UNIVERSITY OF LJUBLJANA
SCHOOL OF ECONOMICS AND BUSINESS

MASTER’S THESIS

AN EMPIRICAL INVESTIGATION OF FORECASTING STOCK MARKET VOLATILITY WITH SUPPORT VECTOR MACHINES

Ljubljana, July 2020

CHENG YAO

AUTHORSHIP STATEMENT

The undersigned Cheng Yao, a student at the University of Ljubljana, School of Economics and Business, (hereafter: SEB LU), author of this written final work of studies with the title AN EMPIRICAL INVESTIGATION OF FORECASTING STOCK MARKET VOLATILITY WITH SUPPORT VECTOR MACHINES, prepared under supervision of prof. dr. Igor Masten.

DECLARE

1. this written final work of studies to be based on the results of my own research;

2. the printed form of this written final work of studies to be identical to its electronic form;

3. the text of this written final work of studies to be language-edited and technically in adherence with the SEB LU's Technical Guidelines for Written Works, which means that I cited and / or quoted works and opinions of other authors in this written final work of studies in accordance with the SEB LU's Technical Guidelines for Written Works;

4. to be aware of the fact that plagiarism (in written or graphical form) is a criminal offence and can be prosecuted in accordance with the Criminal Code of the Republic of Slovenia;

5. to be aware of the consequences a proven plagiarism charge based on the this written final work could have for my status at the SEB LU in accordance with the relevant SEB LU Rules;

6. to have obtained all the necessary permits to use the data and works of other authors which are (in written or graphical form) referred to in this written final work of studies and to have clearly marked them;

7. to have acted in accordance with ethical principles during the preparation of this written final work of studies and to have, where necessary, obtained permission of the Ethics Committee;

8. my consent to use the electronic form of this written final work of studies for the detection of content similarity with other written works, using similarity detection software that is connected with the SEB LU Study Information System;

9. to transfer to the University of Ljubljana free of charge, non-exclusively, geographically and time-wise unlimited the right of saving this written final work of studies in the electronic form, the right of its reproduction, as well as the right of making this written final work of studies available to the public on the World Wide Web via the Repository of the University of Ljubljana;

10. my consent to publication of my personal data that are included in this written final work of studies and in this declaration, when this written final work of studies is published.

Ljubljana, September 7th, 2020 Author's signature:

(Month in words / Day / Year,
e.g. June 1st, 2012)

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

GARCH - Generalized Autoregressive Conditional Heteroskedatic
MLE - Maximum Likelihood Estimation
SVM - Support Vector Machine
ARCH - Autoregressive Conditional Heteroskedastic
SVR - Support Vector Regression
ν-SVR - New Support Vector Regression
AR - Autoregressive
MA - Moving Average
ARMA - Autoregressive Moving Average
ANN - Artificial Neural Network
EGARCH - Exponential Generalized Autoregressive Conditional Heteroscedastic
ACF - Autocorrelation Function
PACF - Partial Autocorrelation Function
MSLE - Mean Squared Logarithmic Error
TSA - Time Series Analysis

INTRODUCTION

Purpose of the Thesis

Volatility forecasting has significant importance in academic studies and empirical use. It is one of the most challenging tasks in the financial system. Volatility is calculated from the standard deviation or variance of the closing prices. There are various models studied and applied on financial time series for variance forecasting in purpose of modelling asset returns, portfolio optimization, risk management, etc. The Generalized Autoregressive Conditional Heteroskedastic (GARCH) model (Bollerslev, 1986) is one of the most studied and applied models for volatility forecasting. It models the autoregressive process, based on lagged squared returns and variance. The GARCH(1,1) model is mainly used in this work for variance forecasting.

Financial time series has certain characteristics. There are mainly zero mean in return series. The squared returns usually have strong autocorrelations. Regarding modelling conditional variance, the GARCH model is suitable for the estimation process. The parameters in the statistic model are usually estimated by Maximum Likelihood Estimation (MLE) method. The method has assumptions for error distribution. We usually assume normal distribution and student's t distribution when using the method. At the same time, it is known that financial time series doesn't have normal distribution empirically and usually with heavy tails. In this thesis, besides the MLE method, we use Support Vector Machine (SVM) which doesn't need an assumption of error distribution. Under the framework of the GARCH model, we compare the forecasted variance with true variance and examine the effect of different methods.

Structure of the Thesis

The first part of the thesis focuses on the theoretical background. It starts from time series analysis and introduces different models including the autoregressive model, the moving average model and the conditional heteroscedastic models - mainly the Autoregressive Conditional Heteroskedastic (ARCH) model (Engle, 1982) and the GARCH model. We then introduce SVM by presenting basic theory and formulations of SVM classification and regression as well as the main features in SVM applications which are dual formulation, kernels and grid search. After the introduction of time series models and SVM, we proceed with parameter estimation for the GARCH(1,1) model. For MLE method, we assume normal and student's t distributions for the error terms and the formulations under both assumptions. For SVR method, we demonstrate the formulations of the New Support Vector Regression (ν-SVR) (Schölkopf, Smola, Williamson & Bartlett, 2000), approximate functions and how to get the estimated parameters in the GARCH model in the case of linear kernel. To finish the theoretical part, we present the evaluation of volatility forecasting -

using the coefficient of determination (R^2) between the forecasted and true variances.

The second part contains the empirical modelling process and results. It starts with the description and transformation of empirical data that we use in this work. Before the empircal modelling, we also provide a simulation process. We simulate data series with specified parameters in the GARCH model and we compare the forecastibility R^2 between GARCH-MLE and GARCH-SVR. Then we proceed with the empirical data - SP500 and BTC/USD daily returns. We first apply time series ananlysis and normality test on the data to examine autocorrelations, return distributions, etc. We then present the empirical result showing that the GARCH-MLE model outperforms the GARCH-SVR model for SP500 daily returns, which is different from the simulation process with similar parameter settings. The mixed result also shows that the GARCH-SVR model performs better for the BTC returns.

Previous Research and Literature Overview

There are certain characteristics of volatility of asset returns. The most seen and prominent stylized facts about volatility are persistence, mean reverting and an asymmetric effect from innovations on volatility (Engle & Patton, 2001). From empirical modelling, volatility has calm periods followed by more clustering periods. Volatility is not diverging to infinity but mean reverting. Besides, volatility reacts stronger to negative shocks than an equal size of positive shocks, which is also referred to as the leverage effect (Black, 1976).

Conventional econometric models have assumptions of constant variances. The ARCH model introduces the conditional variance changing over time based on past errors and leaving the unconditional variance constant. The ARCH model has widely been used and proven useful in economic and financial modelling (Poon & Granger, 2003). Models for the inflation rate (Engle, 1983) are constructed to recognize the uncertainty of inflation that tends to change over time. The ARCH model and a simple regression model also provide good performance of volatility forecasting (Brailsford & Faff, 1996).

In 1986, Bollerslev introduces the GARCH model. It is a more general class of process. It extends the ARCH model with lagged conditional variances, meaning with resemblance of the standard time series Autoregressive (AR) process to the general Autoregressive Moving Average (ARMA) process (Whittle, 1951).

The GARCH model has similar properties as the ARCH model. It is widely used in the application of inflations, exchange rates, stock markets, etc. It can model the volatility clustering but cannot address the leverage effect and asymmetric impact from positive and negative returns. The addition of conditional variances helps to model the volatility without adding many lagged squared returns compared to the ARCH model. However, studies show that the GARCH model is inferior to models that can capture a leverage effect (Hansen & Lunde, 2005).

Generally, the parameters in the GARCH model are estimated with the Maximum Likelihood Estimation (MLE) method by taking the conditionally Gaussian log-likelihood and maximizing the likelihood function. In this case, we need an assumption for error distribution - normal or student's t (Dutta, 2014). There are also other methods for estimating parameters in the GARCH model, for example using non-linear modifications (Franses & Dijk, 1996).

Support Vector Machine (SVM) (Cortes & Vapnik, 1995) is characterized by number of support vectors, the cost functions, usage of kernel tricks, mapping into high dimensional feature space etc. It can be applied to solve classification and regression problems. The algorithm is widely used in categorization, image recognition, biological classification, etc. Same as the classification approach, Support Vector Regression (SVR) (Drucker et al., 1996) optimizes the generalization bounds with a given trade-off.

The ν-SVR introduces a new parameter $\nu \in (0, 1]$. It includes an upper bound on a fraction of training errors and a lower bound of fraction of support vectors. It controls the number of support vectors and training errors.

The benefits of using SVM or SVR, is that there is no assumption of a probability density function of the error distributions when estimating parameters in the GARCH(1,1) model (Gavrishchaka & Banerjee, 2006). The SVM methods use the empirical risk minimization inductive principle. It looks for an insensitivity zone and a decision boundary. It is a constrained optimization problem and can be solved using quadratic programming schemes. Instead of an assumption of error distribution as Gaussian or student's t, it can bring a better result of forecastibility especially when error does not have normal or standard student's t distribution (Chen, Jeong & Härdle, 2008).

There are studies demonstrating the effects of different hybrid GARCH models. In 1996, Donaldson and Kamstra construct a seminonparametric nonlinear GARCH model based on the Artificial Neural Network (ANN). They reveal that the hybrid model captures the volatility effects overlooked by the GARCH model. The GARCH-SVR model has consistent and stronger parameter estimation and variance forecastibility in empirical study (Perez-Cruz, Afonso-Rodriguez & Giner, 2003). The hybrid model also shows a significant empirical result for forecasting one-period-ahead volatility that outperforms standard GARCH, exponential GARCH (EGARCH) and the ANN-GARCH model (Chen, Hädle & Jeong, 2009). A brief literature overview is listed in table1.

Table 1: Literature overview

Title	Authors and Year	Description
Studies of Stock Price Volatility Changes	(Black, 1976)	Volatility model: Black's leverage effect
What Good is a Volatility Model	(Engle & Patton, 2001)	Volatility model: the most seen and prominent stylized facts
Modelling and Forecasting Realized Volatility	(Anderson, Bollerslev, Diebold & Labys, 2003)	Modelling realized volatility
Hypothesis Testing in Time Series Analysis	(Whittle, 1951)	The ARMA model
General Autoregressive Conditional Heteroskedasticity	(Bollerslev, 1986)	The GARCH model
Autoregressive Conditional Heteroscedasticity with Estimates of the Variance of United Kingdom Inflation	(Engle, 1982)	The ARCH model
Estimates of the Variance of U.S. Inflation Based upon the ARCH Model	(Engle, 1983)	The ARCH model: uncertainty of inflation changing over time
The Nature of Statistical Learning Theory	(Vapnik, 1995)	Support Vector Machine model
Support-Vector Networks	(Cortes & Vapnik, 1995)	Support Vector Machine model
Support Vector Regression Machines	(Drucker, Burges, Kaufman, Smola & Vapnik, 1997)	Support Vector Regression model

Table continues

Table 1: Literature overview (cont.)

Title	**Authors and Year**	**Description**
New Support Vector Algorithms	(Schölkopf, Smola, Williamson & Bartlett, 2000)	New Support Vector model (ν-SVR)
An Artificial Neural Network-GARCH Model for International Stock Return Volatility	(Donaldson & Kamstra, 1996)	The hybrid model ANN-GARCH
Financial Forecasting using Support Vector Machines	(Cao & Tay, 2001)	SVM used in financial forecasting
Estimating GARCH Models Using Support Vector Machines	(Perez-Cruz, Afonso-Rodriguez & Giner, 2003)	Estimating GARCH using SVM
Forecasting Volatility with Support Vector Machine-Based GARCH Model	(Chen, Härdle & Jeong, 2009)	Volatility forecasting using the GARCH, EGARCH, ANN-GARCH and SVM-GARCH models
Modelling and Forecasting Stock Market Volatility by Gaussian Process based on GARCH, EGARCH and GJR model	(Ou & Wang, 2011)	Predicting volatility using Gaussian process
Volatility Forecasting via SVR-GARCH with Mixture of Gaussian Kernels	(Bezerra & Albuquerque, 2017)	Volatility forecasting using GARCH-SVR with mixture of gaussian kernels

Source: Own work.

1 THEORETICAL BACKGROUND

The first part of the thesis demonstrates the theoretical background with four chapters: time series analysis, conditional heteroscedastic models, SVM models and the parameter estimation in GARCH(1,1). We start with the characteristics of time series analysis in financial assets and cover main models including the autoregressive model, moving average model and the ARMA model. We then proceed with the ARCH and GARCH models. In the third chapter, we discuss SVM classification and SVM regression and some basic concepts when applying SVM models - dual formulations, using kernels and grid search. We then explain the formulations for parameter estimation in the GARCH(1,1) model. When using

MLE, we use 2 error distribution assumptions - normal and student's t distributions. And when using SVR, we demonstrate how parameters in the GARCH(1,1) model are estimated from SVR approximate functions in case of applying a linear kernel.

1.1 Time Series Analysis

Time series analysis is for analyzing data in order to get useful information and meaningful statistics from the data. There is a natural temporal ordering in time series analysis that is different from cross-sectional studies. Time series forecasting uses models to predict values in the future based on observed data (Tsay, 2010).

In this section we will start with some characteristics of time series analysis. To get familiar with some basic concepts that are referred throughout the thesis, we cover the topics of the autoregressive (AR) model, the moving average (MA) model and the ARMA model.

1.1.1 Characteristics of Time Series

Stationarity

In time series analysis, stationarity means the statistical properties - mean, variance, autocorrelation don't change over time. Through some use of transformations, time series can be stationalized and it can be used to predict future values (Sollis, 2012). In other words, time series after transformation is without the trend, seasonality and with constant variance and autocorrelation. The purpose of getting a stationary time series is that we want useful statistics when forecasting future values. For example, if time series is not stationary and its mean and variance increase with sample size, consequently we will underestimate the mean and variance in the forecasting values.

Let a time series be $\{r_t\}$, the unconditional joint distribution of $(r_{t_1}, r_{t_2}, ...r_{t_k})$ is identical to the joint distribution of $(r_{t_1+t}, r_{t_2+t}, ...r_{t_k+t})$ for all t, where k is an arbitrary positive integer and $(t_1, ..., t_k)$ is a collection of k positive integers. This is so called strictly stationary. It is a very hard condition to meet empirically, meaning that the joint distribution of random variables remains the same while shifting time index.

A weak stationary time series only requires that the first moment (i.e. the mean) and autocovariance don't change over time and that the second moment is finite for all time. In mathematical terms, $E(r_t) = \mu$ and $Cov(r_t, r_{t-l}) = \gamma_l$, where μ is constant and $Cov_(r_t, r_{t-l})$ only depends on l not on t, where l is an arbitrary integer. It is common that we assume time series is weak stationary for financial assets and we use the weak stationary assumption for this thesis.

Autocorrelation

Autocorrelation in financial time series helps to check if returns r_t is autocorrelated with previous values. The Autocorrelation Function (ACF) estimates the correlation coefficient between r_t and r_{t-l}. The sample lag-l autocorrelation denoted by ρ_l is given by

$$\rho_l = \frac{Cov(r_t, r_{t-l})}{Var(r_t)Var(r_{t-l})} = \frac{Cov(r_t, r_{t-l})}{Var(r_t)} \tag{1}$$

From the assumptions in weak stationarity, the property $Var(r_t) = Var(r_{t-l})$ holds. Series r_t is not autocorrelated if and only if $\rho_l = 0$. Considering a sample of $\{r_t\}_{t=1}^{T}$, the sample autocorrelation at lag-l is then

$$\hat{\rho_l} = \frac{\sum_{t=l+1}^{T}(r_t - \bar{r})(r_{t-l} - \bar{r})}{\sqrt{\sum_{t=1}^{T}(r_t - \bar{r})^2}} \tag{2}$$

where $\bar{r}$ is the sample mean. And $\hat{\rho_1}$, $\hat{\rho_2}$,..., $\hat{\rho}_{T-1}$ is the sample autocorrelation function (ACF).

White Noise

We often assume that a time series is a sum of series with deterministic linear process. It is dependent on explanatory variables and a series of random white noise. White noise is independent and identically distributed with finite mean and variance. For example, a Gaussian white noise is normally distributed with zero mean and σ^2 variance. A white noise series is not autocorrelated. In practical modelling, sample returns are not autocorrelated if sample ACFs are statistically significant close to zero.

1.1.2 Autoregressive (AR) Model

An Autoregressive (AR) model is a model that value r_t is regressed on previous values. The AR(1) is $r_t = \beta_0 + \beta_1 y_{t-1} + \epsilon_t$, where ϵ_t is white noise. The AR(1) model can be generalized to the AR(p) model as following:

$$r_t = \beta_0 + \beta_1 r_{t-1} + \cdots + \beta_p r_{t-p} + \epsilon_t \tag{3}$$

where p is a positive integer and ϵ_t is white noise. In the AR(p) model, r_t is dependent jointly on previous values.

To choose the order p in the AR(p) model, we can use partial autocorrelation function (PACF). Let time series be r_t, the PACF of lag l is denoted as $\alpha(l)$, that is the autocorrelation between r_t and r_{t+l}, with r_{t+l-1} removed. The PACF of lag 1 is given then $\alpha(1) = Corr(r_{t+1}, r_t)$, for $l = 1$. PACF of lag l is then

$$\alpha(l) = Corr(r_{t+l} - P_{t,l}(r_{t+l}), r_t - P_{t,l}(r_t)) \tag{4}$$

for $k \geq 2$, where $P_{t,l}(x)$ is so called surjective operator of orthogonal projection onto the linear subspace of Hilbert space spanned by $r_{t+1}, ..., r_{t+l-1}$. (Box, Jenkins & Reinsel, 2008).

ACF and PACF plots are commonly used to identify the AR model and its lag. We look for the order when PACF with higher lags are close to zero. When checking ACF and PACF plots, an indication of sampling uncertainty is usually placed. Under the assumption that there are moderately big data points ($n > 30$) and with finite second moment, we check when PACF is zero at 5% significance level.

1.1.3 Moving Average (MA) Model

A moving average (MA) model focuses on the variables that are dependent on current and previous values of white noise. Given order q for the MA model, MA(q) is then

$$r_t = \mu + \epsilon_t + \theta_1\epsilon_{t-1} + \cdots + \theta_q\epsilon_{t-q} \tag{5}$$

where μ is the series mean, $\{\epsilon_t, ..., \epsilon_{t-q}\}$ are white noise and $\{\theta_1, ..., \theta_q\}$ are the parameters of the MA model. The white noise is also called random shocks or error terms. We assume that the error terms are mutually independent and have a distribution with zero mean and constant variance.

1.1.4 Autoregressive Moving Average (ARMA) Model

For the Autoregressive Moving Average (ARMA) model, we combine the AR and MA models. The ARMA(1,1) model is then constructed with the AR(1) and MA(1) model:

$$r_t = \mu + \beta r_{t-1} + \epsilon_t + \theta\epsilon_{t-1} \tag{6}$$

where $\beta, \theta \neq 0$, μ is a constant mean, ϵ_t is white noise with zero mean and σ_ϵ^2 variance. Furthermore, the ARMA(p,q) model is generalized in the form:

$$\begin{aligned} r_t &= \mu + \beta_1 r_{t-1} + \cdots + \beta_p r_{t-p} + \epsilon_t + \theta_1\epsilon_{t-1} + \cdots + \theta_q\epsilon_{t-q} && (7) \\ &= \mu + \sum_{i=1}^{p}\beta_i r_{t-i} + \epsilon_t + \sum_{i=1}^{q}\theta_i\epsilon_{t-i} && (8) \end{aligned}$$

1.2 Conditional Heteroscedastic Model

In this chapter, we discuss conditional heteroscedastic models - the ARCH and GARCH models. The ARCH model introduces the conditional variance that changes over time based on past error terms and leaving the unconditional variance constant. The GARCH model is an extension of the ARCH model. It adds the resemblance of the standard time series AR process to the general ARMA model. The addition of conditional variances helps to model the volatility without adding many lagged squared returns compared with the ARCH model. Both models have similar properties that capture volatility clustering but cannot address the leverage effect and asymmetric impact from positive and negative returns.

1.2.1 ARCH Model

In the ARCH model (Bollerslev, Engle & Nelson, 1994), if asset returns denoted as ε_t, we can write the ARCH(q) model as following:

$$\begin{aligned} \varepsilon_t &= \sigma_t e_t \\ \sigma_t^2 &= \alpha_0 + \alpha_1 \varepsilon_{t-1}^2 + ... + \alpha_q \varepsilon_{t-q}^2 \\ &= \alpha_0 + \sum_{i=1}^{q} \alpha_i \varepsilon_{t-i}^2 \end{aligned} \tag{9}$$

where ε_t is a stochastic process of white noise e_t and a time-dependent standard deviation σ_t. The parameter $\alpha_0 > 0$, and returns ε_t is not autocorrelated. The white noise e_t is the error term that is independent and identically distributed (i.i.d.) with zero mean and unit variance.

1.2.2 GARCH Model

The GARCH model introduces a more general class of process. It extends from the ARCH model and includes lagged conditional variances. Let y_t be the return, and ε_t be the innovation. The GARCH(p, q) process is as following:

$$\begin{aligned} y_t &= \mu + \varepsilon_t \\ \varepsilon_t &= \sigma_t e_t \\ \sigma_t^2 &= \omega + \alpha_1 \varepsilon_{t-1}^2 + ... + \alpha_q \varepsilon_{t-q}^2 + \beta_1 \sigma_{t-1}^2 + ... + \beta_p \sigma_{t-p}^2 \\ &= \omega + \sum_{i=1}^{q} \alpha_i \varepsilon_{t-i}^2 + \sum_{i=1}^{p} \beta_i \sigma_{t-i}^2 \end{aligned} \tag{10}$$

where p is the order for the conditional variance σ_t^2 and q is the order for innovation ε_t^2. White noise e_t is independent and identically distributed with zero mean and unit variance.

The parameters must satisfy that $\omega > 0$, $\alpha \geq 0$, $\beta \geq 0$ for conditional variance to be positive. Intercept ω needs to be strictly positive for the process y_t not degenerating. y_t is stationary if and only if $\alpha + \beta < 1$. In empirical modelling, mean of financial return series μ can be seen as 0.

1.3 Support Vector Machine

Support Vector Machine is applied to solving varied classification and regression problems. The algorithm is widely used in categorization, image recognition, biological classification, etc. We characterize SVM with a number of support vectors, the cost functions, usage of kernel tricks, mapping into high dimensional feature space etc. (Smola & Schölkopf, 2004).

In this section, we demonstrate the basic idea and formulations for Support Vector Machine and focus on the Support Vector Regression. Then we proceed with topics of dual formulation, kernels and grid search that are used in the SVR.

1.3.1 SVM Regression

In SVM Regression (Vapnik 1998), the basic idea is to find a function $f(x)$ that forms the targets with training data that has most ε deviation and as flat as possible. The errors that are less than ε are ignored, and ones larger than ε are not accepted (Smola, Murata, Schölkopf & Müller, 1998).

Given training data $\{(x_1, y_1), ..., (x_l, y_l)\} \subset \mathbf{X} \times \mathbb{R}$, where $\mathbf{X}$ denotes the space of the input patterns. In the case of linear functions f, it is defined as (Chang & Lin, 2001):

$$f(x) = \langle \omega, x \rangle + b \text{ with } \omega \in \mathbf{X}, b \in \mathbb{R} \tag{11}$$

where $\langle \cdot, \cdot \rangle$ denotes the dot product in $\mathbf{X}$. It is then a constrained optimization problem for seeking small ω because of the flatness in equation 11.

Then the optimization problem is given as:

$$\begin{aligned} &\text{minimize } \frac{1}{2} \parallel \omega \parallel^2 \\ &\text{subject to } \begin{cases} y_i - \langle \omega, x_i \rangle - b \leq \varepsilon, \\ \langle \omega, x_i \rangle + b - y_i \leq \varepsilon \end{cases} \end{aligned} \tag{12}$$

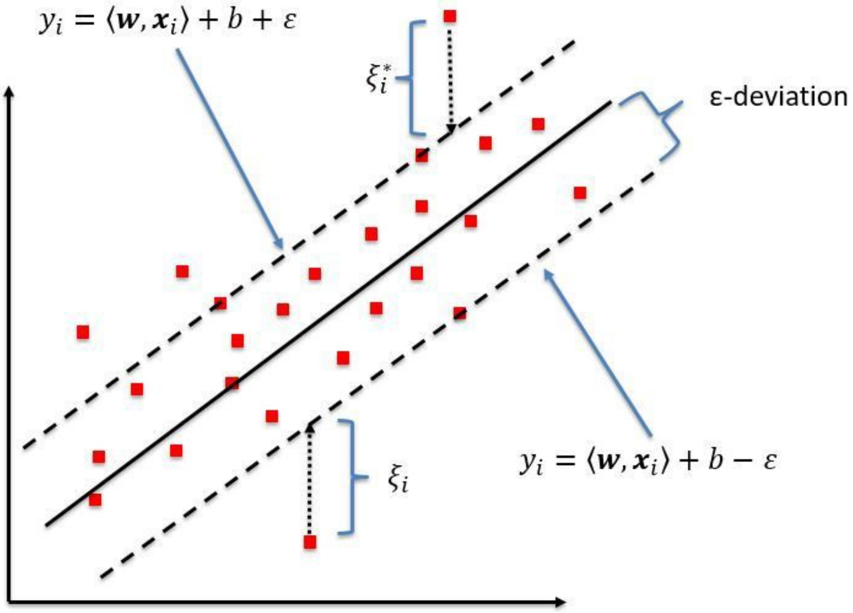
It assumes that the convex optimization problem is feasible that such a function f exists and pairs approximates all (x_i, y_i) with ε precision. When assumption is not met and if we want to allow for some errors, a soft margin loss function in (Cortes & Vapnik, 1995) was introduced. As stated in (Vapnik, 1995):

$$\text{minimize } \frac{1}{2}w^T w + C\sum_{i=1}^{n}(\zeta_i + \zeta_i^*)$$
$$\text{subject to } \begin{cases} y_i - \langle\omega, x_i\rangle - b \leq \varepsilon + \zeta_i, \\ \langle\omega, x_i\rangle + b - y_i \leq \varepsilon + \zeta_i^*, \\ \zeta_i, \zeta_i^* \geq 0, i = 1, ..., n \end{cases} \tag{13}$$

The constant $C > 0$ is the determination of the trade off between how flat f should be and up to which amount deviations larger than ε can be tolerated.

The figure1 illustrates the optimization problem with ε-insensitive loss function.

Figure 1: One-dimensional SVR model


$y_i = \langle \mathbf{w}, \mathbf{x}_i \rangle + b + \varepsilon$
ξ_i^*
ε-deviation
ξ_i
$y_i = \langle \mathbf{w}, \mathbf{x}_i \rangle + b - \varepsilon$

Source:Kleynhans, Montanaro, Gerace & Kanan (2017).

The goal is to look for a decision boundary with a distance of ε. The points outside contribute to the cost. The boundary that contains points is the margin of tolerance, which is then the decision boundary.

1.3.2 Dual Formulation

A standard dualization method using Lagrange multipliers provides the key for extending support vector machine to nonlinear functions (Fletcher & Sainz de la Maza, 1989). From the primal objective function and the constraints, a Lagrange function is constructed by introducing a dual set of variables, leading to the minimization of:

$$\begin{aligned} L := \frac{1}{2} \parallel \omega \parallel^2 + C \sum_{i=1}^{l} (\zeta_i + \zeta_i^*) - \sum_{i=1}^{l} \alpha_i (\varepsilon + \zeta_i - y_i + \langle \omega, x_i \rangle + b) \\ - \sum_{i=1}^{l} \alpha_i^* (\varepsilon + \zeta_i^* + y_i - \langle \omega, x_i \rangle - b) - \sum_{i=1}^{l} (\eta_i \zeta_i + \eta_i^* \zeta_i^*) \end{aligned} \tag{14}$$

Dual variables have to satisfy positivity constraints that $\alpha_i, \alpha_i^*, \eta_i, \eta_i^* \geq 0$. Partial derivatives of L with respect to the primal variables are then set to 0:

$$\partial_b L = \sum_{i=1}^{l} (\alpha_i^* - \alpha_i) = 0 \tag{15}$$

$$\partial_\omega L = \omega - \sum_{i=1}^{l} (\alpha_i - \alpha_i^*) x_i = 0 \tag{16}$$

$$\partial_{\zeta_i(*)} L = C - \alpha_i^{(*)} - \eta_i^{(*)} = 0 \tag{17}$$

The dual optimization problem is obtained by substituting equations 15, 16 and 17 into equation 14, leading to the maximization of:

$$\begin{aligned} \text{maximize} \quad -\frac{1}{2} \sum_{i,j=1}^{l} (\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*) \langle x_i, x_j \rangle - \varepsilon \sum_{i=1}^{l} (\alpha_i + \alpha_i^*) + \sum_{i=1}^{l} y_i (\alpha_i - \alpha_i^*) \\ \text{subject to} \begin{cases} \sum_{i=1}^{l} (\alpha_i - \alpha_i^*) = 0, \\ \alpha_i, \alpha_i^* \in [0, C] \end{cases} \end{aligned} \tag{18}$$

The dual variables ζ_i, ζ_i^* are eliminated through conditions in equation 17. They are not in the dual objection function but are present in the dual feasibility conditions. To rewrite equation 16, we get:

$$\begin{aligned} \omega = \sum_{i-1}^{l} (\alpha_i - \alpha_i^*) x_i \\ f(x) = \sum_{i=1} (\alpha_i - \alpha_i^*) \langle x_i, x \rangle + b \end{aligned} \tag{19}$$

The function ω can be described as a linear combination of the training set x_i. The algorithm is dependent of dot products between the training data. In a sense, the function is independent of the dimensionality of the input space $\mathbf{X}$ but dependent on the number of support vectors that are training instances within the distance of margin. When in linear setting, it is more efficient to compute ω explicitly. In this thesis, we use Python scikit learn to conduct SVR. In the case of linear kernel, we can obtain weights assigned to the features.

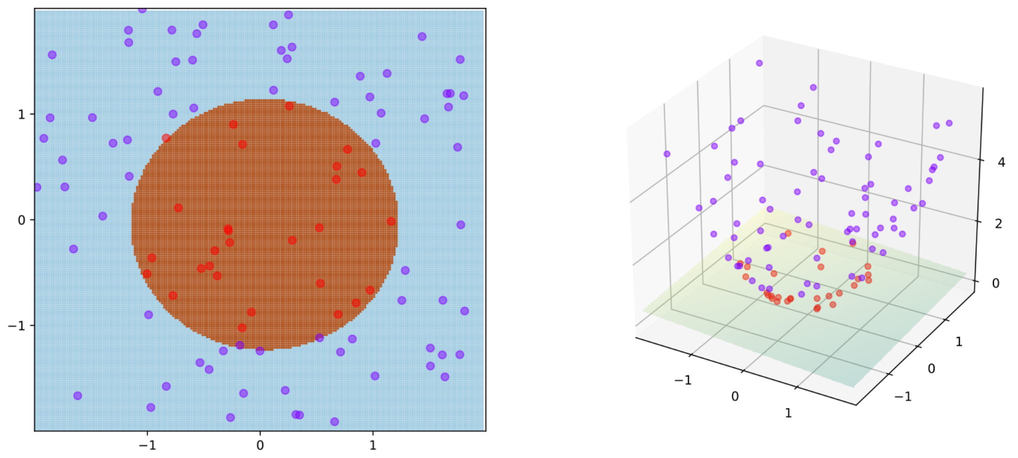
1.3.3 Kernels

Kernels can be understood as an instance-based learning method, also called memory-based learning. It compares the instances in training that are stored in the memory to the ones in the new problem, instead of performing explicit generalization. For example, it doesn't learn a fixed set of parameters and features but it learns the corresponding weights of training samples (Hofmann, Schölkopf & Smola, 2008).

To get a decision boundary of the nonlinear function, we use the implicit mapping via kernels. The kernel functions are expressed as an inner product in another space. The Mercer theorem (Schölkopf, Platt, Shawe-Taylor & Smola, 2001) states the conditions for a function $K(x_i, x_j)$ to be sufficient kernel. Through the implicit mapping of the training points to a higher dimensional space, a hyperplane can be easier found while the training points aren't linearly separable in the actual space.

For example, given by $\varphi((a,b)) = (a, b, a^2 + b^2)$, an SVM kernel is $K(\mathbf{x}, \mathbf{y}) = \mathbf{x} \cdot \mathbf{y} + \| \mathbf{x} \|^2 \| \mathbf{y} \|^2$. Through the kernel, the training points can be mapped to a 3-dimensional space and the separating hyperplane can be found, as shown in Figure 2.

Figure 2: Kernel trick $K(\boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{x} \cdot \boldsymbol{y} + \| \boldsymbol{x} \|^2 \| \boldsymbol{y} \|^2$



Source: Ji (2017).

Some of the most widely used kernels are in table 2, where k is a natural number and σ is a real positive number. In this study, these three kernels are used in the grid search for getting the best parameter values for an estimator in SVR functions.

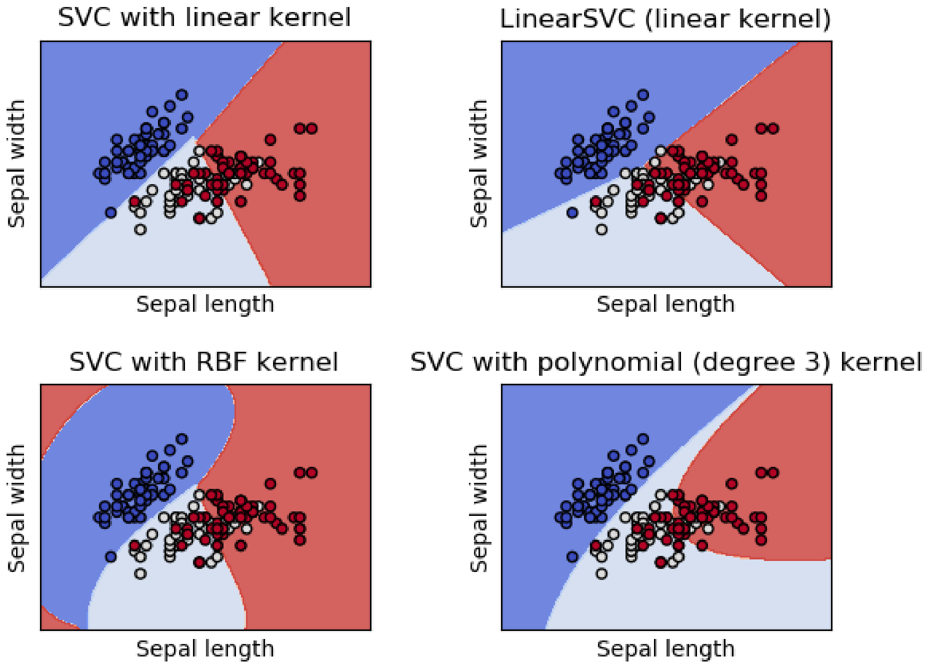
Table 2: Kernels used in SVR

Linear	$K(x_i, x_j) = x_i^T x_j$
Polynomial	$K(x_i, x_j) = (x_i^T x_j + 1)^k$
Rbf	$exp(- \parallel x_i - x_j \parallel^2 / (2\sigma^2))$

Source: Gavrishchaka, Ganguli (2003).

The kernel trick makes it possible that we can apply the SVM for non-linear data sample by mapping the training data to a higher dimensional space. We apply three kernels as mentioned in table 2 using scikit-learn Python library. The linear models have linear decision boundaries. And for the polynomial and Rbf kernels, there are decision boundaries with shapes and more flexibility. Examples of kernel models and decision boundaries are demonstrated in figure 3.

Figure 3: An example of SVM classfiers with different kernels



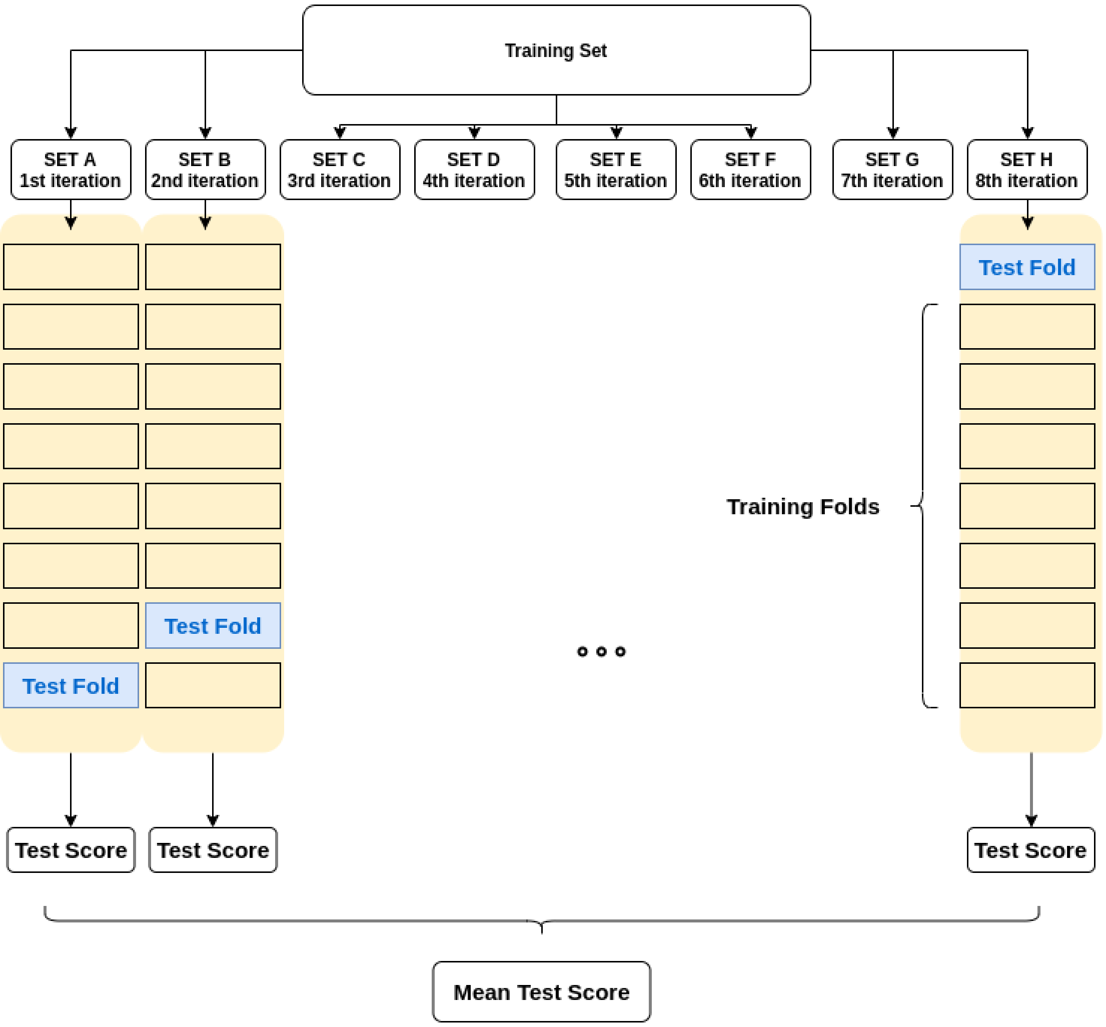
Source: Pedregosa et al. (2011).

1.3.4 Grid Search

A grid search consists of an estimator, a parameter space, a method for searching or sampling candidates, a cross-validation scheme and a score function. The hyper-parameters are parameters not directly learnt within estimators, but passed as arguments to the constructor of the estimator classes.

The grid search exhaustively generates candidates from a grid of parameter values. After fitting on a dataset, all possible combinations of parameter values are evaluated using a specified score method. We use the eightfold cross validation. We divide in-sample data into 8 sets, using 7 sets for training and the last for validation. The process is repeated 8 times and we get the best combination for hyper-parameters with the lowest mean test score for lowest error. We show a demonstration of cross validation in figure 4.

Figure 4: A Demonstration of Cross Validation



Source: Own work.

In this study, we use Mean Squared Logarithmic Error (MSLE) as the scoring method in grid search. It computes a risk metric corresponding to the expected value of the squared logarithmic error or loss. It can be interpreted as a measure of the ratio between the true and predicted values. It is the best to use when targets having exponential growth. It has a characteristic that it penalizes an under-predicted estimate greater than an over-predicted estimate.

Given $\hat{y}_i$ as the forecasted value and y_i as the corresponding true value, the MSLE in n samples is shown in equation 20, where $log_e(x)$ is the natural logarithm of x.

$$MSLE(y, \hat{y}) = \frac{1}{n}\sum_{i=0}^{n-1}(log_e(1+y_i) - log_e(1+\hat{y}_i))^2 \tag{20}$$

1.4 GARCH(1,1) Parameter Estimation

We use the GARCH(1,1) model in this thesis for the comparison of parameter estimation. The GARCH(1,1) has its simplicity and good representation of characteristics of financial assets. The GARCH(1,1) model is as following:

$$y_t = \mu + \varepsilon_t \tag{21}$$

$$\varepsilon_t = \sigma_t e_t \tag{22}$$

$$\sigma_t^2 = \omega + \alpha\varepsilon_{t-1}^2 + \beta\sigma_{t-1}^2 \tag{23}$$

In equation 23, the parameters ω, α and β are usually estimated with MLE methods, which requires an assumption for the distribution of error term e_t in equation 22. In this section, we start with the general MLE method including two distribution assumptions - normal and student's t distribution. Then we demonstrate the MLE estimation process for both assumptions. After MLE method, we proceed with SVR method and explain how we estimate parameters ω, α and β in case of linear kernel. In the last part of the section, we show the evaluation of forecasted volatility using coefficient determination (R^2).

1.4.1 Distributions of $\boldsymbol{e_t}$

Recall that the error term e_t is identical and independently distributed with zero mean and unit variance. We have two assumptions for the error term e_t: normal distribution $e_t \sim N(0,1)$ and student's t distribution $\sqrt{\frac{\nu}{\nu-2}}e_t \sim t_\nu$.

The density function of normal distribution is

$$f(z) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(z-\mu)^2}{2\sigma^2}}, -\infty < z < \infty \tag{24}$$

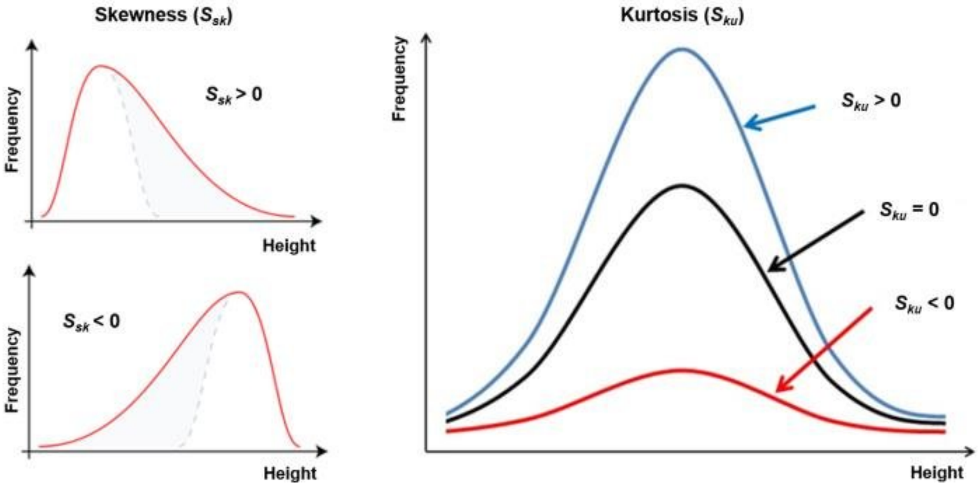
The density function of students't distribution is

$$f(z) = \frac{\Gamma(\frac{\nu+1}{2})}{\sqrt{\nu\pi}\Gamma(\frac{\nu}{2})}(1 + \frac{z^2}{\nu})^{-(\frac{\nu+1}{2})}, -\infty < z < \infty \tag{25}$$

In student's t density function, ν denotes the number of degrees of freedom, Γ denotes the function $\Gamma(x) = \int_0^\infty y^{x-1} e^{-y} dy$.

To check possible distribution and normality of e_t, we use skewness test (D'Agostino, Belanger & D'Agostino J.R.B., 1990) and kurtosis test (Anscombe & Glynn, 1983). Skewsness s is the z-score returned by skew test and kurtosis k is the z-score returned by kurtosis test. The normality test refers to the statistics $s^2 + k^2$. A brief illustration of skewness and kurtosis is shown in figure 5. A positive skewness means that the right tail is longer and the mean is skewed to the right of a typical data center. Similarly, when the skewness is negative, the left tail is longer. We can interpret that the mass of the distribution is concentrated to the right with a longer left tail. Regarding the kurtosis, it also demonstrates the shape of distributions. A positive kurtosis is called leptokurtic with fatter tails. Comparing to a normal distribution or zero excess kurtosis, the positive kurtosis means that there are more extreme outliers. And a negative kurtosis is called platykurtic. It means thinner tails and the outliers are less extreme.

Figure 5: Skewness and Kurtosis



Source: Bonyar (2015).

1.4.2 Maximum Likelihood Estimation

With an assumption of normal distribution, recalling the error term $e_t \sim N(0,1)$, ε_t is then given by $\varepsilon_t = y_t - \mu \sim N(0, \sigma_t^2)$. When performing MLE, the joint distribution $f(\varepsilon_1, ..., \varepsilon_T; \theta)$ is interested with θ as the parameter vector. The joint distribution is equal to the product of the conditional and the marginal density. We then have the joint distribution as following:

$$\begin{aligned} f(\varepsilon_0, ..., \varepsilon_T; \theta) &= f(\varepsilon_0; \theta) f(\varepsilon_1, ..., \varepsilon_T | \varepsilon_1; \theta) \\ &= f(\varepsilon_0; \theta) \prod_{t=1}^{T} f(\varepsilon_t | \varepsilon_{t-1}, ..., \varepsilon_0, ; \theta) \\ &= f(\varepsilon_0; \theta) \prod_{t=1}^{T} f(\varepsilon_t | \varepsilon_{t-1}, ; \theta) \\ &= f(\varepsilon_0; \theta) \prod_{t=1}^{T} \frac{1}{\sqrt{2\pi\sigma_t^2}} \exp\left(-\frac{\varepsilon_t^2}{2\sigma_t^2}\right) \end{aligned} \tag{26}$$

Now by taking logs, log-likelihood function is obtained as

$$L(\theta) = \sum_{t=1}^{T} \frac{1}{2} \left[-\log 2\pi - \log(\sigma_t^2) - \frac{\varepsilon_t^2}{\sigma_t^2} \right] \tag{27}$$

When the assumption that error distribution is student's t with ν degress of freedom, the log-likelihood function is:

$$L(\theta) = nlog \left[\frac{\Gamma(\frac{\nu+1}{2})}{\sqrt{\pi(\nu-2)}\Gamma(\frac{\nu}{2})} \right] - \frac{1}{2} \sum_{t=1}^{T} log(\sigma_t^2) - \frac{\nu+1}{2} \sum_{t=1}^{T} log \left[1 + \frac{\varepsilon_t^2}{\sigma_t^2(\nu-2)} \right] \tag{28}$$

The Gaussian log-likelihood function is maximized, through an iterative algorithm. The estimates are called maximum likelihood when normal distribution is the underlyting pdf from the sample data. In other case, it is called quasi-maximum likelihood. In this thesis, we carry out the process of maximization with Python ARCH package (Sheppard, 2019).

1.4.3 New Support Vector Regression (**ν**-SVR)

The New Support Vector Regression introduces a new parameter $\nu \in (0, 1]$. It adds an upper bound on a fraction of training errors and a lower bound of fraction of support vectors, which controls the number of support vectors and training errors. With (C, ν) as parameters, ν-SVR solves

$$\text{minimize } \frac{1}{2} \parallel \omega \parallel^2 + C(\nu\varepsilon + \frac{1}{l}\sum_{i=1}^{l})(\zeta_i + \zeta_i^*)$$
$$\text{subject to } \begin{cases} y_i - \langle \omega, x_i \rangle - b \leq \varepsilon + \zeta_i^*, \\ \langle \omega, x_i \rangle + b - y_i \leq \varepsilon + \zeta_i, \\ \zeta_i, \zeta_i^* \geq 0, i = 1, ..., l, \varepsilon \geq 0 \end{cases} \tag{29}$$

A kernel k for the dot product is used via a nonlinear map Φ: $k(\mathbf{x}, \mathbf{y}) = (\Phi(\mathbf{x}) \cdot \Phi(\mathbf{y}))$. The ν-SVR optimization problem is:

$$\text{maximize } \sum_{i=1}^{l}(\alpha_i - \alpha_i^*)y_i - \frac{1}{2}\sum_{i,j=1}^{l}(\alpha_i^* - \alpha_i)(\alpha_j^* - \alpha_j)k(x_i, x_j)$$
$$\text{subject to } \begin{cases} \sum_{i=1}^{l}(\alpha_i - \alpha_i^*) = 0, \\ \alpha_i^{(*)} \in [0, \frac{C}{l}], \\ \sum_{i=1}^{l}(\alpha_i + \alpha_i^*) \leq C \cdot \nu \end{cases} \tag{30}$$

for $\nu \geq 0, C > 0$. The approximate function gives as following:

$$f(x) = \sum_{i=1}^{l}(\alpha_i^* - \alpha_i)k(\mathbf{x}_i, \mathbf{x}) + b \tag{31}$$

Three kernels - linear, polynomial and rbf are used in the grid search. The linear kernel is found as the best hyper-parameter in empirical modeling. In linear SVM, the separating plane is the same as input features. In the case of applying linear kernel where $k(\mathbf{x}_i, \mathbf{x}) = \mathbf{x}_i^T \cdot \mathbf{x}$, the approximate function is:

$$f(x) = \sum_{i=1}^{l}(\alpha_i^* - \alpha_i)\mathbf{x}_i \cdot \mathbf{x} + b \tag{32}$$

Recalling equation 11 and rewriting as $f(x) = \mathbf{w} \cdot \mathbf{x} + b$, weights $\mathbf{w}$ or coefficients for the regressor are:

$$\mathbf{w} = \sum_{i=1}^{l}(\alpha_i^* - \alpha_i)\mathbf{x}_i \tag{33}$$

In SVR, the machine needs observable variables in the training set to train the model. In the GARCH(1,1) model recalling that $\sigma_t^2 = \omega + \alpha\varepsilon_{t-1}^2 + \beta\sigma_{t-1}^2$, σ_t^2 is the dependent variable, while ε_{t-1}^2 and σ_{t-1}^2 serve as the regressor. Through solving the optimization problem of SVR, the dual parameters and support vectors will return the weights of input features in case of the linear kernel.

We use Python library LIBSVM and scikit learn in this study. Coefficients can be obtained directly or by multiplying dual parameters with support vectors. In correspondence with parameters in GARCH(1,1) model: $\sigma_t^2 = \omega + \alpha\varepsilon_{t-1}^2 + \beta\sigma_{t-1}^2$, the feature vector **x** is then $[\varepsilon_t - 1^2, \sigma_{t-1}^2]$ and dependent variable is σ_t^2. The weights **w** return estimated parameters $\hat{\alpha}$ and $\hat{\beta}$ and the intercept returns $\hat{\omega}$.

1.5 Volatility Forecasting and Evaluation

Analytical forecasts are applied for variance forecasting. For the one-step ahead conditional variances, we construct the forecasts as follows:

$$\hat{\sigma}_{t+1}^2 = \hat{\omega} + \hat{\alpha}y_t^2 + \hat{\beta}\sigma_t^2 \tag{34}$$

We use squared returns as proxy of daily volatility. Given the GARCH(1,1) model and $\mu = 0$, we have $y_t = \sigma_t e_t$. Under the condition that error term has unit variance, we get $Var_t[e_{t+1}^2] = 1 = E_t[e_{t+1}^2] - \{E_t[e_{t+1}]\}^2 = E_t[e_{t+1}^2] = 1$. As demonstrated in equation 35, we have now $E_t[y_{t+1}^2] = \sigma_{t+1}^2$. It shows that the squared returns are unbiased and efficient for conditional volatility forecasting as proxy. However, it is imprecise and often performs poorly as realized variance (Triacca, 2007).

$$\begin{aligned} E_t[y_{t+1}^2] &= E_t[\sigma_{t+1}^2 e_{t+1}^2] \\ &= \sigma_{t+1}^2 E_t[e_{t+1}^2] \\ &= \sigma_{t+1}^2 \end{aligned} \tag{35}$$

We use coefficient determination (R^2) (Wright, 1921) for evaluation of volatility forecasting. It explains the proportion of the sample variation with the corresponding forecasts and indicates the goodness of fit. We compare between the forecasted variance from equation 34 and true values using daily squared returns.

Given $\hat{y}$ as forecasted value and $\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$, R^2 is defined as following:

$$R^2(y, \hat{y}) = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{36}$$

Best score of R^2 is 1.0. When R^2 is zero, it means that the model accounts for 0% of the variability. R^2 can also be negative. The model then introduces more variability than the sample mean. When R^2 = 0.15, it means that the model explains 15% of variability in the sample data. For all the evaluation results, R^2 is increased by a factor of 100.

2 EMPIRICAL MODELLING

In the second part, we cover the simulation process and the empirical modelling. We first start with the description of empirical data: SP500 and BTC/USD daily returns. We then show a simulation process to examine the effect of parameter estimation and the forecastibility with different methods: GARCH-MLE and GARCH-SVR. In the next chapter, we demonstrate the empirical modelling with each method. The GARCH-MLE method first shows results of estimation summary, calculated standardized residuals as well as TSA and QQ plots. For the GARCH-SVR methods, we present one part of the grid search table that contains the best hyper-parameters in the estimation process. At the end of the section, we show the evaluation as well as the plots of forecasted variance.

2.1 Empirical Data

We use SP500 and BTC/USD daily prices for empirical modelling. Data is retrieved from Yahoo finance using Python. We use daily prices from January 2014 to September 2019 for both financial assets. SP500 daily prices contain 1434 data points. Bitcoin daily prices have 2081 data samples. Daily prices are transformed into log return series. It is divided into half, with the first half as training set and the second as testing set. And we use time series analysis (TSA) plots to check if there is autocorrelation. Also we can observe the distributions of (squared) returns.

2.1.1 Data Transformation

Let p_t be the prices at time t and log returns y_t are calculated:

$$y_t = \ln p_t - \ln p_{t-1} \tag{37}$$

Assuming the mean of return as zero ($\mu = 0$), we have now the GARCH(1,1) model:

$$y_t = \sigma_t e_t \tag{38}$$

$$\sigma_t^2 = \omega + \alpha y_{t-1}^2 + \beta \sigma_{t-1}^2 \tag{39}$$

We use daily squared returns as realized variance. From the GARCH(1,1) model in 38 and 39, we estimate the parameters $\hat{\omega}$, $\hat{\alpha}$ and $\hat{\beta}$. And the forecasted variance becomes $\hat{\sigma}_{t+1}^2 = \hat{\omega} + \hat{\alpha} y_t^2 + \hat{\beta} \sigma_t^2$. The evaluation R^2 is calculated from y_{t+1}^2 and forecasted variance $\hat{\sigma}_{t+1}^2$. In order to compare among different data sets and sample sizes, we use standardized residual $\hat{e}_t$ for checking their distribution and normality. The standardized residual is calculated from $\hat{e}_t = y_t / \hat{\sigma}_t$.

In GARCH-SVR there need to be observable variables for SVR to estimate parameters in the GARCH model. By trying to smooth and eliminate noise from the data series, we use a moving average of the contemporaneous and four lagged squared returns at each point. The proxy $\tilde{\sigma}$ is given by

$$\tilde{\sigma_t}^2 = \frac{1}{5} \sum_{k=0}^{4} y_{t-k}^2 \tag{40}$$

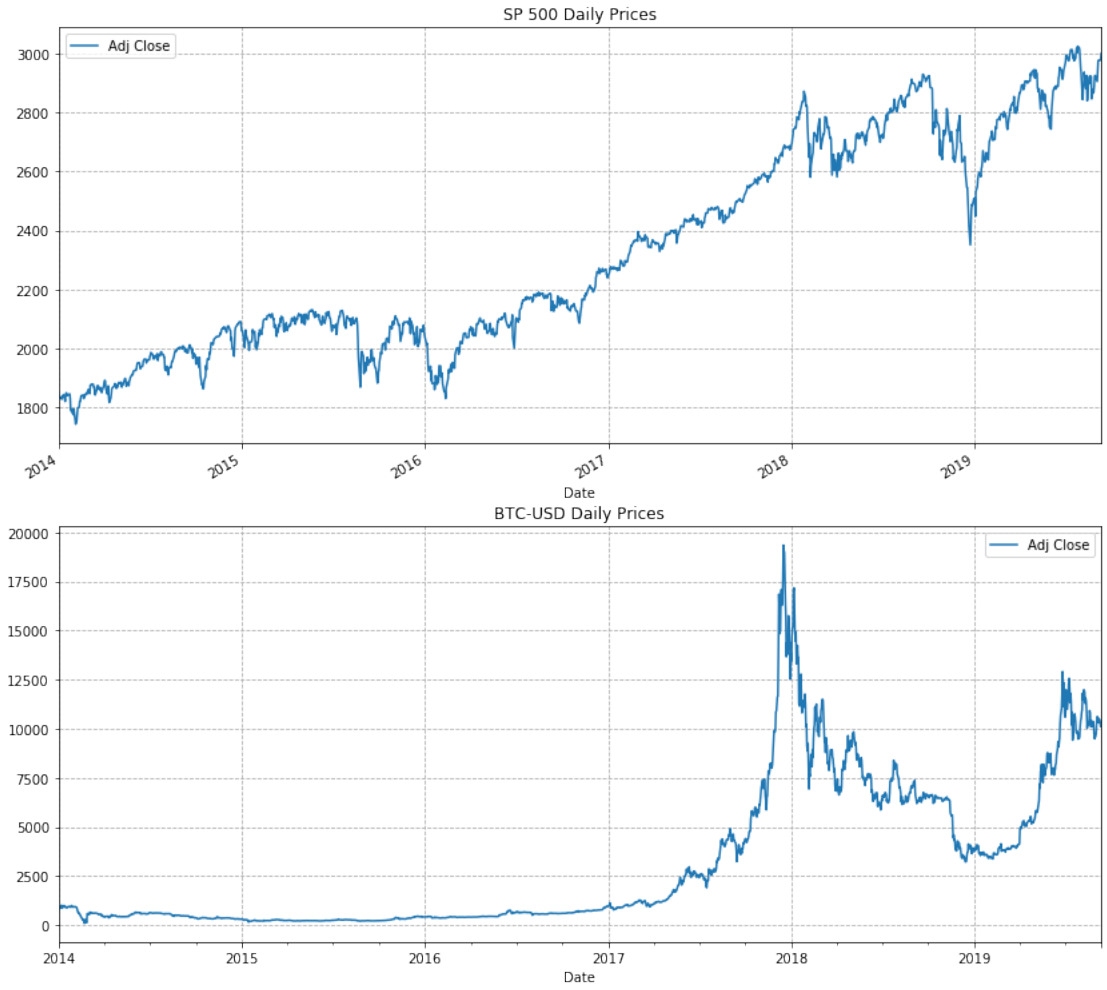
2.1.2 Characteristics of SP500 and Bitcoin Prices

There are a number different characteristics between SP500 and Bitcoin markets. The correlation between the two assets is generally very weak. Unless there is a global fear and a sudden crash, as for example in March 2020, the correlation between the two assets reached all-time highs because of the pandandemic and liquidity crisis. However it fell back to low correlation at the end of the month.

Bitcoin as a cryptocurrency, is considered as an investment rather than a currency. One uniqueness of Bitcoin is that the total amount of the assets is fixed. There are studies showing its potential comparing to gold that it can replace gold as a 'safe-heaven' investment (Meech & Gu, 2014). It is found that Bitcoin prices react more to public interests, e.g. Google views increasing its transaction volume (Bouoiyour & Selmi, 2014). Since Bitcoin is in its early stages, it is also reacting strongly to extreme events and price movements comparing to those in mature markets. It is very volatile with extreme price peaks and drops, which happens more in less mature/less liquid markets (Bouchaud & Potters, 1999). In same period of time, BTC prices contain more data samples because the financial asset is traded 24/7 on the market.

The empirical data we use in this thesis - SP500 and BTC/USD prices from 2014 are shown in figure 6. It starts from 2014 and lasts till the end of 2019. Since the establishment of

Figure 6: Data prices

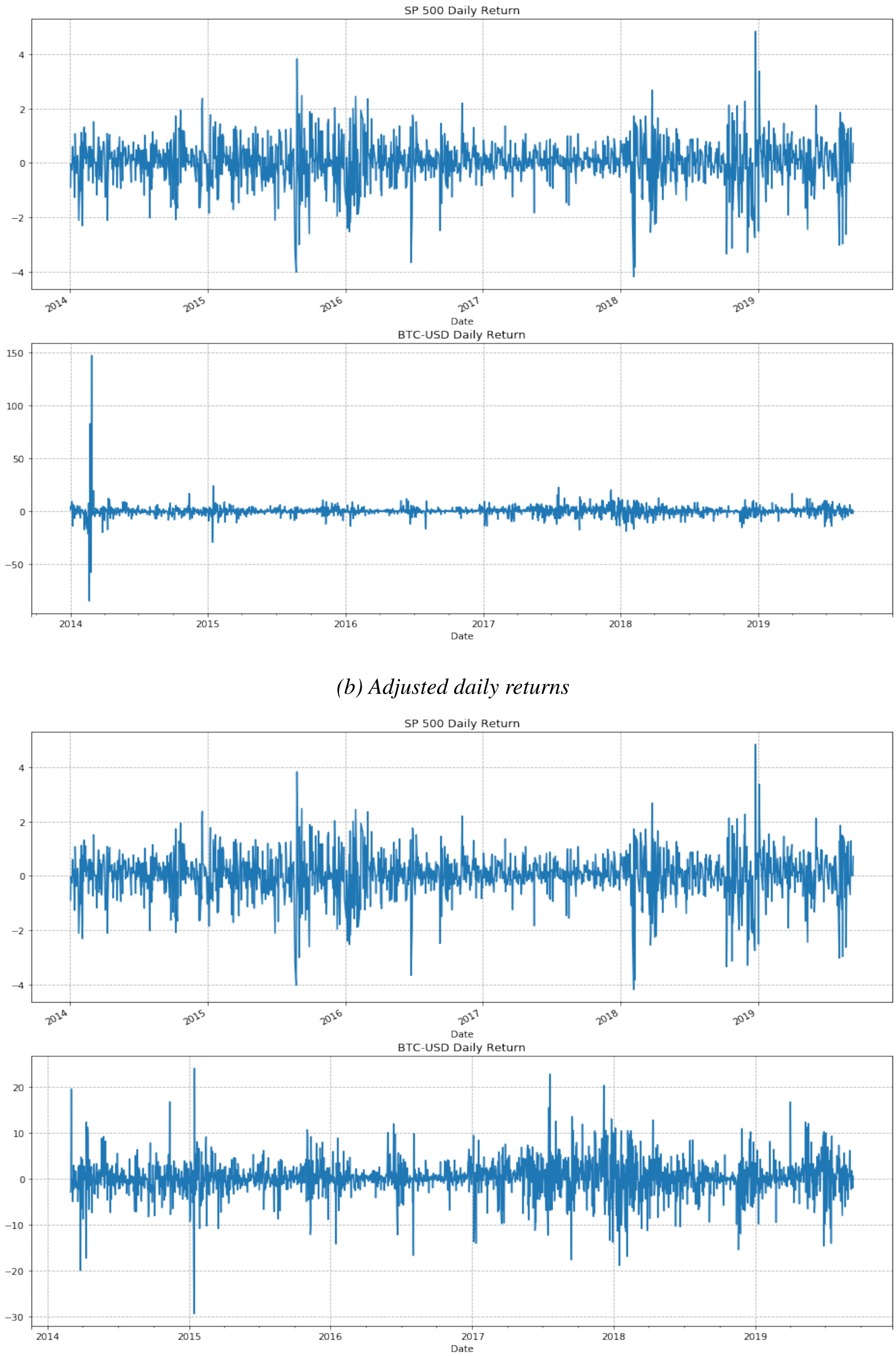


Source: Own work.

the first exchange in 2010, there have been numbers of extreme price spikes and drops. In July 2010, the prices skyrocketed by 900% in five days. From the charts below, we can see the significant spike from year 2017 onwards. The price reached new highs throughout the year and came to 19,783 USD on December 17th. By the end of 2018, the price fell by 76% and came below 3,300 USD. However, Bitcoin daily returns were most volatile in February 2014 when major exchanges got hit with distributed denial-of-service attacks (DDoS). On February 6th, Mt. Gox halted withdrawals which contributed to a sharp drop in Bitcoin prices from 940.42 USD/BTC on 01-02-2014 to 111.92 USD/BTC on 20-02-2014. When modelling the conditional variance, the event has a significant influence on the estimation process. Also as mentioned in the description of GARCH model, it doesn't capture the asymmetric news impact (Zivot, 2009) - volatility reacts to a stronger negative effect than an equal size of a positive shock. In this case, daily prices for Bitcoin are adjusted from March 2014 onwards for modelling. We show the calculated daily returns and the adjusted return series in figure 7.

Figure 7: Daily Returns 2014-2019

(a) Daily returns January 2014 to September 2019



(b) Adjusted daily returns

Source: Own work.

From the charts of adjusted daily returns, we can observe the clustering effects for both SP500 and BTC returns. For example, from the beginning of 2018, the BTC daily returns clusters on the negative side and last a period of time and repeated at the end of 2018. This is the similar case for SP500 returns in the same periods. Besides, BTC daily returns show more extreme spikes. In our sample for BTC return, the biggest daily drop is -29.4% on January 14th while the next day has the biggest positive return of 24%. For SP500 daily returns, it has maximum of positive return of 4.8% and negative return of -4.2% happening in 2018.

Now we have a further investigation into the empirical data. First we check the mean, standard deviation, skewness and kurtosis for the return y_t. Descriptive statistics are in table 3. From the observations, both return series have zero mean. It is much more volatile for BTC returns according to the standard deviation. They both have negative skewness - the left tail is longer and positive kurtosis - the outliers are more extreme. BTC also has relative high returns and with extreme occasions.

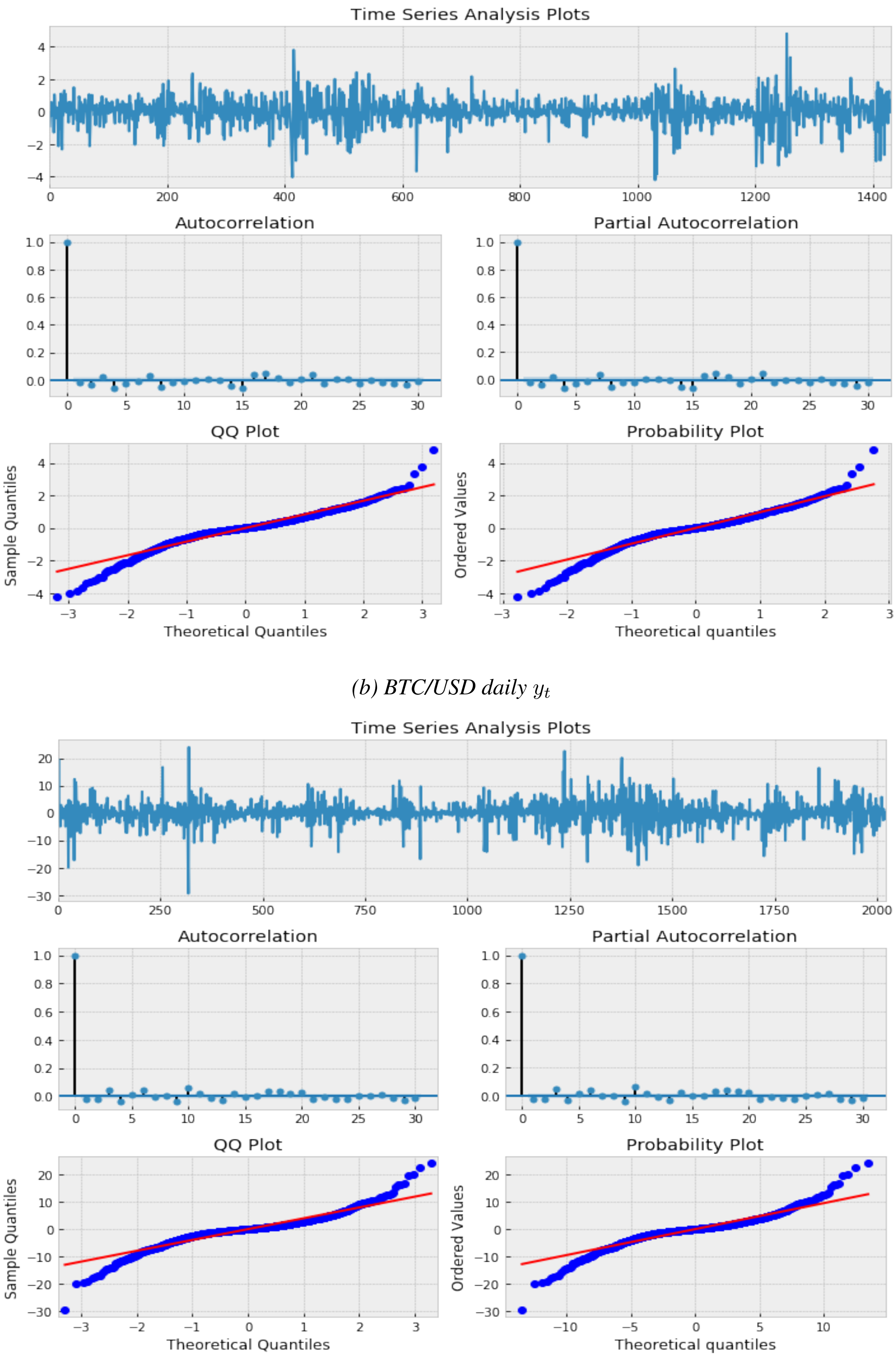
Table 3: Descriptive Statistics for returns y_t

y_t	**SP500 Daily**	**BTC Daily**
N	1434	2021
Mean	-6.35E-18	4.23E-17
Standard deviation	0.839	3.949
Skewness	-7.46	-5.27
Kurtosis	11.23	16.41

Source: Own work.

We also check time series analysis (TSA) plots for returns y_t and squared return y_t^2 to observe if any clustering effects and lags of autocorrelation. We add QQ and probability plots that are helpful to examine the distribution of the residuals. QQ plot is a scatterplot of two sets of quantiles against one another. The theoretical quantiles is normal distribution. If the sample quantile is normally distributed, scattered data then formes a straight line (as the red line shown in the figure). Probability plot serves in a similar way that it plots probability against probability. Details of TSA plots are in figure 8 and 9. Based on TSA plots, we can see that the return series is around zero mean. We also observe the clustering effect in both data series. It doesn't show autocorrelation for y_t but shows strong autocorrelation for y_t^2 in both time series. There is clustering effect for both return series. SP500 and BTC returns have heavy tails on both sides. SP500 has a heavier tail on its negative side than on its positive side. When comparing between two return series, BTC has a heavier tail on positive side.

Figure 8: Time Serie Analysis: Daily y_t

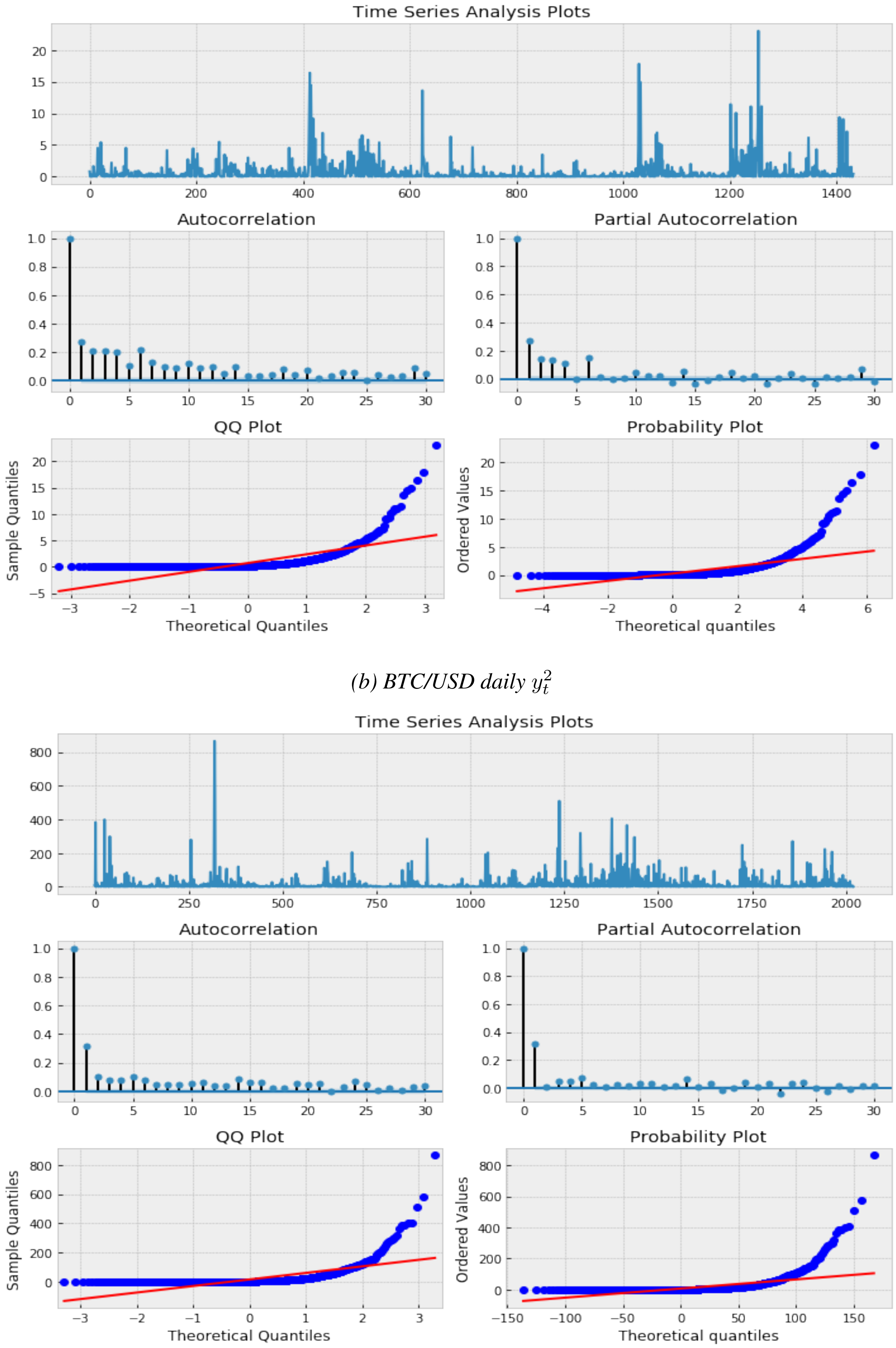


(a) SP500 daily y_t

(b) BTC/USD daily y_t

Source: Own work.

Figure 9: Time Serie Analysis: Daily y_t^2



(a) SP500 daily y_t^2

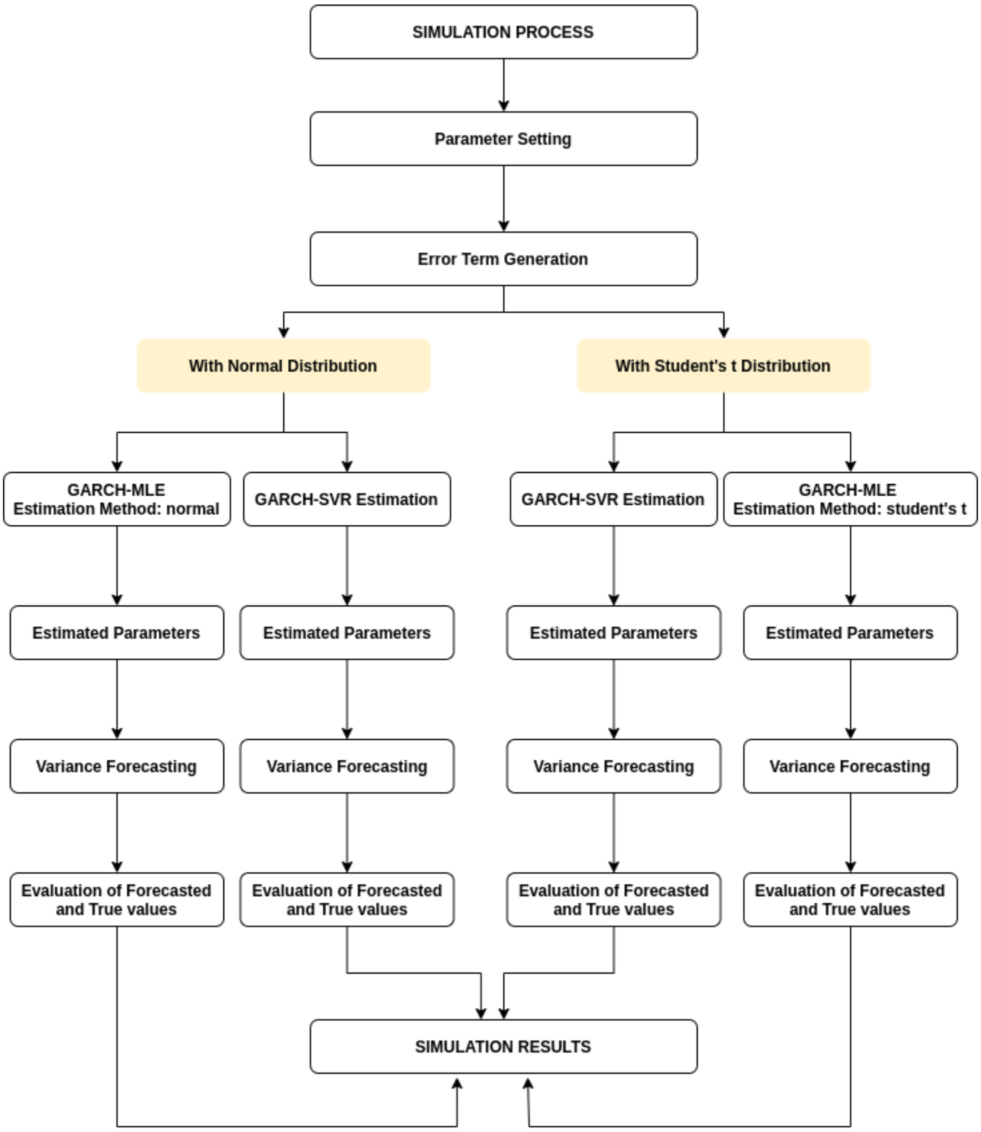
(b) BTC/USD daily y_t^2

Source: Own work.

2.2 Simulation Process

Before empirical modelling, we show a simulation process with 2000 samples for different methods. First we demonstrate an example of simulation process - using GARCH-MLE and GARCH-SVR with generated series and compare the results. The simulation process is demonstrated in figure 10. In addition, we simulate data with two parameter settings and we repeat the process for 100 times so that we obtain the mean of R^2 - the evaluation we use for variance forecasting. The simulation results help us understand the predictibility and the evaluation process.

Figure 10: Simulation Process Demonstration



Source: Own work.

2.2.1 An Example of Simulation Process

Recalling the GARCH(1,1) model with zero mean in equation 38 and 39, we set parameters ω, α and β and we generate data series y_t with 2000 samples. The first half is used as the training set and second as the testing test. The error term e_t is generated with two separate assumptions - normal and student's t distribution. We then use GARCH-MLE and GARCH-SVR to estimate the parameters and forecast variance $\hat{\sigma}^2_{t+1}$.

Now to demonstrate the process, we take an example with the parameter set as: ω = 0.1, α = 0.1 and β = 0.8. The error term e_t series is generated with zero mean and unit variance. The distributions of e_t are normal distribution and student's t with 6 degrees of freedom.

We show the generated series in figure 11 and figure 12. In the two generated series, e_t are generated with normal and student's t distribution. We see that in the QQ plot with student's t distribution, it has significantly heavier tails comparing to the normal distribution. From ACF plots, there isn't obvious autocorrelation for y_t, but a strong autocorrelation in y^2_t.

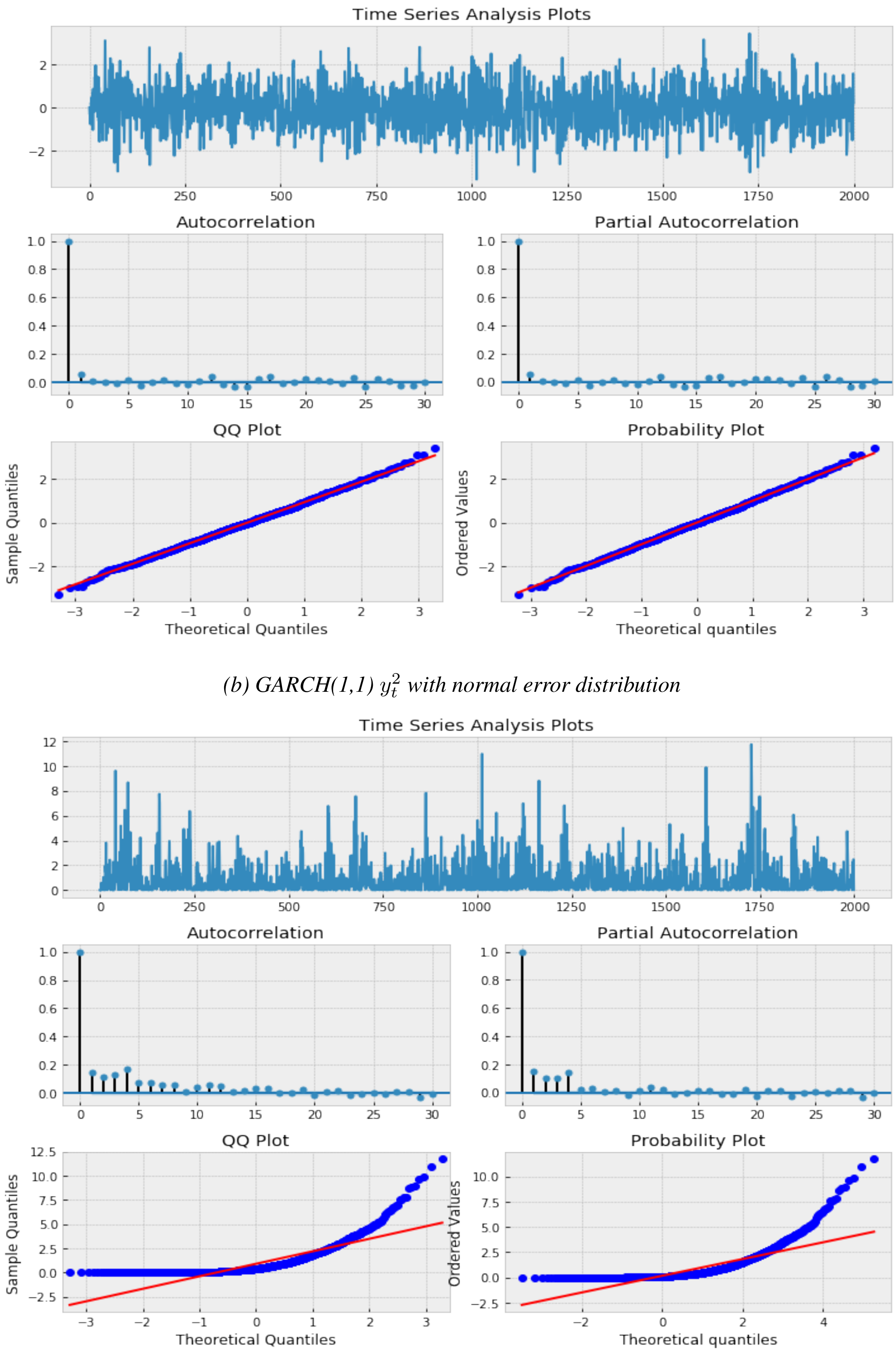
After generating simulated series, we fit the GARCH model with MLE and SVR methods. We store the results separately for different distribution assumptions. Now we get $\hat{\sigma}^2_{t+1}$ from estimated parameters and calculated R^2 from y^2_{t+1} and $\hat{\sigma}^2_{t+1}$. Both in-sample and out-sample R^2 are computed.

Now we proceed with GARCH-SVR for both generated series. According to equation 39, y^2_{t-1} and σ^2_{t-1} in-sample are used as regressor and σ^2_t is used as dependent variable to train the model. There are 3 hyper-parameters to set in the ν-SVR for the grid search. We set C value for loss function as 10, the values of ν in ν-SVR are set in a range between 0.1 and 1, with 0.1 for each step. We use linear, polynomial and rbf kernels. And we apply grid search using eightfold cross validation. In this example, we get ν = 0.1, C set as 10 and linear kernel.

One simulation example is not enough to demonstrate the effect of different methods. Once we plot true returns and forecasted $\hat{\sigma}^2_{t+1}$, the two generated processes are very different from one another, so are the characteristics between training and testing, see in figure 13 and 14.

Figure 11: Simulation process: normal error distribution

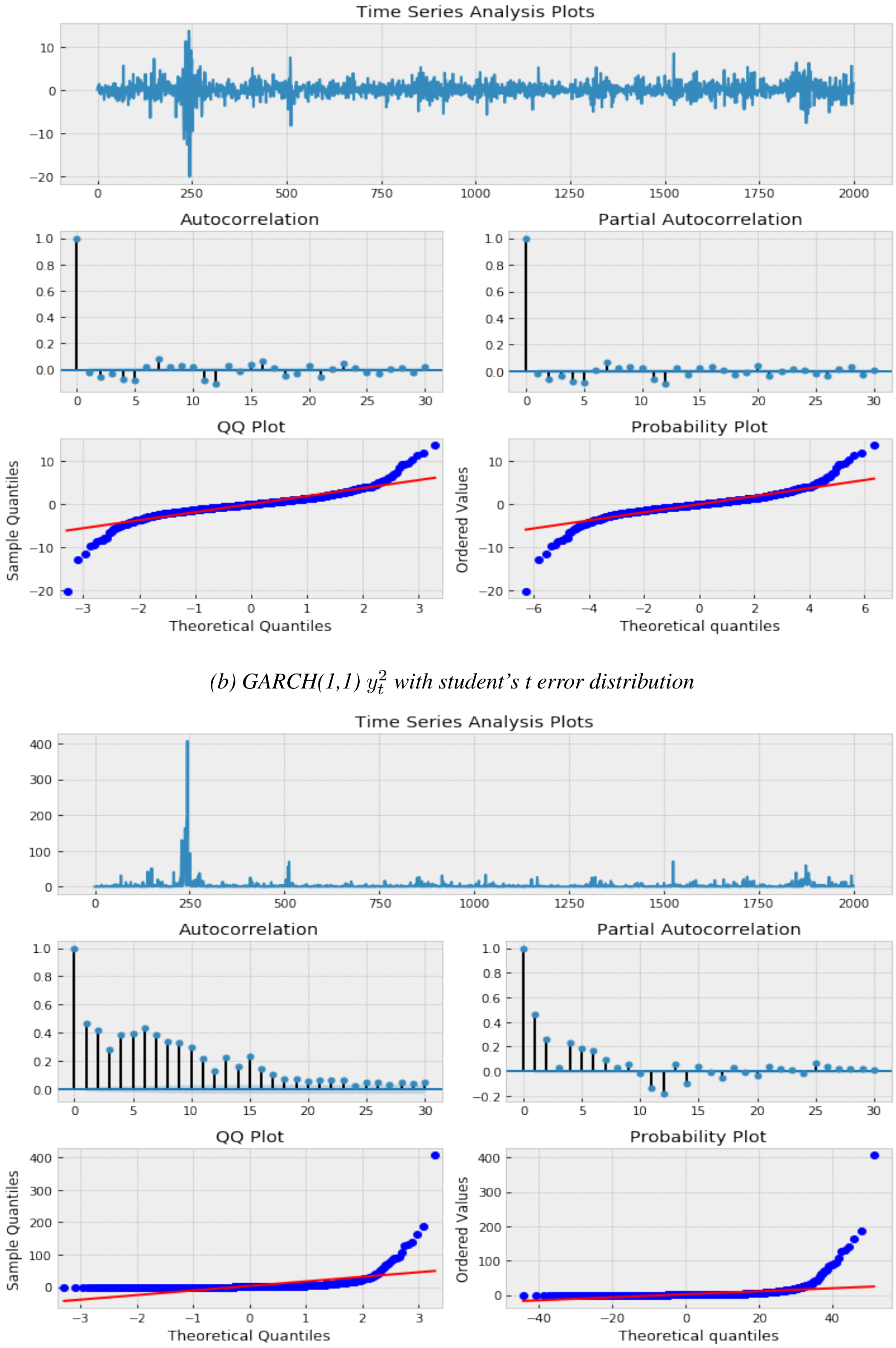
(a) GARCH(1,1) y_t with normal error distribution



(b) GARCH(1,1) y_t^2 with normal error distribution

Source: Own work.

Figure 12: Simulation process: student's t error distribution

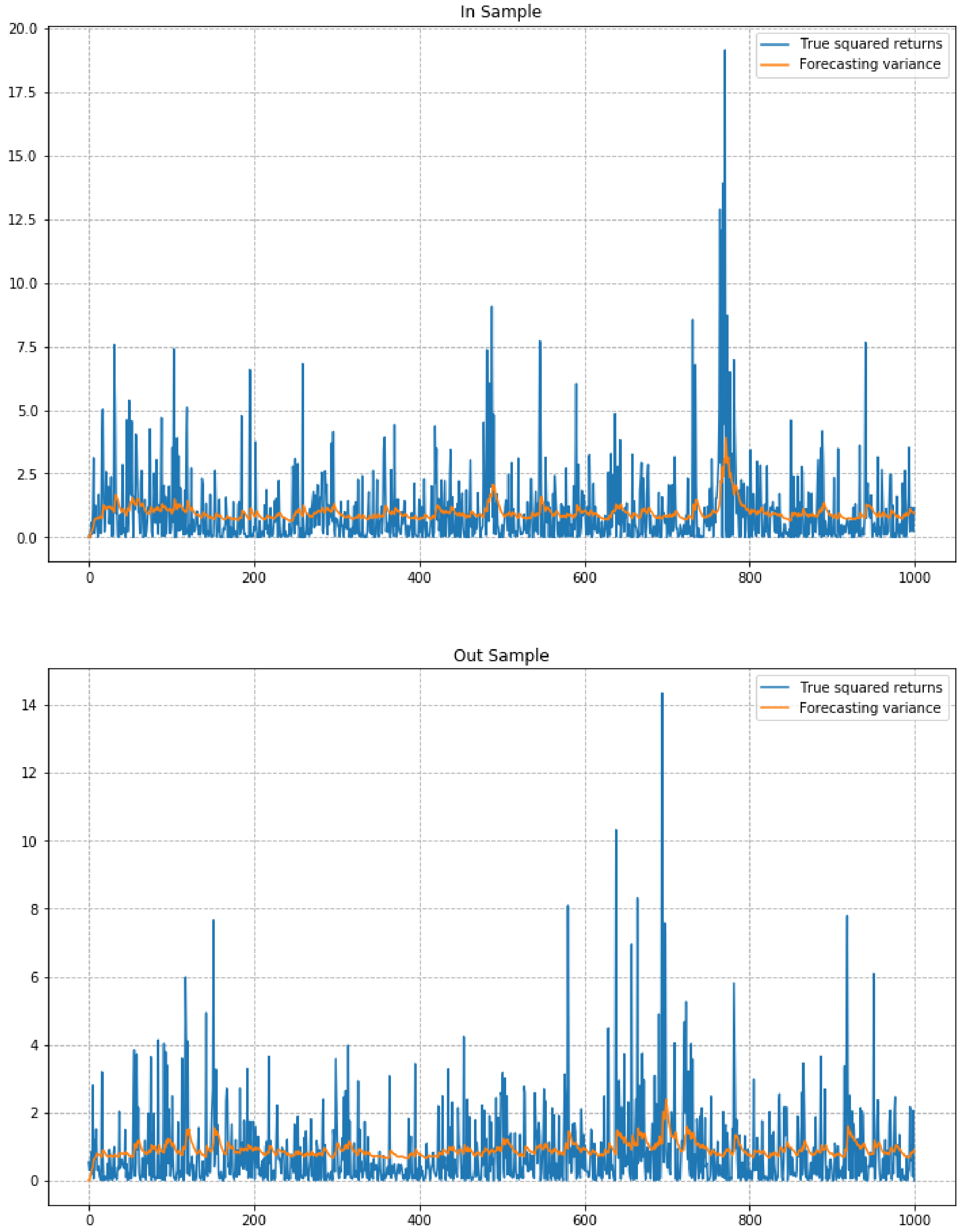


(a) GARCH(1,1) y_t with student's t error distribution

(b) GARCH(1,1) y_t^2 with student's t error distribution

Source: Own work.

Figure 13: An example of GARCH-MLE simulation with normal distribution

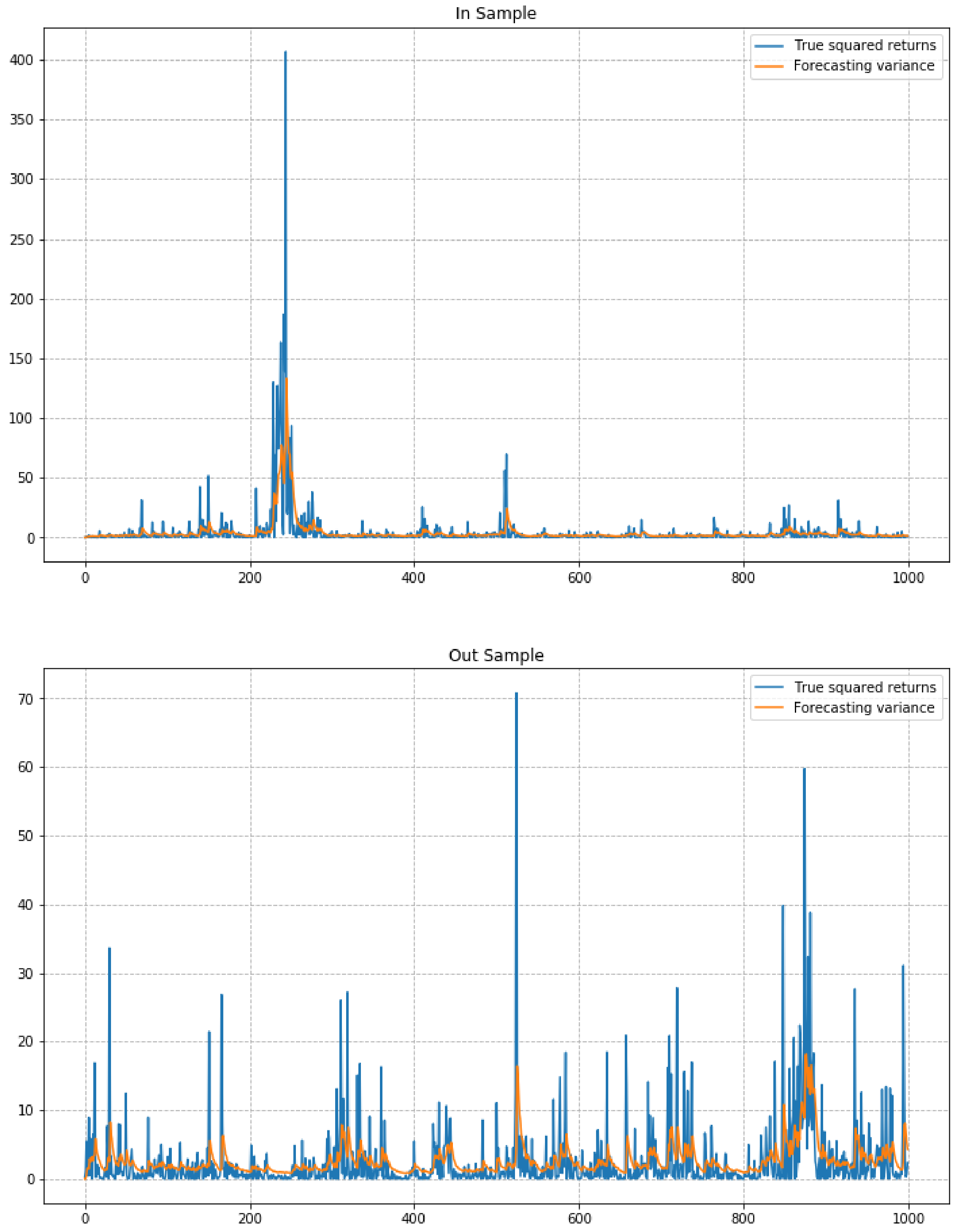


In Sample

Out Sample

Source: Own work.

Figure 14: An example of GARCH-MLE simulation with student's t distribution



Source: Own work.

Therefore, in order to get a better understanding for different estimation methods, we run the simulation process for 100 times and get values of mean of R^2. Two sets of parameters are tested. Each R^2 is stored for both methods and the mean of R^2 is obtained at the end. We present the simulation result in the next section.

2.2.2 Simulation Results

After 100 times of the simulatin process - with 2 parameter settings, 2 distribution assumptions and 2000 sample size for each, we find that GARCH-SVR outperforms GARCH-MLE in variance forecasting. A summary of simulation results R^2 is in table 4. The values of results are varied given different parameter settings. In all cases, GARCH-SVR performs better for out-sample data and GARCH-MLE has better R^2 for in-sample data. At the same time, there is no significant difference between the two methods.

Table 4: GARCH Simulation Results: Mean of R^2 of 100 Simulation Process

Normal Distribution	**In-sample**	**Out-sample**
GARCH-MLE	**11.87 (7.01)**	11.59 (6.87)
GARCH-SVR	11.73 (7.26)	**11.73 (6.92)**

Student's t(6) Distribution	**In-sample**	**Out-sample**
GARCH-MLE	**12.71 (9.20)**	12.89 (9.01)
GARCH-SVR	14.24 (8.33)	**14.52 (8.19)**

ω = 0.075, α = 0.2, β = 0.7

Normal Distribution	**In-sample**	**Out-sample**
GARCH-MLE	**4.62 (2.29)**	2.30 (2.15)
GARCH-SVR	4.52 (2.30)	**2.47 (2.14)**

Student's t(3) Distribution	**In-sample**	**Out-sample**
GARCH-MLE	**4.01 (2.71)**	2.29 (2.44)
GARCH-SVR	3.86 (2.75)	**2.50 (2.42)**

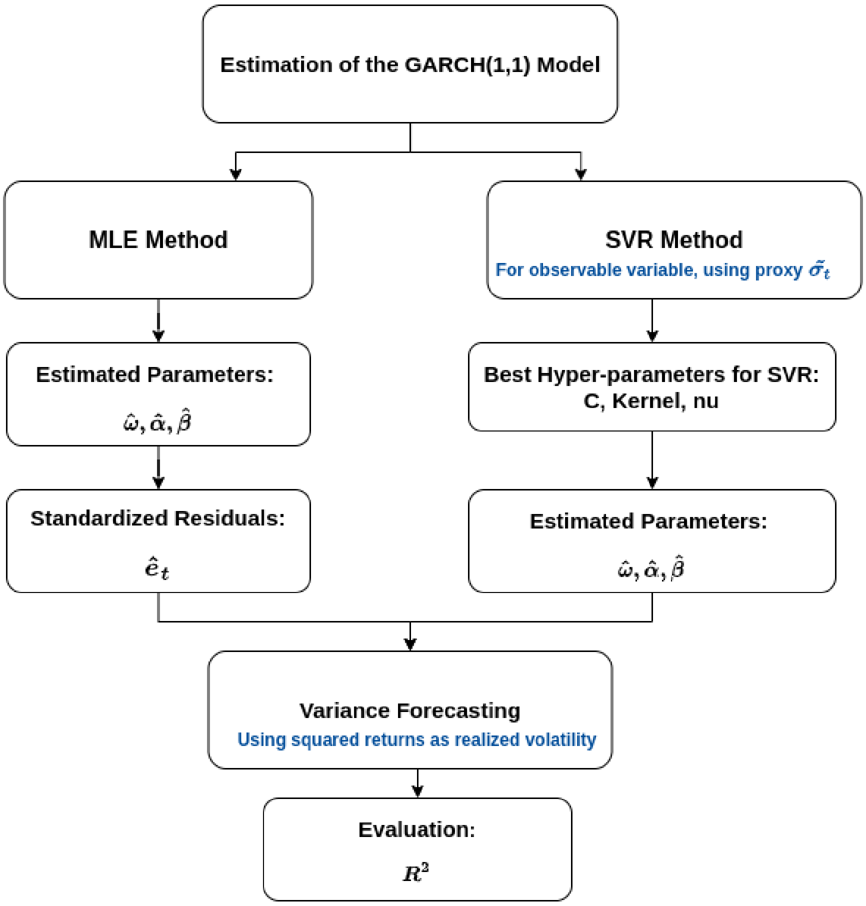
ω = 0.27, α = 0.06, β = 0.9

Source: Own work.

2.3 Empirical Modelling

After the simulation process, we proceed with empirical modelling using SP500 and BTC data. We start with the GARCH-MLE model and continue to the GARCH-SVR model. In the end, we show the estimated parameters and forecastibility R^2.

Figure 15: Empirical Modelling Process


Estimation of the GARCH(1,1) Model
MLE Method
SVR Method
For observable variable, using proxy $\tilde{\sigma_t}$
Estimated Parameters:
$\hat{\omega}, \hat{\alpha}, \hat{\beta}$
Best Hyper-parameters for SVR:
C, Kernel, nu
Standardized Residuals:
$\hat{e}_t$
Estimated Parameters:
$\hat{\omega}, \hat{\alpha}, \hat{\beta}$
Variance Forecasting
Using squared returns as realized volatility
Evaluation:
R^2

Source: Own work.

2.3.1 GARCH-MLE Method

We first fit the GARCH(1,1) model using the MLE method with both normal and student's t error distribution assumption. Estimated parameters $\hat{\omega}$, $\hat{\alpha}$, $\hat{\beta}$ in the GARCH(1,1) model are obtained. Model results with t-statistcs and p values are listed in the summary, see table 5 for SP500 and table 6 for BTC returns. Residuals are plotted in the manner of TSA plots. We use them to observe if they are white noise or if any more autocorrelation is presented. QQ plots are helpful to examine the distribution. In addition, we calculate the standardized residuals to compare the normality - skewness and kurtosis between the financial assets.

For SP500 daily returns, estimated $\hat{\omega}$, $\hat{\alpha}$, $\hat{\beta}$ and degrees of freedom in student's t method are all significant, meaning that the GARCH(1,1) model well captures the effect of conditional variance. In both error distribution methods, the estimated $\hat{\mu}$ is not significant. The estimated parameters are then used for constructing forecasted variance. From the ACF and PACF plots in figure 16, the autocorrelation effect doesn't show in the residuals. From the QQ plot, there shows a heavier tail focusing on the negative side.

Table 5: GARCH(1,1)-MLE SP500 daily summary

Normal error distribution

No. Observations	716
Distribution	Normal
Method	Maximum Likelihood

	Coef	**Std err**	**t**	**P > |t|**	**95 % Conf. Int.**
$\hat{\mu}$	8.11e-03	2.54e-02	0.32	0.75	[-4.16e-02, 5.78e-02]
$\hat{\omega}$	0.075	3.026e-02	2.48	1.33e-02	[1.56e-02, 0.13]
$\hat{\alpha}$	0.20	5.54e-02	3.68	2.33e-04	[9.53e-02, 0.31]
$\hat{\beta}$	0.70	7.25e-02	9.60	8.03e-22	[0.56, 0.84]

Student's t error distribution

No. Observations	716
Distribution	Standardized Student's t
Method	Maximum Likelihood

	Coef	**Std err**	**t**	**P > |t|**	**95 % Conf. Int.**
$\hat{\mu}$	0.03	2.34e-02	1.18	0.24	[-1.82e-02, 7.35e-02]
$\hat{\omega}$	0.05	1.69e-02	3.01	2.61e-03	[1.78e-02, 8.39e-02]
$\hat{\alpha}$	0.22	4.62e-02	4.81	1.54e-06	[0.13, 0.31]
$\hat{\beta}$	0.73	4.50e-02	16.16	1.05e-58	[0.64, 0.82]
nu	6.30	1.52	4.14	3.47e-05	[3.31, 9.27]

Source: Own work.

In BTC data, estimated parameter $\hat{\omega}$ and $\hat{\mu}$ are not significant in either assumptions. The number of observations is around 300 more than SP500 data. Between the two error assumptions, student's t error distribution has a better performance regarding the t-stastics for the estimated parameters. It shows that the estimated $\hat{\alpha}$, $\hat{\beta}$ and the degrees of freedom are statistically significant. While in normal error distribution, only estimated $\hat{\beta}$ has significant t-statistics. The residuals also show much heavier tails on both sides in QQ plots comparing to SP500 daily returns, see in figure 17.

Table 6: GARCH(1,1)-MLE BTC daily summary

Normal error distribution

No. Observations	1010
Distribution	Normal
Method	Maximum Likelihood

	Coef	**Std err**	**t**	**P > |t|**	**95 % Conf. Int.**
$\hat{\mu}$	0.02	6.93e-02	0.22	0.82	[-0.12, 0.15]
$\hat{\omega}$	0.16	0.17	0.95	0.34	[-0.17, 0.50]
$\hat{\alpha}$	0.11	6.09e-02	1.86	6.27e-02	[-6.01e-03, 0.23]
$\hat{\beta}$	0.89	6.00e-02	14.82	1.11e-49	[0.77, 1.00]

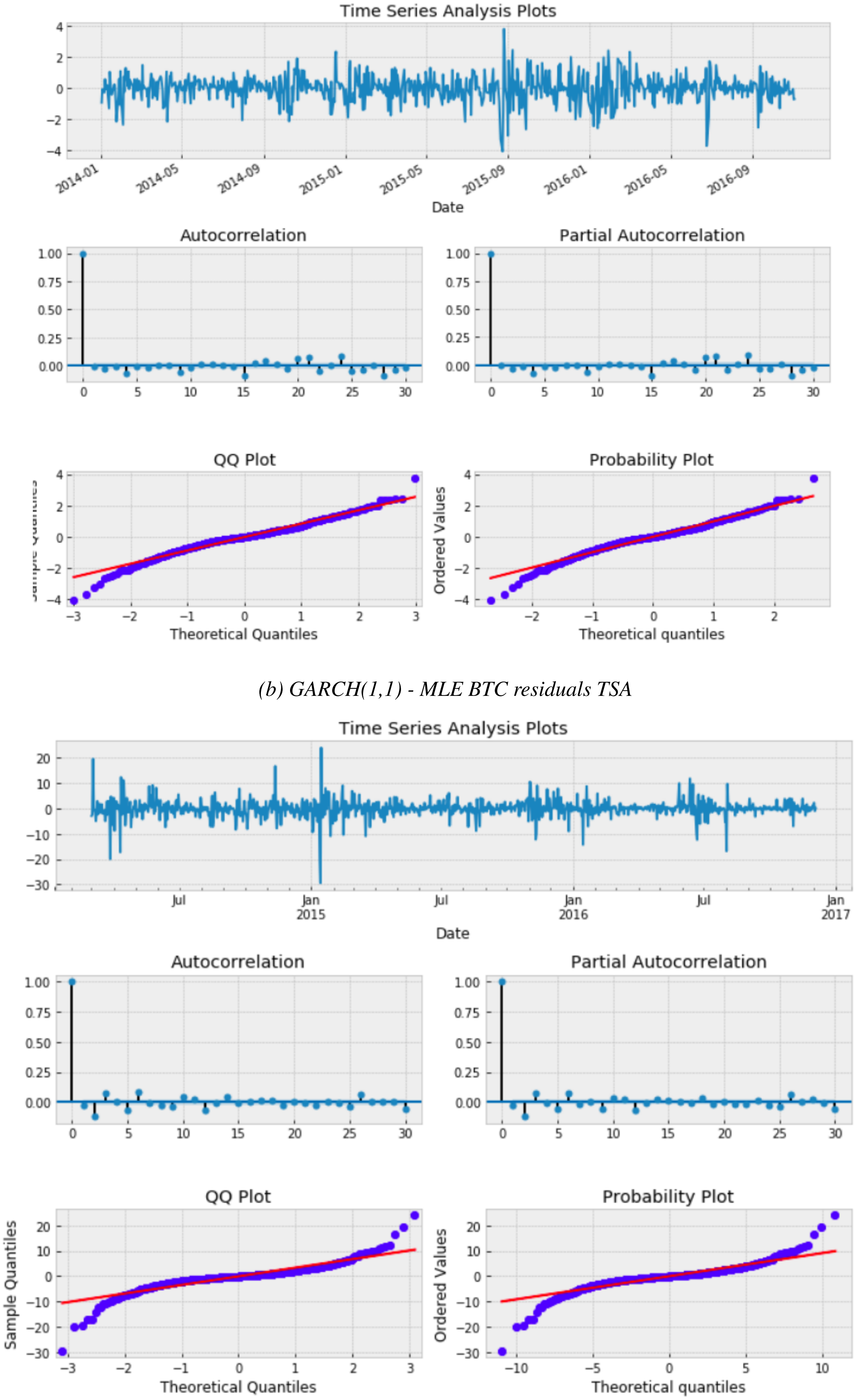
Student's t error distribution

No. Observations	1010
Distribution	Standardized Student's t
Method	Maximum Likelihood

	Coef	**Std err**	**t**	**P > |t|**	**95 % Conf. Int.**
$\hat{\mu}$	-5.42	4.63e-02	-0.12	0.91	[-9.62e-02, 8.54e-02]
$\hat{\omega}$	0.37	0.23	1.63	0.10	[0.12, 0.24]
$\hat{\alpha}$	0.18	3.18e-02	5.69	1.29e-08	[0.12, 0.24]
$\hat{\beta}$	0.82	4.67e-02	17.54	7.02e-69	[0.73, 0.91]
nu	2.76	0.13	20.61	2.17e-94	[2.50, 3.03]

Source: Own work.

Figure 16: GARCH(1,1) - MLE normal residuals TSA

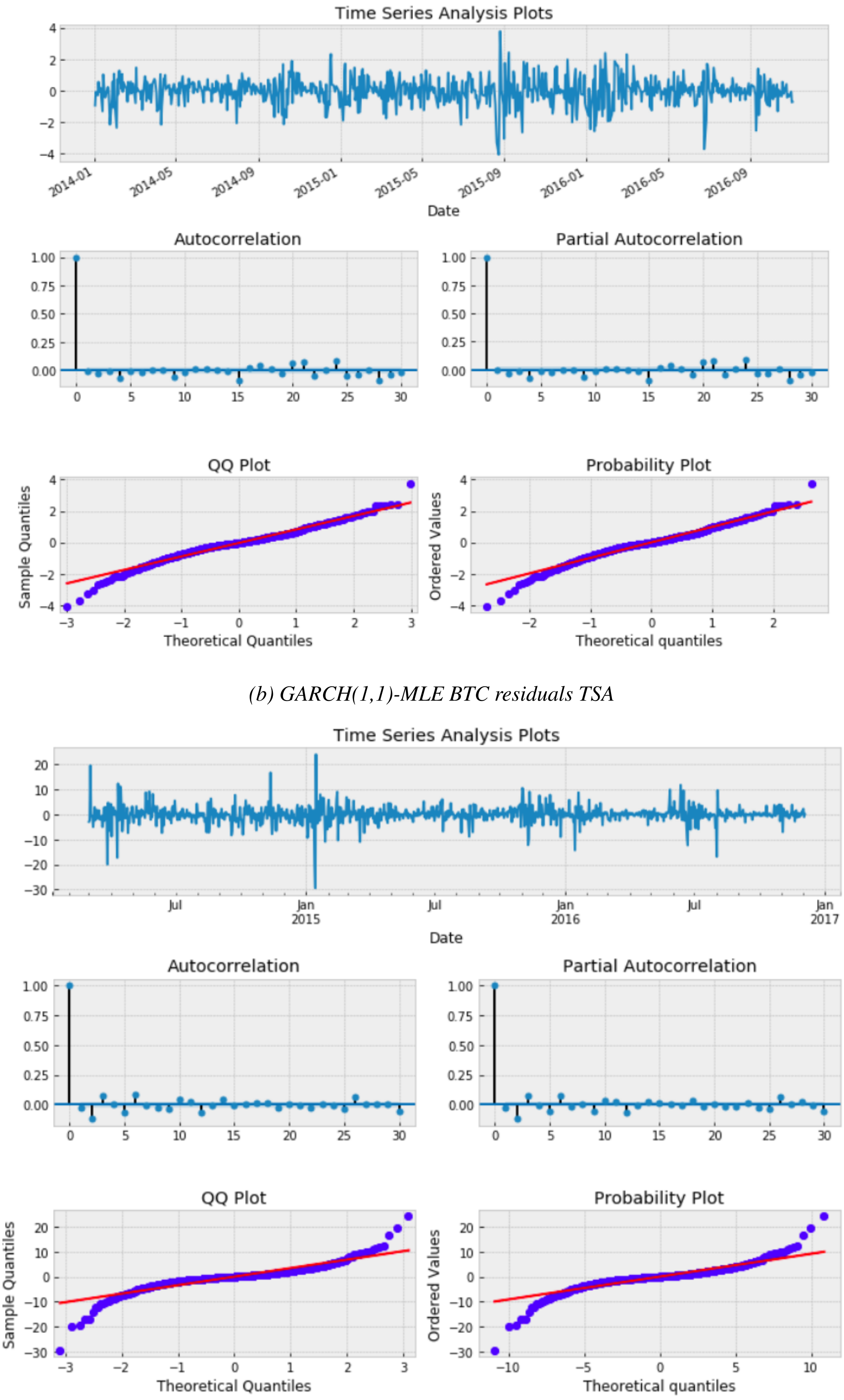


(a) GARCH(1,1) - MLE SP500 residuals TSA

(b) GARCH(1,1) - MLE BTC residuals TSA

Source: Own work.

Figure 17: GARCH(1,1)-MLE students't residuals TSA



(a) GARCH(1,1)-MLE SP500 residuals TSA

(b) GARCH(1,1)-MLE BTC residuals TSA

Source: Own work.

From the observations of the first results after fitting the GARCH(1,1) model, there is a big difference between the empirical data series. The first idea is that the GARCH-MLE model performs better for SP500 daily returns. The model returns a better result of estimated parameters and it doesn't show any autocorrelation in the residuals. Also residuals are much closer to normal distributions comparing to BTC daily returns.

Both residuals are not likely to have normal distributions based on the normality test given by $s^2 + k^2$ in table 7. QQ plots show that both financial assets have heavier tails on the negative side. Besides, BTC daily returns have much heavier tails comparing to SP500 daily returns. With estimated parameters in GARCH(1,1), $\hat{\sigma}^2_{t+1}$ is constructed as forecasted variance. Forecastibility R^2 values are computed and stored. We present the summary results after conducting the GARCH-SVR model.

Table 7: Skewness and kurtosis for $\hat{e_t}$

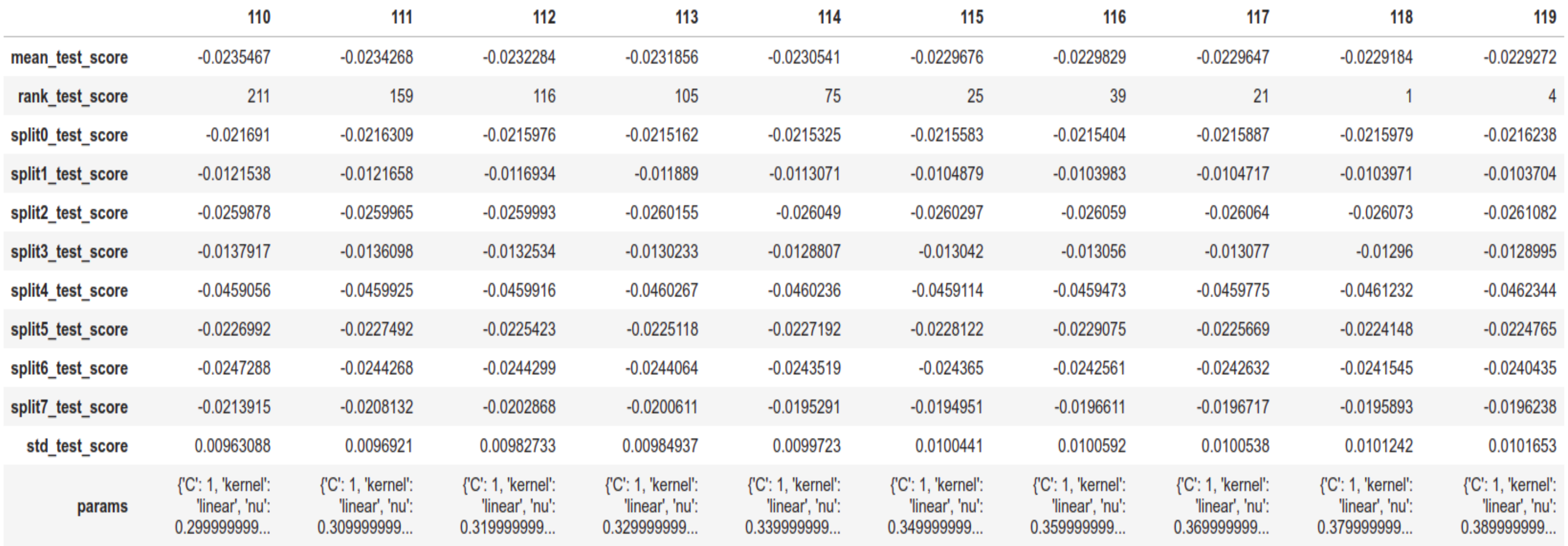
$\hat{e_t}$	**SP500 Daily**	**BTC Daily**
Skewness	-6.34	-9.07
p-value	2.26e-10	1.21e-19
Kurtosis	5.03	13.37
p-value	4.93e-07	9.66e-41
Normality test (s^2 + k^2)	65.52	261.02

Source: Own work.

2.3.2 GARCH-SVR Method

Now for GARCH-SVR, dependent variables are the proxy $\tilde{\sigma_t}^2$ and regressor vector is $\mathbf{x}_t = [y_{t-1}, \sigma_{t-1}]$. When linear kernel is used, the coefficients can be directly obtained for the primal problem. In ν-SVR, C is set as 0.1, 1 and 10. ν is in a range from 0.1 to 1, with 0.01 for each step. Kernel sets as linear, polynomial and rbf. A grid search is done for selecting best hyper-parameters in ν-SVR. An example of grid search and cross validation process is as follows. For each value of C, kernel and ν, training data does a cross validation process and gets a test score - MSLE for each split. Mean of test score is then obtained from all the splits for the specific parameter set. At the end of grid search process, it ranks the test score from lowest to highest. We use scikit-learn Python for the model selection. Since it usually returns higher scores for better performance, we use the negated mean squared error that returns the negated values. An example of a part of the grid search process is in figure 18.

Figure 18: Grid search table



	110	111	112	113	114	115	116	117	118	119
mean_test_score	-0.0235467	-0.0234268	-0.0232284	-0.0231856	-0.0230541	-0.0229676	-0.0229829	-0.0229647	-0.0229184	-0.0229272
rank_test_score	211	159	116	105	75	25	39	21	1	4
split0_test_score	-0.021691	-0.0216309	-0.0215976	-0.0215162	-0.0215325	-0.0215583	-0.0215404	-0.0215887	-0.0215979	-0.0216238
split1_test_score	-0.0121538	-0.0121658	-0.0116934	-0.011889	-0.0113071	-0.0104879	-0.0103983	-0.0104717	-0.0103971	-0.0103704
split2_test_score	-0.0259878	-0.0259965	-0.0259993	-0.0260155	-0.026049	-0.0260297	-0.026059	-0.026064	-0.026073	-0.0261082
split3_test_score	-0.0137917	-0.0136098	-0.0132534	-0.0130233	-0.0128807	-0.013042	-0.013056	-0.013077	-0.01296	-0.0128995
split4_test_score	-0.0459056	-0.0459925	-0.0459916	-0.0460267	-0.0460236	-0.0459114	-0.0459473	-0.0459775	-0.0461232	-0.0462344
split5_test_score	-0.0226992	-0.0227492	-0.0225423	-0.0225118	-0.0227192	-0.0228122	-0.0229075	-0.0225669	-0.0224148	-0.0224765
split6_test_score	-0.0247288	-0.0244268	-0.0244299	-0.0244064	-0.0243519	-0.024365	-0.0242561	-0.0242632	-0.0241545	-0.0240435
split7_test_score	-0.0213915	-0.0208132	-0.0202868	-0.0200611	-0.0195291	-0.0194951	-0.0196611	-0.0196717	-0.0195893	-0.0196238
std_test_score	0.00963088	0.0096921	0.00982733	0.00984937	0.0099723	0.0100441	0.0100592	0.0100538	0.0101242	0.0101653
params	{'C': 1, 'kernel': 'linear', 'nu': 0.299999999...	{'C': 1, 'kernel': 'linear', 'nu': 0.309999999...	{'C': 1, 'kernel': 'linear', 'nu': 0.319999999...	{'C': 1, 'kernel': 'linear', 'nu': 0.329999999...	{'C': 1, 'kernel': 'linear', 'nu': 0.339999999...	{'C': 1, 'kernel': 'linear', 'nu': 0.349999999...	{'C': 1, 'kernel': 'linear', 'nu': 0.359999999...	{'C': 1, 'kernel': 'linear', 'nu': 0.369999999...	{'C': 1, 'kernel': 'linear', 'nu': 0.379999999...	{'C': 1, 'kernel': 'linear', 'nu': 0.389999999...

Source: Own work.

As shown in the grid search table, hyper-parameters for ν-SVR are constructed in different combinations. For each combination, a cross validation process is done with 8-folded splits. For each split, a test score - MSLE is stored. It focuses on the percentual difference between estimated and true values. It treats the difference in same manner when significantly big or small errors happen between predicted and true variance. Test score mean and standard deviation are calculated at the end from the cross validation process. Based on the mean test score, a rank is given for the best hyper-parameters when solving the ν-SVR problem. After the grid search, the best hyper-parameters for SP500 daily returns are C:0.1, ν: 0.38 and linear kernel, and for BTC daily returns are C:10, ν: 0.33 and linear kernel.

The parameters in the GARCH(1,1) model are estimated using ν-SVR with best combination of hyper-parameters. In the case of linear kernel, we get the intercept and weights of the features, that correspond to the parameters in the GARCH(1,1) model. We can now construct forecasted variance: $\hat{\sigma}^2_{t+1} = \hat{\omega} + \hat{\alpha} y_t^2 + \hat{\beta}\sigma_t^2$. To compare the forecastibility, we calculate R^2 from forecasted and true variances. The empirical results are presented in the next section.

2.3.3 Empirical Modelling Results

To evaluate the forecastbility, we show a summary of R^2 that is calculated from forecasted and true variance. Results of R^2 with the GARCH-MLE model and the GARCH-SVR model as well as the estimated parameters are shown in table 8. The forecasted and true variances are as shown in figure 19 to figure 22.

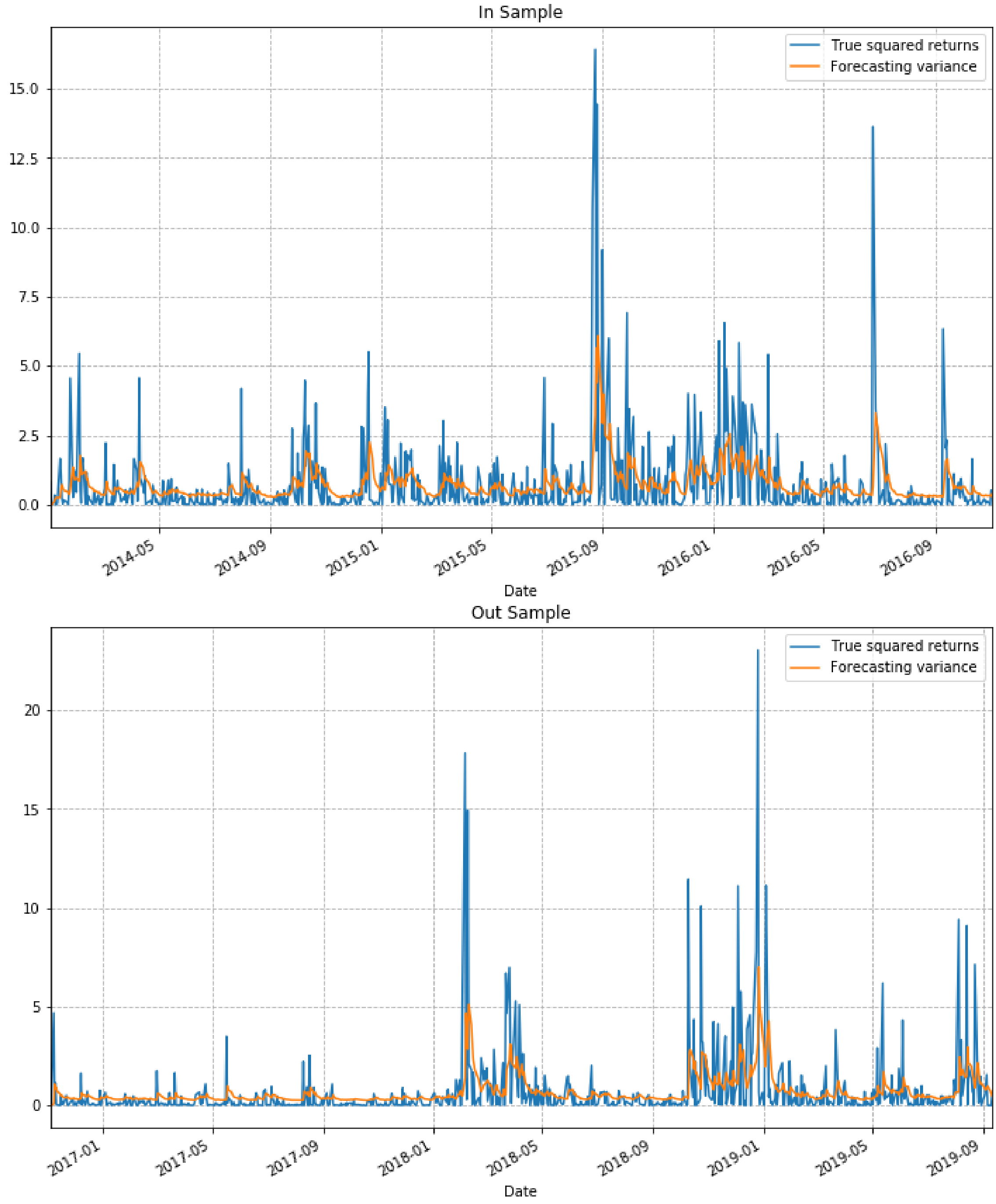
Table 8: GARCH(1,1) Forecasting results of R^2 and estimated parameters

SP500 Daily	In-sample	Out-sample	$\hat{\omega}$	$\hat{\alpha}$	$\hat{\beta}$
MLE-normal	**14.997**	**10.089**	0.075	0.204	0.695
MLE-student's t	13.759	8.952	0.0696	0.0698	0.823
nuSVR	10.532	9.033	0.0508	0.222	0.728

BTC Daily	In-sample	Out-sample	$\hat{\omega}$	$\hat{\alpha}$	$\hat{\beta}$
MLE-normal	2.817	3.206	0.163	0.113	0.886
MLE-student's t	2.659	0.889	0.369	0.181	0.819
nuSVR	**4.407**	**3.434**	0.265	0.059	0.896

Source: Own work.

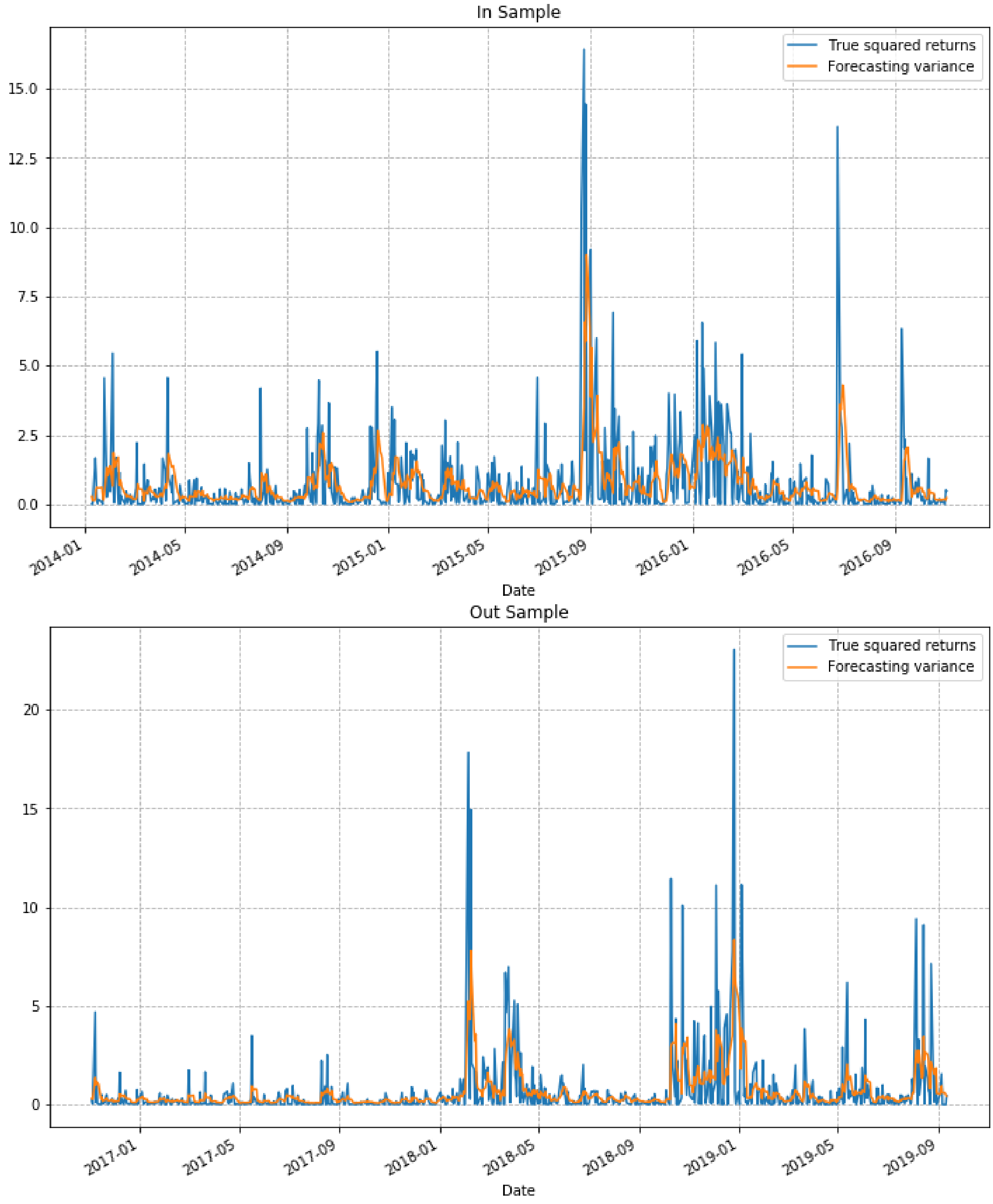
Figure 19: GARCH-MLE forecasted variance for SP500 daily returns


In Sample

Out Sample

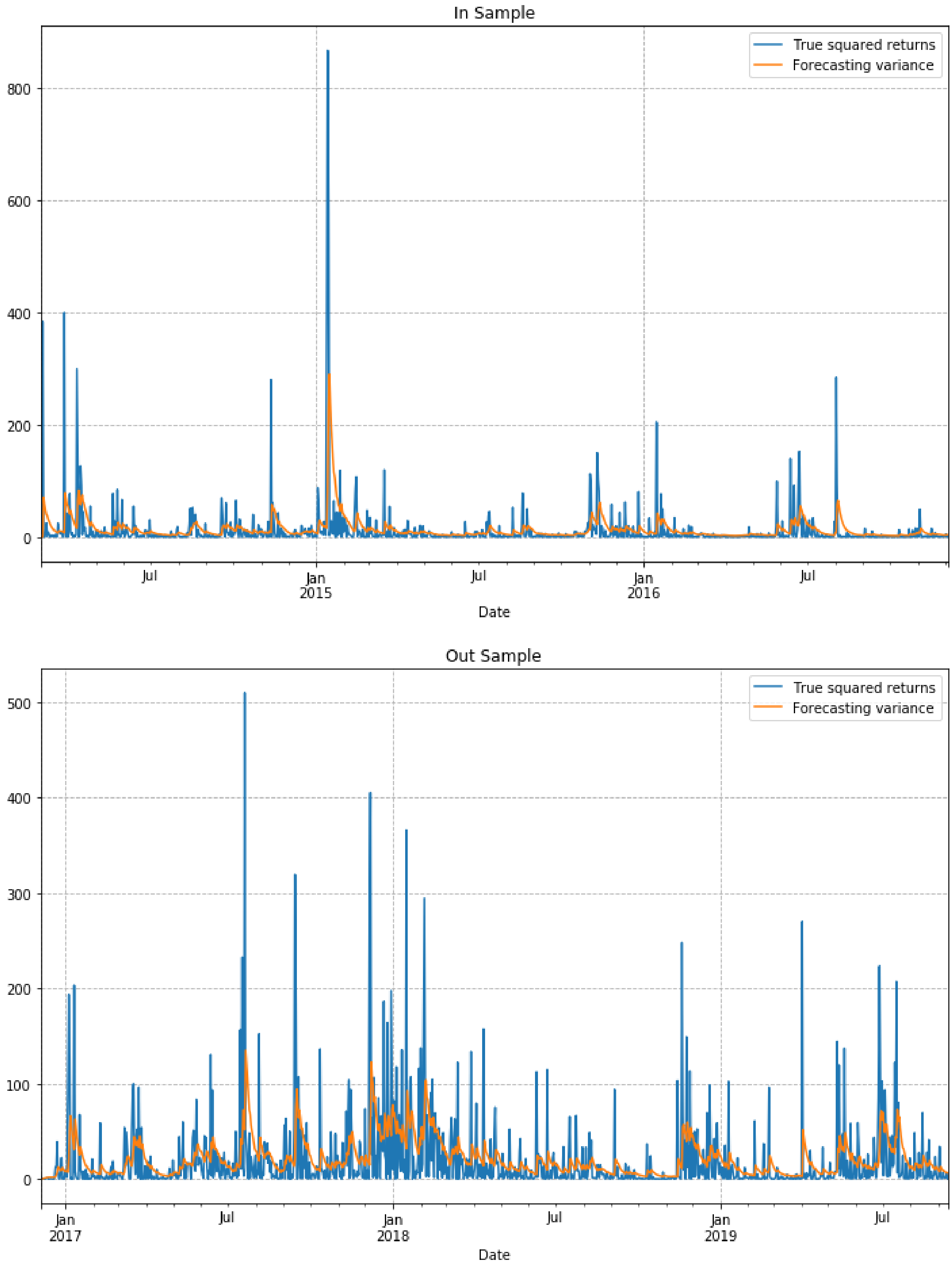
Source: Own work.

Figure 20: GARCH-SVR forecasted variance for SP500 daily returns



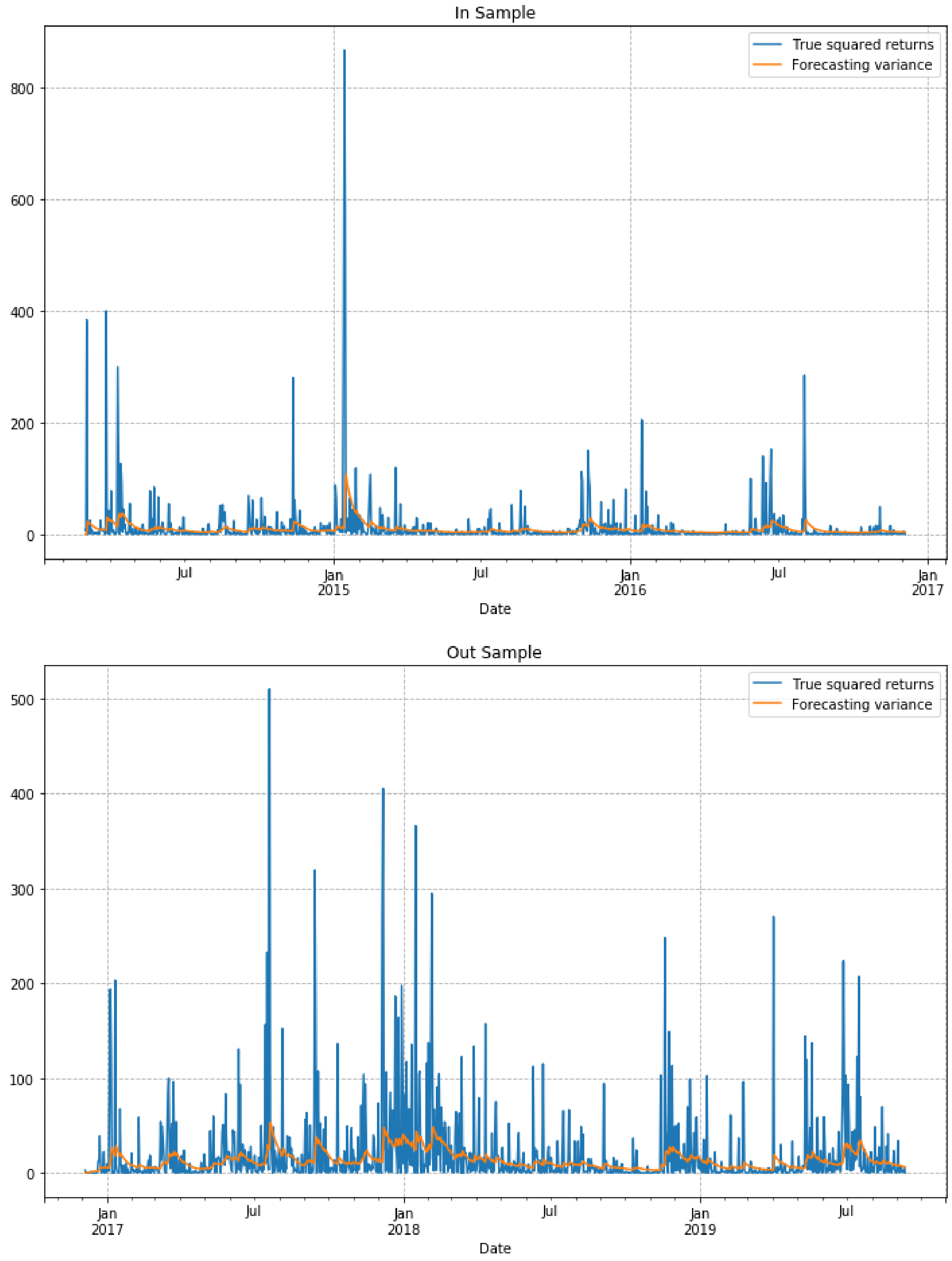
Source: Own work.

Figure 21: GARCH-MLE forecasted variance for BTC/USD daily returns



Source: Own work.

Figure 22: GARCH-SVR forecasted variance for BTC/USD daily returns



Source: Own work.

From computed R^2, GARCH-MLE shows a better result in SP500 daily returns, while GARCH-SVR works better in BTC daily returns. Between the financial assets, R^2 for SP500 daily returns performs better, explaining around 10 percent of variability for out-sample variance. On the other hand, in the same period of BTC returns, it explains only 3.4 percent. Recalling the skewness and kurtosis of the residuals in table 7, both $\hat{e}_t$ are not likely to be normal. BTC daily returns have more negative skewness and bigger positive kurtosis. The estimated intercept $\hat{\omega}$ in BTC returns is also much bigger than the one in SP500 returns.

From the plots of forecasted variance, the method with overreaction leads to a lower R^2 for its performance. In SP500 daily returns, the GARCH-SVR method has a stronger reaction when variance has significant changes, for example after January in 2018. On the other hand, GARCH-MLE has a lower reaction towards extreme events. In BTC daily returns, it is similar case that GARCH-SVR has better performance with less reactions to significant spikes. The reasons that empirical modelling shows different results as the simulation process can be the different characteristics of the two markets: SP500 has a much longer history than the crypto market, BTC returns are much more volatile and have heavier tails.

CONCLUSION

In this thesis, we focus on volatility forecasting using the General Autoregressive Conditional Heteroskedasticity (GARCH) model. The basic idea is to use two different methods for parametenr estimation in the GARCH(1,1) model - they are Maximum Likelihood Estimatio (MLE) and Support Vector Regression (SVR) methods. The main difference between the two is that we don't need an assumption for error distribution in the GARCH-SVR estimation process. In the GARCH-MLE model, we usually assume error term e_t having normal or student's t distribution.

Empirical data contains SP500 and Bitcoin daily returns. The data series has around 1500 and 2000 data samples respectively in the period of 2014 - 2019. After data transformation, both return series y_t have zero mean with negative skewness and positive kurtosis. BTC has very high returns and very volatile in some extreme occasions. From the TSA plots, we don't see an autocorrelation for y_t but a strong autocorrelation for squared return y_t^2 in both cases. The clustering effect also exists. In terms of the distributions, SP500 and BTC returns have heavy tails on both sides. When comparing the two, BTC has a heavier tail on the positive side.

The simulation process presents how we use the GARCH-MLE and GARCH-SVR to forecast variance. We start with one specific example of simulated series. Data is simulated under framework of GARCH by setting specific parameters. To evaluate the ability of forecasting variance, R^2 from one specific example is not enough. We generate data series 100 different times for each distribution assumption - normal and student's t with 6 degrees of freedom. We repeat the process and obtain mean of R^2 at the end. The result shows that GARCH-SVR outperforms GARCH-MLE in variance forecasting. The values vary given different parameter settings, i.e. $R^2 = 14.52$ and $R^2 = 2.5$ - both using the GARCH-SVR model. In all cases, GARCH-SVR has better results of R^2 for out-sample data while GARCH-MLE has better R^2 for in-sample data. There is no significant difference between the two methods. From the simulation process, we see that the GARCH-SVR method has better performance in terms of parameter estimation and variance forecasting since we don't need assumptions for error distributions.

In empirical modelling, the error term e_t becomes important. We check distribution of the standardized residuals and its normality calculated from skewness and kurtosis. The purpose is to see the difference between empirical data and the assumptions. First, there doesn't show the autocorrelation of residuals in both cases according to ACF plots. Second, both residuals are not likely to have normal distribution. BTC's standardized residuals $\hat{e}_t$ has much bigger value of $s^2 + k^2$, especially the kurtosis. Third, the estimated parameter $\hat{\omega}$ for BTC returns is not significant. It performs slightly better in case of student's t assumption but still not statistically significant. From QQ plots, we also observe much heavier tails in BTC data.

In the GARCH-SVR model, We first look for the best hyper-parameters in SVR through a grid search. It returns a set of values: C, kernel and ν, with which it has the lowest error score. We get linear kernel as the best kernel for both return series. Then we are able to estimate the parameters in the GARCH(1,1) model by extracting the intercept and feature weights. We also use a proxy in this method. In GARCH-SVR, we need observable variables for model learning. We use a moving average of the contemporaneous and four lagged squared returns at each point. A proxy also helps to smooth and eliminate noise from the realized data.

The empirical results show that GARCH-MLE has a better performance for SP500 daily returns while GARCH-SVR is better for BTC daily returns. Both residuals are not likely to be normally distributed. For BTC daily returns, the residuals are much more skewed to the right and have higher kurtosis. The estimated parameters are very different. The estimated intercept $\hat{\omega}$ in BTC returns is much bigger than the one in SP500 returns. The evaluation R^2 of SP500 explains around 10 percent of variability for out-sample data. In same period, R^2 is only around 3.4 for BTC returns. From the plots of forecasted and true variance, the method that overreacts gains a lower R^2. When SP500 daily is very volatile, the GARCH-MLE model has less reaction forecasting variance comparing to the GARCH-SVR model. It is similar for the BTC returns that the GARCH-SVR has better performance without an overreaction towards significant spikes.

The reasons that empirical modelling shows different result than the simulation process can be the following. First, the two markets have very different characteristics. SP500 has a much longer history than crypto markets. Second, the returns of BTC are much more volatile and have heavier tails on both sides. Recalling the residual distributions, $\hat{e}_t$ has bigger value of $s^2 + k^2$. In other words, BTC residuals are less likely to be normally distributed comparing to SP500. Third, the data samples we use in training and testing sets cover a long period of time. It is not frequent enough for the BTC returns due to its high volatility and fast changing characteristics.

For this study, we have also tried 15-min intervals besides the daily returns for both SP500 and BTC returns. When checking the TSA and ACF plots, we find that SP500 15-min squared returns have 26 lags of autocorrelation, meaning it is daily autocorrelated based on the trading hours per day. BTC/USD 15-min squared returns show 4 lags of autocorrelation, meaning an hourly autocorrelation. It is worthwhile to try other different data intervals in the future study, i.e. hourly returns.

There are many ways to improve the work in this thesis. First, it shows that the model performance is dependent on the characteristics of the market. More financial assets can be included for empirical modelling. For example, to include different crypto currencies that share similar characteristics. We can also use other proxy as realized volatility, e.g. intra-day returns. Furtheremore, a rolling window basis can be applied on the GARCH models. The process is computationally demanding with big datasets especially for the GARCH-SVR model. It is expected to improve variance forecasting and get a better understanding of the forecastibility between the two estimation methods.

REFERENCE LIST

1. Andersen, T. G., Bollerslev, T., Diebold, F. X., & Labys, P. (2003). Modeling and forecasting realized volatility. *Econometrica*, *71*(2), 579–625.
2. Anscombe, F. J. & Glynn, W. J. (1983). Distribution of the kurtosis statistic b 2 for normal samples. *Biometrika*, *70*(1), 227–234.
3. Bezerra, P. C. S. & Albuquerque, P. H. M. (2017). Volatility forecasting via svr-garch with mixture of gaussian kernels. *Computational Management Science*, *14*(2), 179–196.
4. Black, F. (1976). Studies of stock price volatility changes. *Proceedings of the 1976 Meetings of the American Statistical Association*, 171–181.
5. Bollerslev, T. (1986). General autoregressive conditional heteroskedasticity. *Journal*

of Econometrics, *31*, 307–327.

6. Bollerslev, T., Engle, R. F., & Nelson, D. B. (1994). *The Handbook of Econometrics (4)*. Amsterdam: Elsevier.
7. Bouchaud, J. P. & Potters, M. (1999). *Theory of Financial risk: From statistical Physics to Risk Management*. Cambridge: Cambridge University Press
8. Bouoiyour, J. & Selmi, R. (2014). "what bitcoin looks like?". *Munich Personal Repec Archive Paper No. 58091*.
9. Box, G. E. P., Jenkins, G. M., & Reinsel, G. C. (2008). *Time Series Analysis, Forecasting and Control*. New Jersey: Wiley.
10. Brailsford, T. J. & Faff, R. W. (1996). An evaluation of volatility forecasting techniques. *Journal of Banking and Finance*, *20*(3), 419–438.
11. Cao, L. & Tay, F. E. H. (2001). Financial forecasting using support vector machines. *Neural Computing and Applications*, *10*, 184–192.
12. Chang, C. & Lin, C. (2001, Last updated: 2013). LIBSVM: A Library for Support Vector Machines. Retrieved from https://www.csie.ntu.edu.tw/cjlin/papers/libsvm.pdf.
13. Chen, S., Härdle, W. K., & Jeong, K. (2009). Forecasting volatility with support vector machine-based garch model. *Journal of Forecasting*, *29*(4), 406–433.
14. Chen, S., Jeong, K., & Härdle, W. (2008). Support vector regression based garch model with application to forecasting volatility of financial returns. *SFB 649 "Economic Risk"*, Discussion Paper (2008-014).
15. Cortes, C. & Vapnik, V. (1995). Support-vector networks. *Machine Learning*, *20*, 273–297.
16. D′Agostino, R., Belanger, A., & D′Agostino, J. R. B. (1990). A suggestion for using powerful and informative tests of normality. *The American Statistician*, *44*(4), 316–321.
17. Donaldson, R. G. & Kamstra, M. (1996). An artificial neural network-garch model for international stock return volatility. *Journal of Empirical Finance*, *70*, 17–46.
18. Drucker, H., Burges, C., Kaufman, L., Smola, A. J., & Vapnik, V. (1996). Support vector regression machines. *Advances in Neural Information Processing Systems*, *9*, 155–161.
19. Dutta, A. (2014). Modelling volatility: symmetric or asymmetric garch models. *Journal of Statistics: Advances in Theory and Applications*, *12*(2), 99–108.
20. Engle, R. F. (1982). Autoregressive conditional heteroscedasticity with estimates of the variance of united kingdom inflation. *Econometrica*, *50*(4), 987–1008.
21. Engle, R. F. (1983). Estimates of the variance of u.s. inflation based upon the arch model. *Journal of Money, Credit and Banking*, *15*(3), 286–301.
22. Engle, R. F. & Patton, A. J. (2001). What good is a volatility model. *Quantitative Finance*, *1*, 237–245.
23. Fletcher, R. & Sainz de la Maza, E. (1989). Nonlinear programming and nonsmooth

optimization by successive linear programming. *Mathematical Programming*, *43*, 235–256.

24. Franses, P. H. & Dijk, D. V. (1995). Forecasting stock market volatility using (nonlinear) garch models. *Journal of Forecasting*, *15*(3), 229–235.
25. Gavrishchaka, V. V. & Banerjee, S. (2006). Support vector machine as an efficient framework for stock market volatility forecasting. *Computational Management Science*, *3*, 147–160.
26. Gavrishchaka, V. V. & Ganguli, S. B. (2003). Volatility forecasting from multiscale and high-dimensional market data. *Neurocomputing*, *55*, 285–305.
27. Hansen, P. R. & Lunde, A. (2005). A forecast comparison of volatility models: Does anything beat a garch(1,1)? *Journal of Applied Econometrics*, *20*(7), 873–889.
28. Hofmann, T., Schölkopf, B., & Smola, A. J. (2008). Kernel methods in machine learning. *The Annals of Statistics*, *36*(3), 1171–1220.
29. Ji, S. (2017). An illustration of kernel trick in svm. Retrieved from https://en.wikipedia.org/wiki/kernel_method/media/file:Kernel_trick_idea.svg.
30. Kleynhans, T., Montanaro, M., Gerace, A., & Kanan, C. (2017). Predicting Top-of-Atmosphere Thermal Radiance Using MERRA-2 Atmospheric Data with Deep Learning. *Remote Sensing*, *9*(11), 1133.
31. Meech, J. and Gu, R. (2014). Bitcoin - the "new gold" for a "safe-haven" investment. Unpublished manuscript available at http://www.jmeech.mining.ubc.ca.
32. Ou, P. & Wang, H. (2011). Modeling and forecasting stock market volatility by gaussian process based on garch, egarch and gjr models. *Proceedings of the World Congress on Engineering*, *1*.
33. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, *12*, 2825–2830.
34. Perez-Cruz, F., Afonso-Rodriguez, J. A., & Giner, J. (2003). Estimating garch models using support vector machines. *Quantitative Finance*, *3*, 1–10.
35. Poon, S. & Granger, C. W. J. (2003). Forecasting volatility in financial markets: A review. *Journal of Economic Literature*, *41*(2), 478–539.
36. Schölkopf, B., Platt, J., Shawe-Taylor, J., & Smola, A. J. (2001). Estimating support of a high-dimensional distribution. *Neural Computation*, *13*(7), 1443–1471.
37. Schölkopf, B., Smola, A. J., Williamson, R. C., & Bartlett, P. (2000). New support vector algorithms. *Neural Computation*, *12*(5), 1207–1245.
38. Sheppard, K. (2019). Univariate volatility modelling, bootstrapping, multiple comparison procedures and unit root tests. Retrieved from https://arch.readthedocs.io/en/latest/.
39. Smola, A. J., Murata, N., Schölkopf, B., & Müller, K. R. (1998). Asymptotically optimal choice of ε-loss for support vector machines. *ICANN 98*, *International*

Conference on Artificial Neural Network, 105–110.

40. Smola, A. J. & Schölkopf, B. (2004). A tutorial on support vector regression. *Statistics and Computing*, *14*, 199–222.
41. Sollis, R. (2012). *Empirical Finance for Finance and Banking*. New Jersey: Wiley.
42. Triacca, U. (2007). On the variance of the error associated to the squared returns as proxy of volatility. *Applied Financial Economics Letters*, *3*(4), 255–257.
43. Tsay, R. S. (2010). *Analysis of Financial Time Series (Third Edition)*. New Jersey: Wiley.
44. Vapnik, V. (1995). *The Nature of Statistical Learning Theory.* Berlin: Springer.
45. Vapnik, V. (1998). *Statistical Learning Theory*. New Jersey: Wiley.
46. Whittle, P. (1951). *Hypothesis Testing in Time Series Analysis*. Upssala: Almqvist and Wiksells bokt.
47. Wright, S. (1921). Correlation and causation. *Journal of Agricultural Research*, *20*, 557–585.
48. Zivot, E. (2008). Practical issues in the analysis of univariate garch models. *Handbook of Financial Time Series*, 113–155.

APPENDICES

Appendix 1: Povzetek

Osredotočimo se na problem modeliranja in napovedovanja variance časovnih vrst z GARCH modeli. Posebej raziskujemo in primerjamo naslednja dva pristopa ocenjevanja parametrov GARCH modela: dobro uveljavljen pristop po metodi največjega verjetja (GARCH-MLE) in inovativen pristop z uporabo algoritma strojnega učenja podpornih vektorjev za regresijo (GARCH-SVR).

Sodeč po empiričnem modeliranju, dnevni donosi niso normalno distribuirani in se običajno ponašajo z debelimi repi. Predvidevamo, da model GARCH-SVR podaja bolj robustne ocene, saj ne predvideva doloCenega vzorca distribucije, hkrati pa ima tudi višji nivo fleksibilnosti.

Najprej predstavimo osnovne napovedne modele iz analize časovnih vrst, vključno z GARCH(p, q). Nato pojasnimo in izpeljemo uporabo algoritma podpornih vektorjev za oceno parametrov modela GARCH. Za primerjave uporabljamo GARCH(1, 1) model. Nato izvedemo simulacijsko študijo. Pristop GARCH-SVR lažje razloži variabilnost napovedane variance v našem simulacijskem postopku. Sicer med obema pristopoma ni večjih razlik. Za porazdelitev napak uporabljamo normalno in t-porazdelitev.

Na koncu izvedemo še empirično raziskavo, ki temelji na podatkih SP500 in BTC/USD od leta 2014 do leta 2019. Obe porazdelitvi dnevnih donosov nista normalni in imata debele repe, zlasti na negativni strani. Naša študija kaže mešane rezultate. Za dnevne donose BTC/USD je bolj uspešen GARCH-SVR. Toda za dnevne donose SP500 je bolj uspešen GARCH-MLE pristop. Zaključimo, da je GARCH-SVR bolj primeren pristop v primeru BTC/USD in GARCH-MLE bolj primeren v primeru SP500. Za nadaljne predvidevanje variance, lahko vključimo različne finančne dobrine, kot tudi urne podatke. V bodoče lahko primerjamo ocenjevanje parametrov modela na podlagi drsečega okna.

Appendix 2: Summary in English

We focus on the problem of time-series variance modeling and forecasting with GARCH models. We specifically investigate and compare the following two approaches of estimating the GARCH parameters. The use of well established Maximum Likelihood Estimation (MLE) method for estimating the GARCH parameters which we refer to as the GARCH-MLE approach. And an innovative approach of using Support Vector Machine learning algorithm for regression (SVR) for estimating the GARCH parameters which we refer to as the GARCH-SVR.

From the empirical modelling, daily returns are not normally distributed and usually have heavy tails. The GARCH-SVR is expected to give more robust estimators since it does not assume a particular distribution and it has a higher level of flexibility.

We first introduce basic models for time series data including the GARCH(p, q) models. Then we explain support vector regression and derive the GARCH-SVR approach for estimating parameters in the GARCH model. We use GARCH(1, 1) model for comparisons. Second we perform a simulation study. The GARCH-SVR is able to better explain variability of forecasted variance in out-sample data in our simulation process. Otherwise there are no significant differences between the two approaches. For the distribution of errors, we use both normal and student's t-distributions.

Finally, we follow up with empirical study based on SP500 and BTC/USD data from 2014 until 2019. Both distributions of daily returns are not normal and tend to have heavy tails, especially on the negative side. Our study shows mixed results. For BTC/USD daily returns, the GARCH-SVR model achieves better performance. But for SP500 daily returns, the GARCH-MLE is better. We conclude that the GARCH-SVR approach is better in the case of Bitcoin and the GARCH-MLE is better in the case of SP500 returns. For future work on variance forecasting we can include different financial assets as well as hourly data. We can also compare model parameters estimation on a rolling window basis.