. URL: https://alkaline-ml.com/pmdarima/modules/generated/pmdarima.arima.auto_arima.html (visited on 03/27/2021).

[42] Robert F. Engle. "Autoregressive Conditional Heteroscedasticity with Estimates of the Variance of United Kingdom Inflation". In: *Econometrica* 50.4 (1982), pp. 987–1007. ISSN: 00129682, 14680262. URL: http://www.jstor.org/stable/1912773.

[43] Tim Bollerslev. "Generalized autoregressive conditional heteroskedasticity". In: *Journal of Econometrics* 31.3 (1986), pp. 307–327. ISSN: 0304-4076. DOI: https://doi.org/10.1016/0304-4076(86)90063-1. URL: https://www.sciencedirect.com/science/article/pii/0304407686900631.

[44] *sklearn.model*$_s election.GridSearchCV$. URL: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html (visited on 04/12/2021).

[45] Leo Breiman. "Random forests". In: *Machine learning* 45.1 (2001), pp. 5–32.

[46] Gérard Biau and Erwan Scornet. "A random forest guided tour". In: *Test* 25.2 (2016), pp. 197–227.

[47] *sklearn.ensemble.RandomForestRegressor*. URL: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html?highlight=randomforestregressor#sklearn.ensemble.RandomForestRegressor (visited on 04/12/2021).

[48] Mariette Awad and Rahul Khanna. "Support vector regression". In: *Efficient learning machines*. Springer, 2015, pp. 67–80.

[49] William S Noble. "What is a support vector machine?" In: *Nature biotechnology* 24.12 (2006), pp. 1565–1567.

[50] Gongde Guo et al. "KNN Model-Based Approach in Classification". In: *On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE*. Ed. by Robert Meersman, Zahir Tari, and Douglas C. Schmidt. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 986–996. ISBN: 978-3-540-39964-3.

[51] Onel Harrison. *Machine Learning Basics with the K-Nearest Neighbors Algorithm*. 2018. URL: https://towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761#:~:text=Summary-,The%20k%2Dnearest%20neighbors%20(KNN)%20algorithm%20is%20a%20simple,that%20data%20in%20use%20grows. (visited on 04/21/2021).

[52] Tao Ban et al. "Referential kNN Regression for Financial Time Series Forecasting". In: *Neural Information Processing*. Ed. by Minho Lee et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 601–608. ISBN: 978-3-642-42054-2.

[53] *sklearn.neighbors.KNeighborsRegressor*. URL: https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsRegressor.html?highlight=kneighborsregressor#sklearn.neighbors.KNeighborsRegressor (visited on 04/21/2021).

[54] François Chollet. *Keras library*. URL: https://keras.io/ (visited on 04/24/2021).

[55] *About Keras*. URL: https://keras.io/about/ (visited on 04/24/2021).

[56] *tf.keras.layers.Dense*. URL: https://www.tensorflow.org/api_docs/python/tf/keras/layers/Dense (visited on 04/24/2021).

[57] *tf.keras.layers.Dropout*. URL: https://www.tensorflow.org/api_docs/python/tf/keras/layers/Dropout (visited on 04/24/2021).

[58] *tf.keras.Model*. URL: https://www.tensorflow.org/api_docs/python/tf/keras/Model (visited on 04/25/2021).

[59] *Layer activation functions*. URL: https://keras.io/api/layers/activations/ (visited on 04/24/2021).

[60] *Optimizers*. URL: https://keras.io/api/optimizers/ (visited on 04/24/2021).

[61] Stephanie Glen. *Standardized Residuals in Statistics: What are They?* 2013. URL: https://www.statisticshowto.com/what-is-a-standardized-residuals/#:~:text=What%20do%20Standardized%20Residuals%20Mean,to%20the%20chi%2Dsquare%20value. (visited on 05/01/2021).

[62] Tianfeng Chai and Roland R Draxler. "Root mean square error (RMSE) or mean absolute error (MAE)?–Arguments against avoiding RMSE in the literature". In: *Geoscientific model development* 7.3 (2014), pp. 1247–1250.

[63] Cort J Willmott and Kenji Matsuura. “Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance”. In: *Climate research* 30.1 (2005), pp. 79–82.

[64] Arnaud De Myttenaere et al. “Mean absolute percentage error for regression models”. In: *Neurocomputing* 192 (2016), pp. 38–48.

[65] Sungil Kim and Heeyoung Kim. “A new metric of absolute percentage error for intermittent demand forecasts”. In: *International Journal of Forecasting* 32.3 (2016), pp. 669–679.

[66] Spyros Makridakis. “Accuracy measures: theoretical and practical concerns”. In: *International journal of forecasting* 9.4 (1993), pp. 527–529.

[67] J SCOTT ARMSTRONG. “LONG-RANGE FORECASTING”. In: (1985).

[68] Benito E Flores. “A pragmatic view of accuracy measurement in forecasting”. In: *Omega* 14.2 (1986), pp. 93–98.

[69] Z. Wang and A. C. Bovik. “Mean squared error: Love it or leave it? A new look at Signal Fidelity Measures”. In: *IEEE Signal Processing Magazine* 26.1 (2009), pp. 98–117. DOI: 10.1109/MSP.2008.930649.

[70] Rohan Naidu (https://stats.stackexchange.com/users/257815/rohan naidu). *Disadvantages of Mean Squared Error?* Cross Validated. URL:https://stats.stackexchange.com/q/424507 (version: 2019-09-01). eprint: https://stats.stackexchange.com/q/424507. URL: https://stats.stackexchange.com/q/424507.

Appendix

I. Tables

Table 27. ARMA orders of p and q test results

p	q	MAE	MSE	RMSE	MAPE	sMAPE
1	1	0.003876	0.00005198	0.0072	126.187%	167.512%
1	2	0.003876	0.00005233	0.0072	125.514%	167.152%
1	3	0.003872	0.00005208	0.0072	128.072%	165.458%
1	4	0.003873	0.0000519	0.0072	126.398%	166.815%
1	5	0.003859	0.0000519	0.0072	137.670%	160.147%
2	1	0.003871	0.00005161	0.0072	124.312%	168.777%
2	2	0.003861	0.00005133	0.0072	133.233%	164.832%
2	3	0.003922	0.00005229	0.0072	169.811%	159.358%
2	4	0.003878	0.0000523	0.0072	126.494%	166.440%
2	5	0.003892	0.00005354	0.0073	136.976%	161.392%
3	1	0.003870	0.00005162	0.0072	128.040%	165.935%
3	2	0.003866	0.00005112	0.0071	125.766%	165.329%
3	3	0.003879	0.00005211	0.0072	137.051%	161.123%
3	4	0.003876	0.00005163	0.0072	132.958%	165.532%
3	5	0.003917	0.00005334	0.0073	148.028%	159.532%
4	1	0.003872	0.00005176	0.0072	128.919%	165.820%
4	2	0.003872	0.00005191	0.0072	128.282%	165.895%
4	3	0.004512	0.00007678	0.0088	204.388%	164.656%
4	4	0.003848	0.00005128	0.0072	128.991%	162.887%
4	5	0.003905	0.0000549	0.0074	167.689%	155.925%
5	1	0.003866	0.00005144	0.0072	133.209%	164.453%
5	2	0.003868	0.0000515	0.0072	132.255%	164.455%
5	3	0.003990	0.00007677	0.0088	148.054%	161.621%
5	4	0.003876	0.00005206	0.0072	134.537%	165.062%
5	5	0.003897	0.00005279	0.0073	154.040%	160.215%

Table 28. ARMA results with hample filtered data

p	q	MAE	MAPE	sMAPE	MSE	RMSe
1	1	151.2827%	108.2022%	0.0020820	0.000011900	0.340%
1	2	149.1015%	109.3829%	0.0020788	0.000011860	0.340%
1	3	151.3090%	110.7059%	0.0020849	0.000012050	0.350%
1	4	148.2539%	111.7240%	0.0020888	0.000012040	0.350%
1	5	149.1430%	111.5622%	0.0020837	0.000011930	0.350%
2	1	152.3296%	108.5059%	0.0020765	0.000011920	0.350%
2	2	149.3010%	109.5005%	0.0020796	0.000011860	0.340%
2	3	149.9222%	110.8953%	0.0020855	0.000012040	0.350%
2	4	155.2814%	111.9066%	0.0020850	0.000012090	0.350%
2	5	148.0365%	112.4938%	0.0020867	0.000011980	0.350%
3	1	148.8403%	110.2076%	0.0020802	0.000011880	0.340%
3	2	148.1360%	110.2123%	0.0020806	0.000011890	0.340%
3	3	149.7610%	110.9058%	0.0020855	0.000012030	0.350%
3	4	154.5968%	112.0088%	0.0020846	0.000012080	0.350%
3	5	155.0702%	112.7864%	0.0020791	0.000011990	0.350%
4	1	154.6200%	109.9070%	0.0020851	0.000012020	0.350%
4	2	150.2723%	109.8711%	0.0020823	0.000011940	0.350%
4	3	151.5830%	109.8495%	0.0020864	0.000012050	0.350%
4	4	159.7955%	111.1426%	0.0020961	0.000012270	0.350%
4	5	172.9221%	113.1554%	0.0021060	0.000012280	0.350%
5	1	152.0785%	110.4971%	0.0020860	0.000012130	0.350%
5	2	148.3106%	109.8094%	0.0020822	0.000012020	0.350%
5	3	157.9302%	111.3432%	0.0020999	0.000012260	0.350%
5	4	168.6365%	112.8379%	0.0020884	0.000012140	0.350%
5	5	172.7790%	113.0267%	0.0021052	0.000012280	0.350%

Table 29. Random forest results with n number of previous returns as features

n prev. returns	MAE	MAPE	sMAPE	MSE	RMSE
1	0.0052959	469.225%	143.818%	0.0000770	0.0088
2	0.0044832	333.410%	147.209%	0.0000563	0.0075
3	0.0042926	300.801%	149.292%	0.0000528	0.0073
4	0.0041397	237.931%	148.567%	0.0000518	0.0072
5	0.0040083	214.825%	147.881%	0.0000511	0.0071
6	0.0039393	196.985%	149.634%	0.0000506	0.0071
7	0.0039410	207.665%	151.329%	0.0000506	0.0071
8	0.0039294	198.348%	154.204%	0.0000490	0.0070
9	0.0039148	186.982%	153.364%	0.0000487	0.0070
10	0.0038973	177.828%	153.518%	0.0000481	0.0069
11	0.0039089	178.874%	156.561%	0.0000487	0.0070
12	0.0038696	175.585%	156.070%	0.0000484	0.0070
13	0.0039050	174.623%	158.025%	0.0000492	0.0070
14	0.0038995	173.675%	156.893%	0.0000490	0.0070
15	0.0039023	166.379%	159.744%	0.0000493	0.0070
16	0.0038977	162.986%	159.815%	0.0000495	0.0070
17	0.0038865	159.830%	159.335%	0.0000495	0.0070
18	0.0038874	161.113%	159.119%	0.0000492	0.0070
19	0.0038810	162.124%	158.474%	0.0000499	0.0071
20	0.0038966	162.431%	159.554%	0.0000501	0.0071
21	0.0038890	154.264%	159.386%	0.0000506	0.0071
22	0.0038973	158.702%	158.886%	0.0000506	0.0071
23	0.0038998	160.707%	159.739%	0.0000499	0.0071
24	0.0039023	160.138%	159.998%	0.0000499	0.0071
25	0.0038973	158.256%	160.553%	0.0000495	0.0070
26	0.0038937	156.805%	160.826%	0.0000493	0.0070
27	0.0038922	150.446%	161.143%	0.0000495	0.0070
28	0.0039050	152.843%	162.439%	0.0000492	0.0070
29	0.0039153	150.154%	162.433%	0.0000493	0.0070
30	0.0039042	151.448%	162.099%	0.0000494	0.0070

Table 30. Random forest results with n number of previous returns, high, low, open and close values as features

n prev. returns	MAE	MAPE	sMAPE	MSE	RMSE
1	0.00453270	340.681%	142.120%	0.00005382	0.0073
2	0.00435649	309.805%	145.690%	0.00005248	0.0072
3	0.00419698	263.201%	146.391%	0.00005028	0.0071
4	0.00408411	230.473%	146.820%	0.00004951	0.0070
5	0.00401075	210.429%	148.088%	0.00004932	0.0070
6	0.00397919	204.374%	149.262%	0.00004974	0.0071
7	0.00396841	205.916%	149.698%	0.00004962	0.0070
8	0.00396711	204.630%	149.607%	0.00004893	0.0070
9	0.00395640	196.495%	149.702%	0.00004832	0.0070
10	0.00392679	181.626%	150.644%	0.00004812	0.0069
11	0.00392788	175.247%	152.698%	0.00004848	0.0070
12	0.00391155	177.379%	153.391%	0.00004824	0.0069
13	0.00392271	171.068%	154.486%	0.00004870	0.0070
14	0.00390076	161.630%	154.954%	0.00004847	0.0070
15	0.00392215	158.702%	156.622%	0.00004917	0.0070
16	0.00391058	159.098%	156.894%	0.00004914	0.0070
17	0.00391353	158.071%	156.554%	0.00004945	0.0070
18	0.00392472	156.476%	157.816%	0.00004945	0.0070
19	0.00391391	149.576%	158.194%	0.00004952	0.0070
20	0.00392486	156.166%	157.615%	0.00004996	0.0071
21	0.00392728	152.449%	157.995%	0.00005053	0.0071
22	0.00392128	150.529%	157.985%	0.00005031	0.0071
23	0.00393199	155.495%	159.228%	0.00004979	0.0071
24	0.00393077	154.785%	159.367%	0.00004964	0.0070
25	0.00392367	159.604%	161.343%	0.00004934	0.0070
26	0.00392391	156.427%	159.232%	0.00004934	0.0070
27	0.00392948	156.751%	161.056%	0.00004946	0.0070
28	0.00393870	155.605%	162.332%	0.00004923	0.0070
29	0.00393208	146.696%	161.457%	0.00004982	0.0071
30	0.00394362	151.616%	162.270%	0.00004974	0.0071

Table 31. SVR results with high, low, close open and n number of previous returns as features

n prev. returns	MAE	MAPE	sMAPE	MSE	RMSE
1	0.0071281	226.278 %	149.743%	0.0001659	0.0129
2	0.0185368	1285.321%	162.623%	0.0005445	0.0233
3	0.0170955	1269.167%	152.241%	0.0004301	0.0207
4	0.0125017	890.102 %	143.206%	0.0002785	0.0167
5	0.0123816	880.922 %	142.846%	0.0002734	0.0165
6	0.0114298	791.442 %	140.674%	0.0002506	0.0158
7	0.0120357	845.513 %	142.364%	0.0002645	0.0163
8	0.0129239	927.041 %	144.476%	0.0002873	0.0169
9	0.0127585	905.226 %	144.252%	0.0002825	0.0168
10	0.0126607	891.926 %	143.986%	0.0002806	0.0167
11	0.0119865	832.990 %	142.376%	0.0002635	0.0162
12	0.0117165	798.489 %	141.653%	0.0002570	0.0160
13	0.0116267	787.114 %	141.432%	0.0002550	0.0160
14	0.0111193	747.410 %	140.006%	0.0002430	0.0156
15	0.0114737	780.215 %	140.884%	0.0002517	0.0159
16	0.0115810	789.367 %	141.229%	0.0002541	0.0159
17	0.0122861	857.448 %	143.114%	0.0002710	0.0165
18	0.0125164	883.829 %	143.721%	0.0002766	0.0166
19	0.0109282	737.403 %	139.530%	0.0002386	0.0154
20	0.0103418	681.046 %	137.796%	0.0002258	0.0150
21	0.0106389	710.246 %	138.630%	0.0002322	0.0152
22	0.0109151	736.288 %	139.435%	0.0002383	0.0154
23	0.0108950	733.836 %	139.386%	0.0002378	0.0154
24	0.0111957	761.162 %	140.179%	0.0002446	0.0156
25	0.0115122	790.003 %	141.057%	0.0002519	0.0159
26	0.0116863	807.025 %	141.500%	0.0002560	0.0160
27	0.0119038	824.726 %	142.045%	0.0002613	0.0162
28	0.0121591	848.943 %	142.699%	0.0002677	0.0164
29	0.0123041	863.794 %	143.059%	0.0002713	0.0165
30	0.0123163	866.428 %	143.149%	0.0002717	0.0165

Table 32. SVR results with n number of previous returns as features

n prev. returns	MAE	MAPE	sMAPE	MSE	RMSE
1	0.0078907	336.644%	142.682%	0.000174	0.0132
2	0.0077225	318.936%	144.688%	0.000170	0.0131
3	0.0080221	349.156%	144.886%	0.000174	0.0132
4	0.0088102	433.743%	148.897%	0.000188	0.0137
5	0.0097488	538.073%	149.077%	0.000207	0.0144
6	0.0097202	537.465%	148.412%	0.000206	0.0143
7	0.0102297	589.310%	148.649%	0.000216	0.0147
8	0.0102767	595.910%	148.678%	0.000217	0.0147
9	0.0109943	669.755%	150.284%	0.000232	0.0152
10	0.0114400	715.953%	150.495%	0.000242	0.0156
11	0.0118227	754.184%	151.028%	0.000251	0.0158
12	0.0121488	785.978%	151.561%	0.000259	0.0161
13	0.0124328	812.874%	152.135%	0.000265	0.0163
14	0.0126798	836.020%	152.629%	0.000271	0.0165
15	0.0128941	855.792%	153.073%	0.000277	0.0166
16	0.0130829	872.963%	153.475%	0.000281	0.0168
17	0.0132521	888.585%	153.846%	0.000286	0.0169
18	0.0133995	901.841%	154.152%	0.000289	0.0170
19	0.0135305	913.591%	154.418%	0.000293	0.0171
20	0.0136539	924.839%	154.732%	0.000296	0.0172
21	0.0137587	934.191%	154.939%	0.000299	0.0173
22	0.0138539	942.619%	155.134%	0.000301	0.0174
23	0.0139399	950.192%	155.313%	0.000304	0.0174
24	0.0140201	956.250%	155.457%	0.000306	0.0175
25	0.0140917	962.508%	155.607%	0.000308	0.0175
26	0.0141570	968.213%	155.742%	0.000310	0.0176
27	0.0142170	971.503%	155.823%	0.000311	0.0176
28	0.0142708	976.186%	155.933%	0.000313	0.0177
29	0.0143196	980.432%	156.032%	0.000314	0.0177
30	0.0143594	984.980%	156.082%	0.000315	0.0178

Table 33. KNN results with n number of previous returns as features

n prev. returns	MAE	MAPE	sMAPE	MSE	RMSE
1	0.0047606	369.925%	144.315%	0.0000641	0.0080
2	0.0046811	357.278%	147.982%	0.0000621	0.0079
3	0.0045044	371.885%	143.125%	0.0000561	0.0075
4	0.0045869	290.439%	145.998%	0.0000579	0.0076
5	0.0045283	310.523%	145.741%	0.0000598	0.0077
6	0.0044336	297.080%	144.074%	0.0000580	0.0076
7	0.0043958	281.959%	146.275%	0.0000575	0.0076
8	0.0044230	259.937%	148.532%	0.0000607	0.0078
9	0.0043117	270.017%	143.964%	0.0000572	0.0076
10	0.0042986	243.104%	143.730%	0.0000623	0.0079
11	0.0043377	233.440%	144.597%	0.0000626	0.0079
12	0.0043024	249.190%	145.794%	0.0000610	0.0078
13	0.0042876	232.070%	144.274%	0.0000612	0.0078
14	0.0043653	253.888%	147.133%	0.0000614	0.0078
15	0.0043717	271.345%	148.517%	0.0000632	0.0079
16	0.0043432	249.885%	147.773%	0.0000632	0.0079
17	0.0043217	279.719%	146.293%	0.0000612	0.0078
18	0.0043395	262.353%	145.598%	0.0000617	0.0079
19	0.0043209	270.228%	147.161%	0.0000610	0.0078
20	0.0043772	278.542%	149.514%	0.0000621	0.0079
21	0.0043470	267.713%	147.895%	0.0000607	0.0078
22	0.0043425	256.855%	148.240%	0.0000623	0.0079
23	0.0043027	244.428%	148.984%	0.0000599	0.0077
24	0.0042582	252.819%	149.057%	0.0000596	0.0077
25	0.0042596	241.657%	147.082%	0.0000606	0.0078
26	0.0043107	236.553%	148.348%	0.0000607	0.0078
27	0.0042963	264.332%	147.221%	0.0000600	0.0077
28	0.0043111	252.394%	146.838%	0.0000613	0.0078
29	0.0042948	265.302%	147.454%	0.0000614	0.0078
30	0.0043344	282.993%	144.467%	0.0000617	0.0079

Table 34. KNN results with n number of previous returns, high, low, open and close values as features

n prev. returns	MAE	MAPE	sMAPE	MSE	RMSE
1	0.0043319	377.420%	138.089%	0.0000449	0.0067
2	0.0043319	377.420%	138.089%	0.0000449	0.0067
3	0.0043319	377.420%	138.089%	0.0000449	0.0067
4	0.0043386	377.459%	138.461%	0.0000450	0.0067
5	0.0043386	377.459%	138.461%	0.0000450	0.0067
6	0.0043386	377.459%	138.461%	0.0000450	0.0067
7	0.0043431	377.931%	138.677%	0.0000450	0.0067
8	0.0043431	377.931%	138.677%	0.0000450	0.0067
9	0.0043431	377.931%	138.677%	0.0000450	0.0067
10	0.0043427	378.271%	138.483%	0.0000450	0.0067
11	0.0043427	378.271%	138.483%	0.0000450	0.0067
12	0.0043427	378.271%	138.483%	0.0000450	0.0067
13	0.0043445	378.317%	138.578%	0.0000450	0.0067
14	0.0043453	378.375%	138.480%	0.0000450	0.0067
15	0.0043453	378.375%	138.480%	0.0000450	0.0067
16	0.0043453	378.375%	138.480%	0.0000450	0.0067
17	0.0043474	378.335%	138.461%	0.0000451	0.0067
18	0.0043474	378.335%	138.461%	0.0000451	0.0067
19	0.0043474	378.335%	138.461%	0.0000451	0.0067
20	0.0043496	378.377%	138.364%	0.0000451	0.0067
21	0.0043496	378.377%	138.364%	0.0000451	0.0067
22	0.0043496	378.377%	138.364%	0.0000451	0.0067
23	0.0043496	378.377%	138.364%	0.0000451	0.0067
24	0.0043529	378.722%	138.437%	0.0000452	0.0067
25	0.0043529	378.722%	138.437%	0.0000452	0.0067
26	0.0043529	378.722%	138.437%	0.0000452	0.0067
27	0.0043569	379.016%	138.522%	0.0000452	0.0067
28	0.0043569	379.016%	138.522%	0.0000452	0.0067
29	0.0043569	379.016%	138.522%	0.0000452	0.0067
30	0.0043565	379.253%	138.437%	0.0000452	0.0067

Table 35. GARCH orders of p and q test results

p	q	MAE	MAPE	sMAPE	MSE	RMSE
1	1	4.16E-06	66.972	51.050	2.04E-10	1.43E-05
1	2	6.16E-06	106.751	57.429	3.92E-10	1.98E-05
1	3	6.33E-06	115.524	59.001	3.91E-10	1.98E-05
1	4	6.40E-06	124.390	60.561	3.72E-10	1.93E-05
1	5	6.46E-06	133.085	61.555	3.84E-10	1.96E-05
2	1	3.97E-06	72.304	52.018	1.33E-10	1.15E-05
2	2	8.43E-06	209.655	71.817	4.28E-10	2.07E-05
2	3	5.89E-06	122.275	60.096	2.67E-10	1.63E-05
2	4	5.94E-06	130.968	61.220	2.50E-10	1.58E-05
2	5	5.97E-06	139.548	62.243	2.44E-10	1.56E-05
3	1	8.02E-06	208.194	72.160	4.14E-10	2.03E-05
3	2	8.78E-06	220.065	73.488	4.74E-10	2.18E-05
3	3	6.51E-06	131.500	62.271	3.31E-10	1.82E-05
3	4	6.51E-06	140.150	63.295	3.10E-10	1.76E-05
3	5	6.54E-06	148.150	64.128	2.99E-10	1.73E-05
4	1	0.002645	162571.037	198.958	7.00E-06	0.002646
4	2	9.04E-06	223.067	74.103	5.16E-10	2.27E-05
4	3	6.84E-06	136.827	63.606	3.81E-10	1.95E-05
4	4	6.81E-06	144.743	64.322	3.59E-10	1.89E-05
4	5	6.86E-06	152.805	65.045	3.47E-10	1.86E-05
5	1	8.39E-06	218.103	73.642	4.62E-10	2.15E-05
5	2	9.26E-06	228.399	74.863	5.38E-10	2.32E-05
5	3	7.03E-06	142.724	64.977	3.93E-10	1.98E-05
5	4	7.00E-06	150.126	65.530	3.74E-10	1.93E-05
5	5	7.00E-06	157.930	66.232	3.58E-10	1.89E-05

Table 36. GARCH-ANN relu layer activation function results

Optimiser	MAE	MAPE	sMAPE	MSE	RMSE
rmsprop	3.12E-06	124.7022	67.33394	2.81E-10	1.68E-05
adam	2.95E-06	88.68045	60.7522	2.83E-10	1.68E-05
adadelta	2.93E-06	80.97395	59.93846	2.84E-10	1.68E-05
adagrad	2.93E-06	78.34828	59.81356	2.84E-10	1.69E-05
adamax	3.01E-06	104.7715	63.38418	2.82E-10	1.68E-05
nadam	2.93E-06	83.11572	60.14456	2.84E-10	1.68E-05
ftrl	4.25E-05	3079.608	200	2.09E-09	4.57E-05

Table 37. GARCH-ANN sigmoid layer activation function results

Optimiser	MAE	MAPE	sMAPE	MSE	RMSE
rmsprop	4.17E-06	115.5875	200	2.97E-10	1.72E-05
adam	2.98E-06	62.8196	61.86965	2.86E-10	1.69E-05
adadelta	2.93E-06	81.31975	60.00108	2.84E-10	1.69E-05
adagrad	2.93E-06	77.43968	59.78577	2.84E-10	1.69E-05
adamax	3.01E-06	60.41674	63.48281	2.87E-10	1.69E-05
nadam	3.05E-06	112.6613	64.87926	2.82E-10	1.68E-05
ftrl	2.95E-06	68.75039	60.20398	2.85E-10	1.69E-05

Table 38. GARCH-ANN softmax layer activation function results

Optimiser	MAE	MAPE	sMAPE	MSE	RMSE
rmsprop	3.79E-06	209.0068	83.89194	2.79E-10	1.67E-05
adam	2.94E-06	69.6599	60.08016	2.85E-10	1.69E-05
adadelta	2.94E-06	86.32181	60.48747	2.84E-10	1.68E-05
adagrad	2.95E-06	88.21622	60.71242	2.83E-10	1.68E-05
adamax	2.94E-06	71.64111	59.85888	2.85E-10	1.69E-05
nadam	2.96E-06	65.20224	60.98233	2.86E-10	1.69E-05
ftrl	2.94E-06	85.69063	60.41954	2.84E-10	1.68E-05

Table 39. GARCH-ANN softplus layer activation function results

Optimiser	MAE	MAPE	sMAPE	MSE	RMSE
rmsprop	3.42E-06	164.3105	75.64893	2.80E-10	1.67E-05
adam	2.96E-06	92.10317	61.2234	2.83E-10	1.68E-05
adadelta	2.93E-06	78.49231	59.82324	2.84E-10	1.69E-05
adagrad	3.02E-06	107.9957	63.98045	2.82E-10	1.68E-05
adamax	2.93E-06	80.29211	59.91874	2.84E-10	1.69E-05
nadam	3.27E-06	145.7434	71.79555	2.81E-10	1.68E-05
ftrl	2.95E-06	90.86379	61.05119	2.83E-10	1.68E-05

Table 40. GARCH-ANN softsign layer activation function results

Optimiser	MAE	MAPE	sMAPE	MSE	RMSE
rmsprop	3.26E-06	58.66021	81.8609	2.90E-10	1.70E-05
adam	2.93E-06	79.36901	59.83047	2.84E-10	1.69E-05
adadelta	2.94E-06	82.87181	60.14216	2.84E-10	1.68E-05
adagrad	2.94E-06	83.9401	60.24264	2.84E-10	1.68E-05
adamax	3.05E-06	112.8718	64.90179	2.82E-10	1.68E-05
nadam	2.93E-06	76.71803	59.73889	2.84E-10	1.69E-05
ftrl	4.26E-05	3086.755	200	2.10E-09	4.58E-05

Table 41. GARCH-ANN tanh layer activation function results

Optimiser	MAE	MAPE	sMAPE	MSE	RMSE
rmsprop	3.05E-06	113.3224	65.00926	2.82E-10	1.68E-05
adam	3.14E-06	57.48152	72.32692	2.88E-10	1.70E-05
adadelta	2.93E-06	81.68106	60.03393	2.84E-10	1.69E-05
adagrad	2.93E-06	72.73026	59.78111	2.85E-10	1.69E-05
adamax	2.94E-06	71.05777	59.89117	2.85E-10	1.69E-05
nadam	3.12E-06	124.7693	67.33987	2.81E-10	1.68E-05
ftrl	4.25E-05	3081.122	200	2.09E-09	4.57E-05

Table 42. GARCH-ANN selu layer activation function results

Optimiser	MAE	MAPE	sMAPE	MSE	RMSE
rmsprop	3.17E-06	57.59447	74.99062	2.89E-10	1.70E-05
adam	2.99E-06	61.72146	62.40456	2.86E-10	1.69E-05
adadelta	2.93E-06	78.83218	59.86649	2.84E-10	1.69E-05
adagrad	2.93E-06	72.7798	59.80047	2.85E-10	1.69E-05
adamax	2.93E-06	83.17015	60.13983	2.84E-10	1.68E-05
nadam	3.28E-06	147.8645	72.22635	2.80E-10	1.67E-05
ftrl	4.21E-05	3047.316	200	2.05E-09	4.53E-05

Table 43. GARCH-ANN elu layer activation function results

Optimiser	MAE	MAPE	sMAPE	MSE	RMSE
rmsprop	3.53E-06	178.9312	78.44511	2.80E-10	1.67E-05
adam	2.94E-06	85.03473	60.31237	2.83E-10	1.68E-05
adadelta	2.94E-06	82.85015	60.13854	2.84E-10	1.68E-05
adagrad	2.93E-06	80.13859	59.91075	2.84E-10	1.69E-05
adamax	3.07E-06	116.181	65.57151	2.82E-10	1.68E-05
nadam	2.94E-06	68.60321	60.18833	2.85E-10	1.69E-05
ftrl	4.24E-05	3069.495	200	2.08E-09	4.56E-05

Table 44. GARCH-ANN exponential layer activation function results

Optimiser	MAE	MAPE	sMAPE	MSE	RMSE
rmsprop	4.06E-06	238.4027	88.72346	2.79E-10	1.67E-05
adam	2.96E-06	93.92799	61.49769	2.83E-10	1.68E-05
adadelta	2.93E-06	82.2518	60.08522	2.84E-10	1.69E-05
adagrad	3.22E-06	58.05996	78.79319	2.89E-10	1.70E-05
adamax	3.00E-06	61.28713	62.76268	2.87E-10	1.69E-05
nadam	2.96E-06	66.34733	60.66262	2.86E-10	1.69E-05
ftrl	2.94E-06	69.03516	60.16305	2.85E-10	1.69E-05

II. Charts

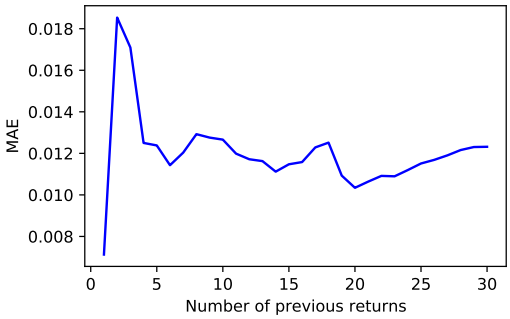

Figure 28. SVR nr of previous returns MAE results

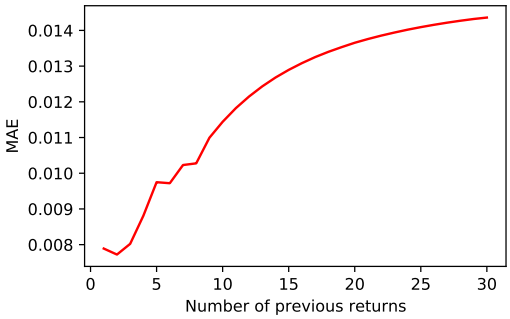

Figure 29. SVR all features MAE results

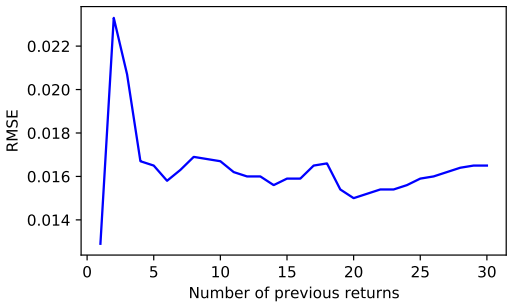

Figure 30. SVR nr of previous returns RMSE results

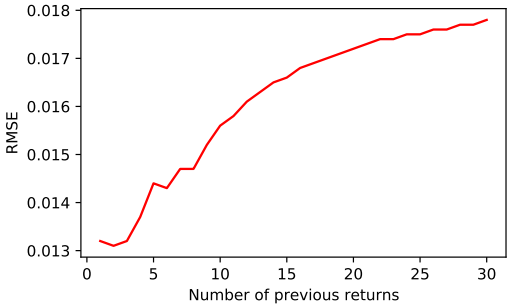

Figure 31. SVR all features RMSE results

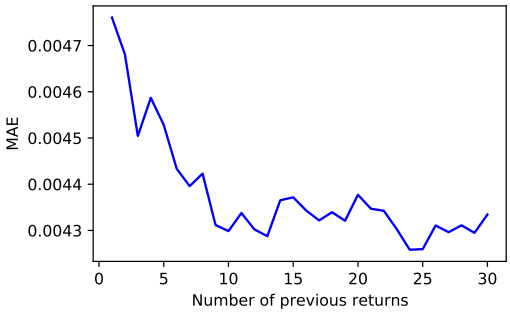

Figure 32. KNN nr of previous returns MAE results

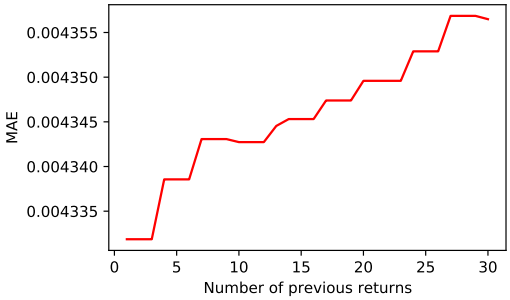

Figure 33. KNN all features MAE results

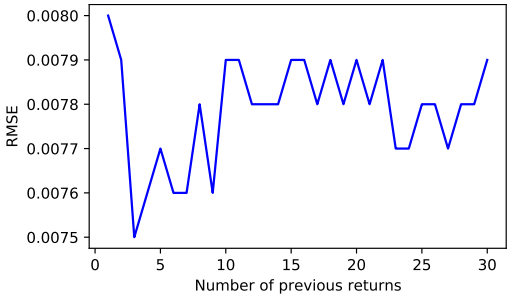

Figure 34. KNN nr of previous returns RMSE results

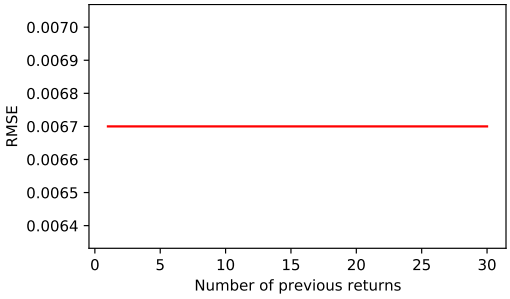

Figure 35. KNN all features RMSE results

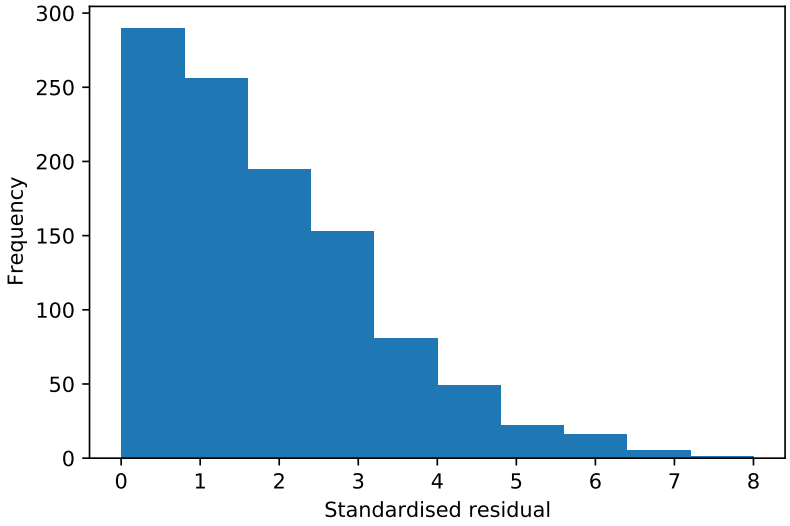

Figure 36. Standardised residuals of econometric models

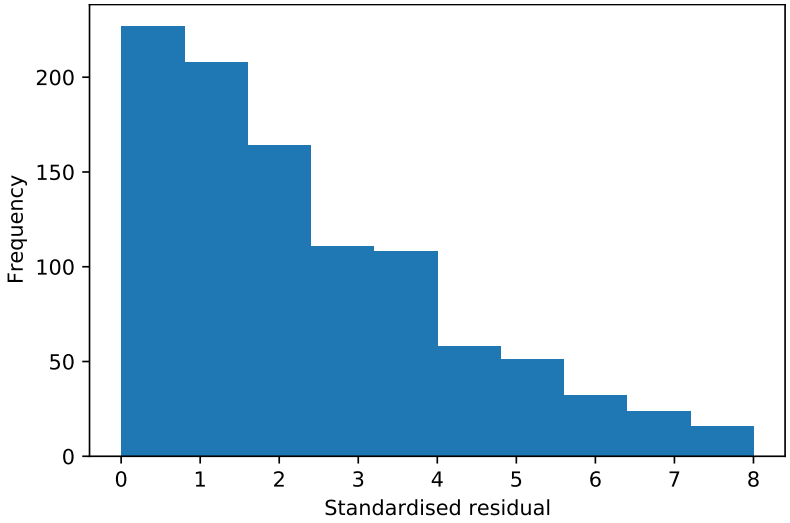

Figure 37. Standardised residuals of random forest and GARCH-ANN

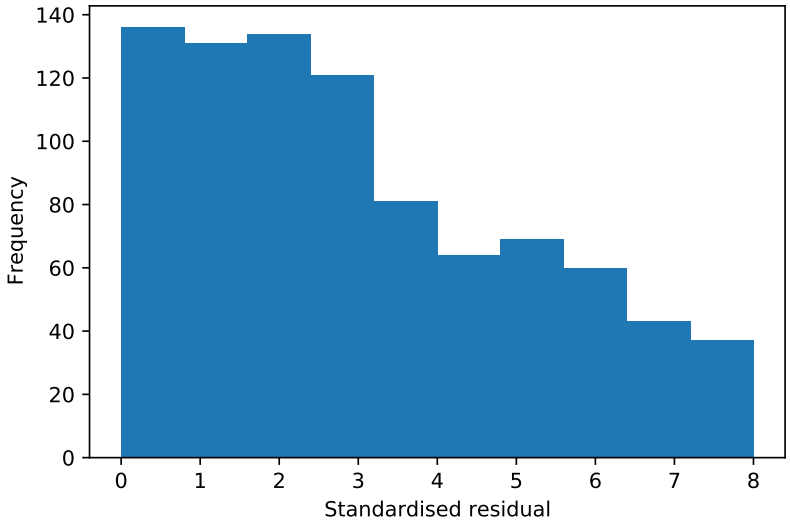

Figure 38. Standardised residuals of SVR and GARCH-ANN

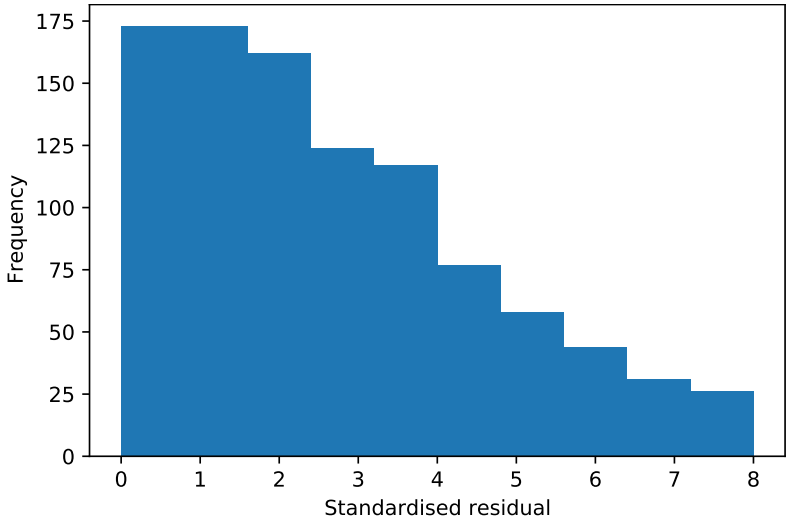

Figure 39. Standardised residuals of KNN and GARCH-ANN

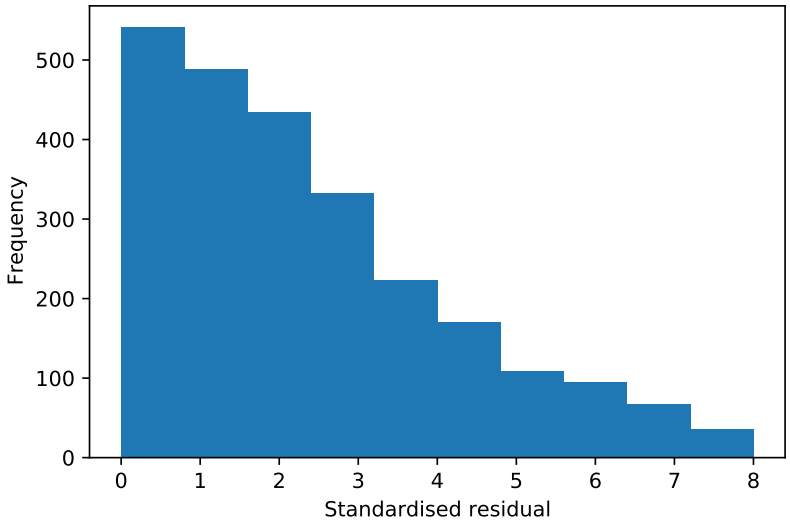

Figure 40. Standardised residuals of econometric models with the sliding window method

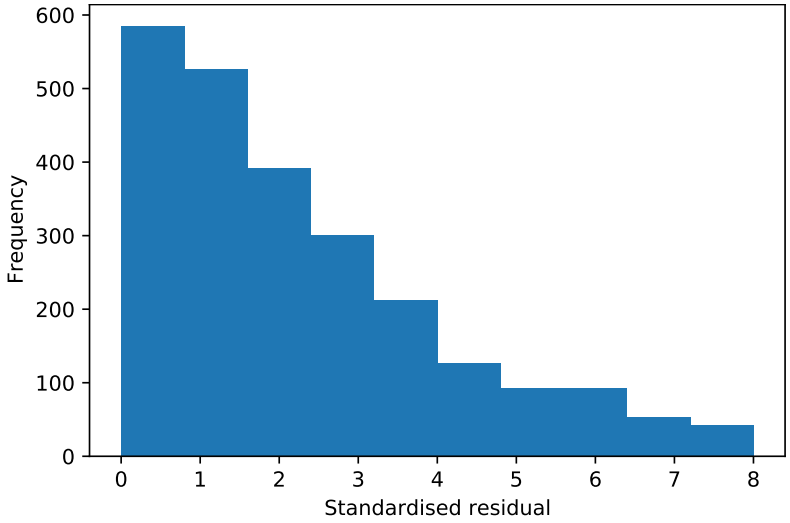

Figure 41. Standardised residuals of random forest and GARCH-ANN with the sliding window method

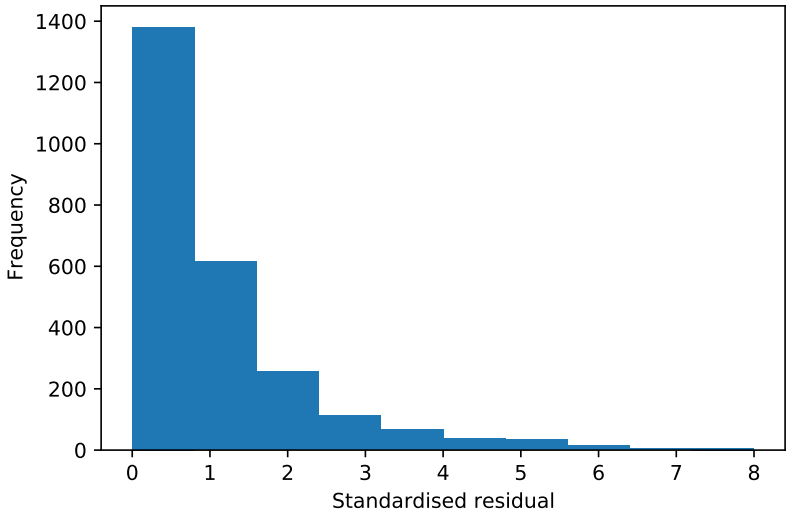

Figure 42. Standardised residuals of SVR and GARCH-ANN with the sliding window method

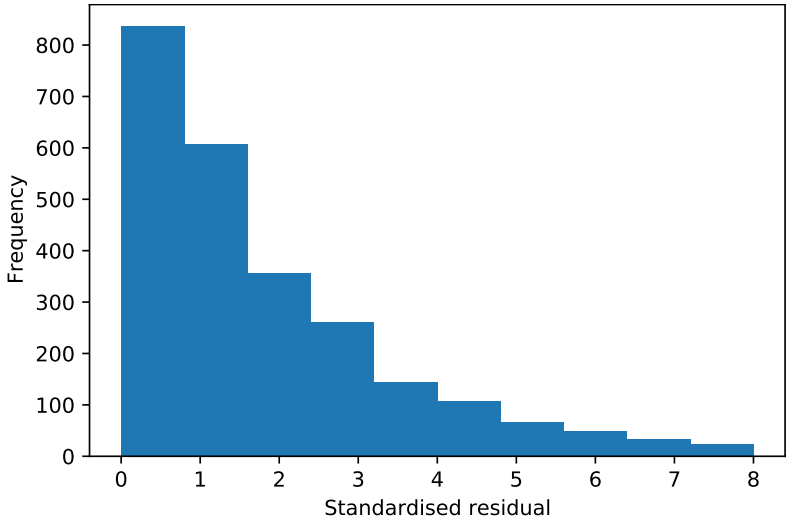

Figure 43. Standardised residuals of KNN and GARCH-ANN with the sliding window method

III. Licence

Non-exclusive licence to reproduce thesis and make thesis public

I, **Anders Nõu**,

1. herewith grant the University of Tartu a free permit (non-exclusive licence) to

   reproduce, for the purpose of preservation, including for adding to the DSpace digital archives until the expiry of the term of copyright,

   Predicting stock return and volatility with machine learning and econometric models — a comparative case study of the Baltic stock market,

   supervised by Rajesh Sharma, Mustafa Hakan Eratalay and Darya Lapitskaya.

2. I grant the University of Tartu a permit to make the work specified in p. 1 available to the public via the web environment of the University of Tartu, including via the DSpace digital archives, under the Creative Commons licence CC BY NC ND 3.0, which allows, by giving appropriate credit to the author, to reproduce, distribute the work and communicate it to the public, and prohibits the creation of derivative works and any commercial use of the work until the expiry of the term of copyright.

3. I am aware of the fact that the author retains the rights specified in p. 1 and 2.

4. I certify that granting the non-exclusive licence does not infringe other persons' intellectual property rights or rights arising from the personal data protection legislation.

Anders Nõu
07/05/2021