

Predictive Analysis for Financial Volatility

Bachelor Thesis

Mathijs de Lepper

Supervised by:

Dr. Tristan van Leeuwen

Prof. Dr. Roberto Fernández



Universiteit Utrecht

Department of Mathematics

Utrecht University

The Netherlands

December 1, 2015

Abstract

In this thesis we take a closer look at the *generalized autoregressive conditional heteroscedasticity* (GARCH) model, which models the variance of the error term in particular stochastic time series. The standard procedure for estimating the model parameters is the maximum likelihood (ML) estimation method. We instead use machine learning to estimate parameters of a *GARCH*(1, 1) model for a specific dataset and find that a machine learning model is able to find the ML estimator when it exists. We conclude that machine learning is a good alternative to the ML estimation method and that it has the potential to better predict data.

Contents

1	Introduction	3
2	Financial Framework	3
3	Mathematical framework	6
4	The GARCH(p, q) process	8
5	Parameter optimization	11
6	An empirical example	15
7	Discussion	25

1 Introduction

Financial markets react nervously to world’s events. Times of economic crises, environmental changes, political disorders, wars and other stress periods have huge impact on financial assets prices. They cause asset prices to fluctuate very much, making it harder to accurately model the time series that predict these values.

It was up to roughly three decades ago that conventional time series and econometric models assumed constant variance of the errors in their models. Yet they found out that the variance changed over time. Hence “a theory for dynamic volatilities [was] needed; this is the role that is filled by the ARCH models and their many extensions” [4]. One of these extensions is the *Generalized Autoregressive Conditional Heteroscedasticity* (GARCH) model, introduced by Tim Bollerslev in 1986. The goal of this thesis is to take a closer look at the GARCH model and estimate its parameters through machine learning. The thesis will be divided into several sections, starting with a short introduction on financial markets. Here we present the assumptions we make on the market and the empirical regularities that tend to occur. Next we present the probabilistic and measure-theoretic notions underlying the GARCH model, after which we describe the model itself. The subsequent section discusses the basic ideas of machine learning and lays the theoretical foundation for the type of machine learning model used in the empirical example in section 6. Finally, we finish with the discussion, summarizing and reviewing the work we have done.

2 Financial Framework

The following part gives some background on finance, a branch where the GARCH model is extensively used. We describe the financial market, the assumptions we make on the market and what empirical regularities tend to occur. If the reader is familiar with finance, he/she can proceed to the next section.

Financial theories are subjective. In other words, there are no proven laws in finance, but rather ideas that try to explain how the market works [1].

There are several financial markets. The ones we deal with throughout this thesis are *the financial asset markets*. Financial asset markets allow *investors* to buy and sell *assets*. Assets are resources with economic value, which change over time. If an asset is priced P_t at time t it is highly likely that the price will be different at time P_{t+k} for $k > 0$. This difference in price makes it possible for investors to make a profit, provided they buy and sell at the right moments. We thus say that

investors buy or sell assets because of the expected *return*. The return R_t at time t is the percentage indicating the gain/loss of the asset's price since time $t - 1$. More precisely:

$$R_t = \frac{P_t - P_{t-1}}{P_{t-1}}.$$

Due to practical reasons, log returns are used. The log return is defined as follows:

$$r_t = \log(R_t) = \log\left(\frac{P_t}{P_{t-1}}\right). \quad (1)$$

As asset prices change over time, investors can make a profit by buying and selling at the right moment. We assume that the market fully reflects available information. This is known as the *efficient market hypothesis* (EFM). The EFM makes the following predictions about the market:

- The price is always right. The current asset price fully reflects available information and any new information will be taken into account and readjust the price before investors can trade on it, i.e. an efficient market adjusts its prices quickly and correctly.
- Investors cannot beat the market. This is also known as the *no arbitrage* condition and simply means that you cannot make a profit without taking risk.

Remark. Due to the no arbitrage condition, the expected return on an asset is zero. More precisely: $\mu_t := \mathbb{E}[r_t] = 0$ for all t .

Furthermore, returns are impossible to predict and can be very volatile. Hence it is hard for investors to make money without taking risks. The volatility of an asset is used as a measure for risk, because it is “a statistical measure of dispersion of returns” [6]. Otherwise said, when an asset has a high volatility, its value can change dramatically over a small period of time. On the contrary, when an asset has a low volatility, the chances of a dramatical change are far less likely. This means that the value of an asset will not fluctuate that much. Therefore we generally say: the higher the volatility, the riskier the asset.

The advantage of knowing about risks is that we can change our behavior to avoid them. [...] we must take risks to achieve rewards but not all risks are equally rewarded. Both the risk and the rewards are in the future,

so it is the expectation of loss that is balanced against the expectation of reward [4].

We assume that, when balancing the expectation of loss against the expectation of reward, investors are *rational*, *risk averse* and thus have *rational expectations*. The expected reward can be based on a lot of different information. This can be information about a company's earnings / creditors / economic stability, financial press or anything else that may provide insights in future price evolvemments. New information causes investors to reassess asset values and causes changes in asset prices. To illustrate the effect of new information, we will have a look at the Volkswagen stock price between 21st of August and 8th of October 2015 (see Fig. 1).

At the 18th of September you see a sudden drop in the stock price. This drop was due to the scandal known as *the Volkswagen emissions scandal*, which was revealed on the 18th of September.

Remark. Figure 1 shows a dramatic drop in stock price, but not all new information causes such fluctuations. This example is chosen because the impact of new information is clearly visible.



Figure 1: Volkswagen stock price between 21st of August and 8th of October 2015.

In the preceding part, we have described the point of view we take regarding the financial market. We end this section by stating empirical regularities, called *stylized facts*, for which (G)ARCH models are designed [4].

- Returns are almost unpredictable. *(unpredictability)*
- Returns have large numbers of extreme values. *(fat tails)*
- Both the extreme and quiet periods are clustered in time. *(volatility clustering)*

As we keep in mind the financial framework, we now continue introducing the mathematics behind the GARCH model.

3 Mathematical framework

The GARCH model is used for modeling the variance of the error in particular stochastic time series. Besides the mathematical specification of the model, there are fundamental mathematical notions underlying the model. These mathematical notions are mostly probabilistic and measure-theoretic and are presented in the current section.

The mathematics we need in order to fully understand our model can be abstract. To make it more intuitively we will use an analogy throughout this section. The analogy consists of flipping coins on discrete time intervals (you can also think of heads to coincide with the asset value going up and tails with the asset value going down). Heads will be denoted by 'H' and we use 'T' for tails. The first notion we introduce is the notion of a *sigma algebra*.

Definition 3.1. A σ -algebra \mathcal{F} on a sample space Ω is a family of subsets of Ω that satisfy the following conditions:

- (i) $\Omega \in \mathcal{F}$,
- (ii) $A \in \mathcal{F} \implies A^c \in \mathcal{F}$, *(closed under complements)*
- (iii) $(A)_{i \in \mathbb{N}} \subseteq \mathcal{F} \implies \bigcup_{i \in \mathbb{N}} A_i \in \mathcal{F}$. *(closed under countable unions)*

From this point onwards we use the convention that \mathcal{F} denotes a σ -algebra on the sample space Ω . The pair (Ω, \mathcal{F}) is called a *measurable space* and we refer to sets in \mathcal{F} as *events*.

Remark. A σ -algebra always contains the empty set \emptyset . (This follows from 3.1(i) and 3.1(ii).) We define the information set at $t = 0$ to be the empty set: $\mathcal{F}_0 = \emptyset$.

Definition 3.2. Given a measurable space (Ω, \mathcal{F}) , a *filtration* is a sequence of σ -algebras $\{\mathcal{F}_t\}_{t \in \mathbb{N}}$ such that

$$\mathcal{F}_0 \subset \mathcal{F}_1 \subset \dots \subset \mathcal{F}_k \subset \dots \subset \mathcal{F}. \quad (2)$$

All the history and possible events up to time t are encapsulated in the σ -algebra \mathcal{F}_t . We therefore refer to \mathcal{F}_t as *the information set* at time t . In the case of our analogy, at time k we have the following:

- The sample space Ω represents a series of k coin flips:

$$\Omega = \{\omega = (\omega_1, \omega_2, \dots, \omega_k) : \omega_i \in \{H, T\}\}.$$

- The σ -algebra \mathcal{F} contains all possible subsets of Ω .

Now suppose we flip a coin once. This will give us $\mathcal{F}_1 = \{\emptyset, \{H\}, \{T\}, \Omega\}$. Additionally, suppose that we lose 1 EUR if a we flip heads and gain 2 EUR if we flip tails. More specifically, let us define the following function that maps events to possible profits or losses:

$$f : \Omega \rightarrow \{-1, 2\}.$$

If we look at $f^{-1}((-\infty, 0))$, we find the event that coincides with losing money, namely H. Similarly, $f^{-1}((0, \infty))$ coincides with T, the event of making money. This is an example of a *measurable function*.

Definition 3.3. A function $f : \Omega \rightarrow \mathbb{R}$ is *measurable* with respect to the σ -algebra \mathcal{F} [\mathcal{F} -measurable] if the pre-image of every interval is in \mathcal{F} . Precisely if:

$$f^{-1}((a, b)) \equiv \{\omega \in \Omega : f(a) \leq f(\omega) \leq f(b)\} \in \mathcal{F} \text{ for } a, b \in \mathbb{R}, a < b.$$

Note that the probability density function always is a measurable function. For this we define the concept of a *probability measure*.

Definition 3.4. A probability measure \mathcal{P} on a sample space Ω is a measurable function $\mathcal{P} : \Omega \rightarrow [0, 1]$ defined on a σ -algebra \mathcal{F} satisfying

$$(i) \quad \mathcal{P}(\emptyset) = 0 \text{ and } \mathcal{P}(\Omega) = 1 \quad (\textit{normalization})$$

and for any countable family of pairwise disjoint sets $(A)_{i \in \mathbb{N}} \subset \mathcal{F}$:

$$(ii) \mathcal{P}(\bigcup_{i \in \mathbb{N}} A_i) = \sum_{i \in \mathbb{N}} \mathcal{P}(A_i). \quad (\sigma\text{-additivity})$$

Definition 3.5. A *probability space* is a triple $(\Omega, \mathcal{F}, \mathcal{P})$, with \mathcal{P} a probability measure on a sample space Ω with σ -algebra \mathcal{F} .

Definition 3.6. A *random variable* on a probability space $(\Omega, \mathcal{F}, \mathcal{P})$ is function $X : \Omega \rightarrow \mathbb{R}$ that is \mathcal{F} -measurable.

Definition 3.7. A *stochastic process adapted to the filtration* is a family of random variables $\{X_i\}_{i \in \mathbb{N}}$ such that X_n is \mathcal{F}_n -measurable for each $n \in \mathbb{N}$.

For reasons of completeness we give the following definitions.

Definition 3.8. Let $(\Omega, \mathcal{F}, \mathcal{P})$ denote a probability space. The *conditional expectation* of a discrete random variable X given \mathcal{F} is the random variable:

$$\mathbb{E}[X|\mathcal{F}] : \Omega \rightarrow \mathbb{R},$$

defined by:

$$\begin{aligned} \mathbb{E}[X|\mathcal{F}](\omega) &= \mathbb{E}[X|B] && \text{if } \omega \in B \in \mathcal{F} \\ &= \sum_{\alpha \in B} X(\alpha) \mathcal{P}(\{\alpha\}|B). \end{aligned}$$

Definition 3.9 (Markov). A stochastic process $\{X_i\}_{i \in \mathbb{N}}$ adapted to the filtration is *Markovian* if there exists a function g such that:

$$\mathbb{E}[X_t|\mathcal{F}_{t-1}] = g(X_{t-1}).$$

We have now given the underlying theoretical mathematical basis for the GARCH process. Therefore, we are now ready to introduce the GARCH process itself, which we do in the forthcoming section.

4 The GARCH(p, q) process

Following up the last section, we are now ready to introduce the GARCH(p, q) process. This section will define the GARCH(p, q) model and present a more in-depth study on the GARCH(1, 1) model. We will use the convention that $\{\mathcal{F}_t\}_{t \in \mathbb{N}}$ denotes a filtration on the probability space $(\Omega, \mathcal{F}, \mathcal{P})$ as defined in 3.2 and that \mathcal{F}_t indicates the information set at time t .

About three decades ago, conventional time series and econometric models assumed constant variance of the error terms in the models. Yet they found that the variance changed over time. Hence a theory for dynamic volatility was needed. It was in 1982 that Robert Engle introduced the Autoregressive Conditional Heteroscedasticity (ARCH) model. The ARCH model allowed conditional variance to change over time as a function of past squared errors. These errors are innovations in a linear regression:

$$\varepsilon_t = y_t - \mu_t. \quad (3)$$

The ARCH model was then generalized by Bollerslev in 1986, allowing “a longer memory and a more flexible lag structure” [2]. This resulted in the *Generalized Autoregressive Conditional Heteroscedasticity* (GARCH) model, which is formally defined as follows:

Definition 4.1. Let (ε_t) denote a stochastic process adapted to the filtration $\{\mathcal{F}_t\}_{t \in \mathbb{N}}$. The process (ε_t) follows a GARCH(p, q) process if ε_t is conditionally normally distributed with mean zero and variance σ_t^2 . More specifically, for $t \geq \max\{p, q\}$ the process is given by:

$$\varepsilon_t | \mathcal{F}_{t-1} \sim \mathcal{N}(0, \sigma_t^2), \quad (4)$$

$$\sigma_t^2 = \alpha_0 + \sum_{i=1}^q \alpha_i \varepsilon_{t-i}^2 + \sum_{j=1}^p \beta_j \sigma_{t-j}^2, \quad (5)$$

where $p \geq 0$, $q > 0$ and parameter restrictions:

$$\begin{aligned} \alpha_0 &> 0, & \alpha_i &\geq 0, & \beta_j &\geq 0, \\ i &= 1, \dots, q, & j &= 1, \dots, p. \end{aligned}$$

Furthermore we assume that the initial conditions are given by $\varepsilon_0 = 0$ and $\sigma_0 = 1$.

Remark. By assuming the conditional distribution to be Gaussian, we follow the convention of Engle [3]. Nevertheless, it is possible to apply other distributions as well. Notice that the GARCH process is Markov by definition and that in this case (4) is defined by:

$$\begin{aligned} \mathbb{E}[\varepsilon_t | \mathcal{F}_{t-1}] &= \int_{\mathbb{R}} \varepsilon_t \frac{1}{\sqrt{2\pi\sigma_t}} e^{-\frac{\varepsilon_t^2}{2\sigma_t^2}} d\varepsilon_t \\ &= \int_{\mathbb{R}} \varepsilon_t \frac{1}{\sqrt{2\pi(\alpha_0 + \sum_{i=1}^q \alpha_i \varepsilon_{t-i}^2 + \sum_{j=1}^p \beta_j \sigma_{t-j}^2)}} e^{-\frac{1}{2} \frac{\varepsilon_t^2}{(\alpha_0 + \sum_{i=1}^q \alpha_i \varepsilon_{t-i}^2 + \sum_{j=1}^p \beta_j \sigma_{t-j}^2)}} d\varepsilon_t. \end{aligned}$$

The conditional variance specified by a GARCH model is thus a weighted average of past squared errors and conditional variances. The restrictions on the parameters ensure strong positivity of the conditional variance (5). Methods for obtaining and optimizing these weights are discussed in section 5.

In general, the GARCH models are suitable to model financial time series because they properly reflect the stylized facts that characterize these series (section 2). Section 6 describes two empirical examples using a GARCH(1, 1) model. The reason we choose this model is because is a simple but useful model. Intuitively the conditional variance of a GARCH(1, 1) model is a weighted average of the long run variance, the last variance and the new information, measured by the last squared innovation. Let ε_t denote a GARCH process as defined in 4.1. Taking $p = q = 1$ we find that for $t > 1$ the GARCH(1, 1) process is given by (4) and:

$$\sigma_t^2 = \alpha_0 + \alpha_1 \varepsilon_{t-1}^2 + \beta_1 \sigma_{t-1}^2, \quad (6)$$

where $\alpha_0 > 0$, $\alpha_1, \beta_1 \geq 0$ and the initial conditions $\varepsilon_0 = 0$ and $\sigma_0 = 1$.

We proceed by introducing some mathematics based on [2] that allows us to analyze the appropriateness of the GARCH(1, 1) model. Firstly we give the necessary and sufficient conditions for the existence of the second and fourth moment. Respectively, these are given by:

$$\begin{aligned} \alpha_1 + \beta_1 &< 1, \\ 3\alpha_1^2 + 2\alpha_1\beta_1 + \beta_1^2 &< 1. \end{aligned}$$

If these conditions are fulfilled, the second and fourth moment are given by:

$$\begin{aligned} \mathbb{E}(\varepsilon_t^2) &= \frac{\alpha_0}{1 - \alpha_1 - \beta_1}, \\ \mathbb{E}(\varepsilon_t^4) &= \frac{3\alpha_0^2(1 + \alpha_1 + \beta_1)}{(1 - \alpha_1 - \beta_1)(1 - \beta_1^2 - 2\alpha_1\beta_1 - 3\alpha_1^2)}. \end{aligned}$$

Using these moments, we find a measure that can indicate fat tails. This measure for “tailedness” is called *kurtosis* and is given by:

$$\kappa = \frac{\mathbb{E}(\varepsilon_t^4)}{[\mathbb{E}(\varepsilon_t^2)]^2} = \frac{3(1 + \alpha_1 + \beta_1)(1 - \alpha_1 - \beta_1)}{1 - \beta_1^2 - 2\alpha_1\beta_1 - 3\alpha_1^2}. \quad (7)$$

Distributions with a kurtosis greater than 3 indicate fat tails and are called *leptokurtic distributions*.

Besides fat tails, the GARCH model should reflect unpredictability and volatility clustering. These can be shown by looking at *autocorrelations*. Consequently, we define the autocorrelation function, which describes the correlation of a random variable with itself at different points in time.

Definition 4.2. Let $\{X_i\}_{i \in \mathbb{N}}$ be a stochastic process adapted to the filtration $\{\mathcal{F}_t\}_{t \in \mathbb{N}}$ with mean μ_t and variance σ_t^2 . The autocorrelation between time i and j is then given by:

$$\rho(X_i, X_j) = \frac{\mathbb{E}[(X_i - \mu_i)(X_j - \mu_j)]}{\sigma_i \sigma_j}. \quad (8)$$

“Predictability may show up as significant autocorrelations in returns, and volatility clustering will show up as significant autocorrelations in squared or absolute returns. [...] autocorrelations bigger than 0.033 in absolute value would be significant at a 5% level” [4].

Before we continue to the empirical examples, the following section will introduce the basic ideas of machine learning. Some machine learning models allow us to estimate/obtain parameters of a GARCH model. One of these models will be explained in the following section, whereafter the respective model is applied in an empirical example in section 6.

5 Parameter optimization

Optimizing and obtaining parameters of a GARCH model can be done in several ways. One of these ways is through machine learning. In this section we discuss the basic ideas of machine learning and lay the theoretical foundation for the type of machine learning model used in the empirical example in section 6.

Machine learning is a part of *data science*. “Data science is the exploration and quantitative analysis of all available structured and unstructured data to develop understanding, extract knowledge and formulate actionable results” [edX course]¹. Machine learning is the exploration of data and algorithms such that the machine can learn from the data and make predictions on the data. The building of a machine learning model involves [edX course]:

¹I’ve taken a course on machine learning from edX, this taught me the basics of machine learning

1. Finding data sources.
2. Acquiring data.
3. Cleaning and transforming data.
4. Understanding relationships in the data.
5. Delivering value from the data.

The building of a machine learning model is an iterative process. We do not go deeply into this process, though the next section does contain two empirical examples whereby we briefly go through these steps.

When estimating GARCH parameters, we apply a regression model. A regression model is a form of *supervised machine learning* and is used to perform a *predictive analysis*. This form of machine learning requires a dataset of which the true outcomes, called *labels*, are known. The model is calibrated on this known data, after which the model is used to make predictions on new data. More formally: Given dataset D with elements $(x_{i1}, x_{i2}, \dots, x_{ip}, y_i)$ for $i = 1, 2, \dots, n$, where the individual regressions are given by:

$$y_i = \eta_1 x_{i1} + \eta_2 x_{i2} + \dots + \eta_p x_{ip} + \epsilon_i. \quad (9)$$

We find that the regression model for n datapoints is given by the following:

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\eta} + \boldsymbol{\epsilon}, \quad (10)$$

where

$$\mathbf{Y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}, \mathbf{X} = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1p} \\ x_{21} & x_{22} & \cdots & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{np} \end{pmatrix}, \boldsymbol{\eta} = \begin{pmatrix} \eta_1 \\ \eta_2 \\ \vdots \\ \eta_n \end{pmatrix} \text{ and } \boldsymbol{\epsilon} = \begin{pmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \epsilon_n \end{pmatrix}.$$

Remark. Note that the residuals ϵ_k are different from the innovations ε_k defined the previous section.

The residuals are the differences between model's predicted labels \hat{y}_i and the true labels y_i . The key to optimizing a model, is finding (estimated) parameters that minimize the sum of squared residuals (SS_R). This method is also known as

ordinary least squares (OLS) method. The estimator $\hat{\boldsymbol{\eta}}$ that minimizes the SS_R is the *least square estimate* (LSE) of $\boldsymbol{\eta}$. Let the regression model be defined as in (9) and (10). Under the assumption that $\mathbf{X}^T\mathbf{X}$ has full rank and hence is invertible, the least square estimate of $\boldsymbol{\eta}$ is then given by:

$$\hat{\boldsymbol{\eta}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y}. \quad (11)$$

Remark. In the case of a Gaussian distribution, the least squares estimate is identical to the maximum likelihood estimate (MLE).

In machine learning, the LSE is obtained through training our model. By training we mean that the computer uses an algorithm to find the weights that fit the data well. This can be done using OLS. When training a model, we firstly split the data D into two nonempty disjoint sets A and B . The n individual regressions are split up over two sets, resulting in two smaller regression models. Formally, this is done as follows:

Let $\bar{A} \subset \{1, 2, \dots, n\}$ and $\bar{B} = \{1, 2, \dots, n\} \setminus \bar{A}$ with $|\bar{A}| = k$ and $|\bar{B}| = n - k$. Then we define:

$$\begin{aligned} A &= \{(x_{j1}, x_{j2}, \dots, x_{jp}, y_j) : y_j = \eta_1 x_{j1} + \eta_2 x_{j2} + \dots + \eta_p x_{jp} + \epsilon_j \text{ for } j \in \bar{A}\}, \\ B &= D \setminus A = \{(x_{l1}, x_{l2}, \dots, x_{lp}, y_l) : y_l = \eta_1 x_{l1} + \eta_2 x_{l2} + \dots + \eta_p x_{lp} + \epsilon_l \text{ for } l \in \bar{B}\}. \end{aligned}$$

Our initial regression model is now split into two smaller regression models. Similar to (10), we define respectively the regression model with k and $n - k$ datapoints:

$$\begin{aligned} \mathbf{Y}_A &= \mathbf{X}_A \boldsymbol{\eta} + \boldsymbol{\epsilon}_A, \\ \mathbf{Y}_B &= \mathbf{X}_B \boldsymbol{\eta} + \boldsymbol{\epsilon}_B. \end{aligned}$$

Remark. In general we use more data to train the model than to evaluate the model, so we assume $k \geq n - k$. Furthermore, the residuals are generally assumed to be Gaussian with mean 0 and constant variance σ^2 .

Without loss of generalization, we set A to be our training dataset and B as our testing dataset. The parameter weights are determined by training our regression model on A . More specifically, we obtain a LSE by the following:

$$\hat{\boldsymbol{\eta}} = (\mathbf{X}_A^T \mathbf{X}_A)^{-1} \mathbf{X}_A^T \mathbf{Y}_A.$$

Subsequently, the regression model is evaluated against the testing dataset B , resulting in a vector of predicted labels. The predicted labels are computed as follows:

$$\hat{\mathbf{Y}}_B = \mathbf{X}_B \hat{\boldsymbol{\eta}}.$$

We then find the residual vector:

$$\boldsymbol{\epsilon}_B = \mathbf{Y}_B - \hat{\mathbf{Y}}_B,$$

with the individual residuals equal to:

$$\epsilon_l = y_l - \hat{y}_l \text{ for } l \in \bar{B}.$$

These residuals allow us to evaluate our model. If a model is a good fit for the data, then the residuals should be randomly distributed. Furthermore, we can evaluate the model using the following statistics:

- The *mean absolute error* (MAE); a measure of how close the predicted values are to the true labels. It is given by:

$$\text{MAE} = \frac{1}{n-k} \sum_{l \in \bar{B}} |\epsilon_l|. \quad (12)$$

- The *root mean squared error* (RMSE); a measure for the error in the model. It is given by:

$$\text{RSME} = \sqrt{\frac{1}{n-k} \sum_{l \in \bar{B}} \epsilon_l^2}. \quad (13)$$

- The *relative absolute error* (RAE); a measure of the errors in the prediction relative to the deviations of the mean of the true labels.

$$\text{RAE} = \frac{\sum_{l \in \bar{B}} \epsilon_l}{\sum_{l \in \bar{B}} (y_l - \bar{y})}, \quad (14)$$

where $\bar{y} = \frac{1}{n-k} \sum_{l \in \bar{B}} y_l$.

- The *relative squared error* (RSE); a measure of the sum of squared residuals (SS_R) relative to the sum of total squares (SS_T).

$$\text{RSE} = \frac{SS_R}{SS_T}, \quad (15)$$

where $SS_R = \sum_{l \in \bar{B}} \epsilon_l^2$ and $SS_T = \sum_{l \in \bar{B}} (y_l - \bar{y})^2$.

- The *coefficient of determination* (CoD); a measure that indicates how well the model fits the data. In statistics, the CoD is denoted by R^2 and it is defined by:

$$R^2 = 1 - \frac{SS_R}{SS_T} = 1 - \text{RSE}. \quad (16)$$

Remark. The RSE can also be described as the variance of the model relative to the variance of the data.

For all statistics mentioned above, except the CoD, we apply the following rule for evaluation: the lower the better. This implies that the CoD of a model is better when it is closer to 1. A perfect model therefore has a CoD equal to 1 (and thus a RSE of 0). A model with a CoD of 0 indicates that the model is random and does not fit the data even the tiniest bit.

Last but not least, we can train and evaluate a model using *cross validation*. Cross validation is a method in which the data is divided into n subsets called *folds*. The model is then trained on $n - 1$ folds and is evaluated on the remaining fold. This is repeated n times using each fold exactly once as testing fold, allowing us to identify possible disparities in the training and testing datasets. Furthermore, the training and evaluating is done in the same way as described above and thus generates the same set of statistics. Additionally, cross validation reports the mean and standard deviation over the evaluation measures of the n folds. A model will fit the data well if it generates similar measures across all folds with a small range of mean values and low standard deviations.

We now have all tools to build, train and evaluate a model. In the next section we describe an empirical example where we use machine learning to find the parameters of a GARCH(1, 1) model applied to a specific dataset.

6 An empirical example

We now combine the knowledge of all previous sections by describing some empirical examples. We will look at two (or three) different datasets whereon we apply machine learning to find the parameters of a GARCH(1, 1) model such that it fits the data well. The first dataset contains data from a simulated GARCH(1, 1) model and the second dataset contains historical prices of an asset. For each dataset, we will go through the machine learning process, after which we discuss the related results. We

then proceed to the Discussion section, where we summarize and review the work we have done.

Example. 1.1

In this example we look at a dataset acquired through a simulation in Matlab. We have simulated a GARCH(1,1) model with known parameters $\alpha_0 = 0.04$, $\alpha_1 = 0.3$ and $\beta_1 = 0.6$. The relevant code is given in Listing 1.

```
1 Mdl = garch('Constant', 0.04, 'GARCH, 0.6', 'ARCH', 0.3);  
2 [V,Y] = simulate(Mdl, 10000, 'Numpaths', 1);
```

Listing 1: Matlab code.

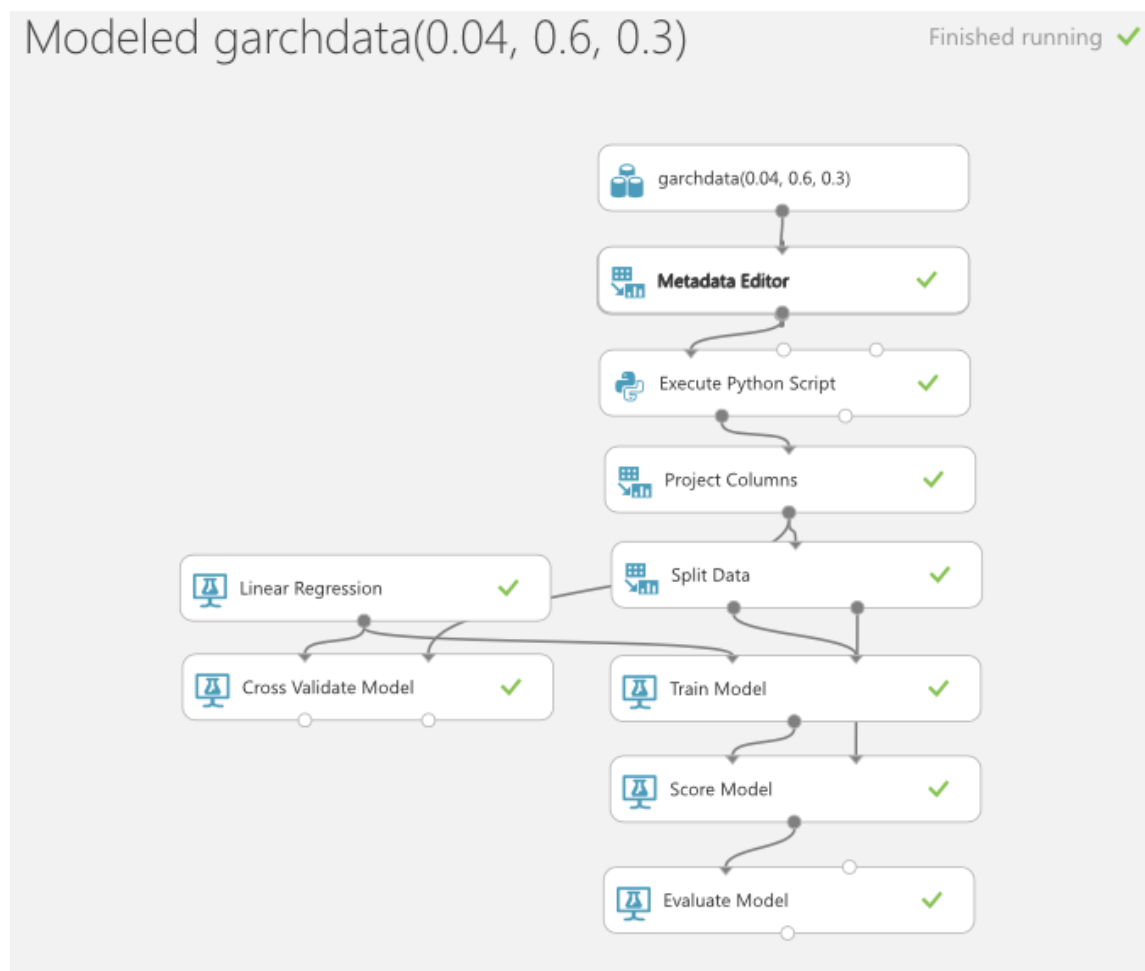


Figure 2: Workflow of example 1.1.

The resulting dataset exists out of an arbitrary amount of data. In this case the dataset contains $n = 10000$ innovations (Y) and variances (V) of a GARCH(1,1) model defined by (4) and (6), where (6) is specified by:

$$\sigma_t^2 = 0.04 + 0.3\varepsilon_{t-1}^2 + 0.6\sigma_{t-1}^2.$$

This generated data will be used in order to check whether we can retrieve the previously mentioned parameter weights of the GARCH(1,1) model through machine learning. The associated workflow is presented in Figure 2. Furthermore, Figure 3 shows the simulated variances σ_t^2 for $t = 1, \dots, 10000$.

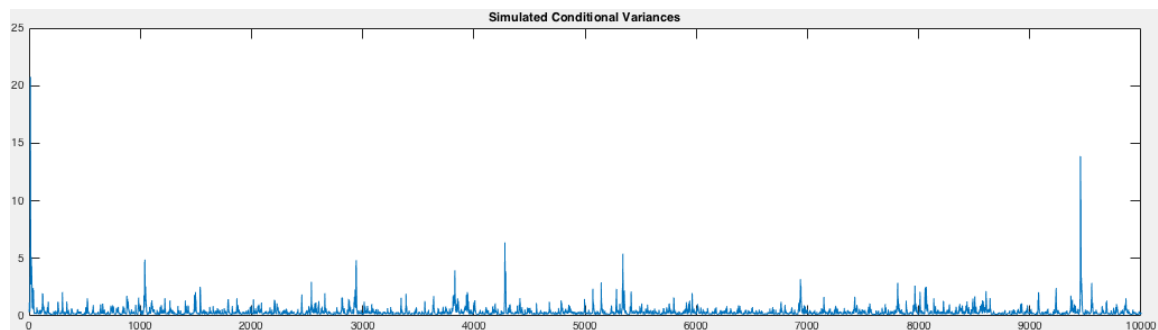


Figure 3: Simulated Variances.

As the dataset consists out of the true innovations and true variances, there is not much cleaning and transforming left to do. However, we do need the squared innovations. Furthermore, when estimating GARCH(1,1) parameters, we look for a linear relationship in the data. Therefore, we select the relevant columns needed for a linear regression. We select the columns containing the squared innovations, the variances and the new variances. Following (10), the regression model is defined by:

$$\begin{pmatrix} \sigma_1^2 \\ \sigma_2^2 \\ \vdots \\ \sigma_n^2 \end{pmatrix} = \begin{pmatrix} 1 & \varepsilon_0^2 & \sigma_0^2 \\ 1 & \varepsilon_1^2 & \sigma_1^2 \\ \vdots & \vdots & \\ 1 & \varepsilon_{n-1}^2 & \sigma_{n-1}^2 \end{pmatrix} \begin{pmatrix} \alpha_0 \\ \alpha_1 \\ \beta_1 \end{pmatrix} + \begin{pmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \epsilon_n \end{pmatrix}. \quad (17)$$

The data is split into two equally sized sets, the training dataset and the testing dataset, and is trained using a regression model specified by (17). We obtain an estimate for $(\alpha_0, \alpha_1, \beta_1)^T$ by using OLS (see section 5). The results of the training are depicted in Table 1 and the predicted labels can be found in the appendix in Figure 6. Notice that the estimated parameters are approximately the same as the

true parameters and that the scored labels are very close to the new variance. In other words, our machine learning model seems to perfectly fit the data. Besides the fact that the estimated parameter weights are significantly close to the real parameter weights, we can strengthen the idea of a good fit by looking at the evaluation measures in Table 2 and in the cross validation results in Table 3. Table 2 shows low evaluation measures and a CoD near to 1. When taking a look at Table 3, we find similar measures across all folds, with a small range of mean values and low standard deviations. Furthermore 9 out of 10 folds have a CoD of 1, which implies a perfect fit. We hence conclude that our model is a good fit for the data and that a machine learning model is able to find the MLE if a theoretical optimum is present.

Example. 1.2

In the previous example, the dataset featured the innovations as well as the variances and we knew that the process followed a GARCH(1, 1) with known parameters. In general however, even if we suspect that a process is GARCH, we do not know the true variances. Therefore, they have to be estimated. Example 2 will apply machine learning at a real financial dataset in order to find parameters for a GARCH(1, 1) model such that it fits the data well. Before looking at this example however, we want to know whether a machine learning model is able to find parameters that fit the data well when we know for sure that it is GARCH but that the variances are unknown. We take the same squared innovations as in example 1.1 and as we assume that the innovations are conditionally normally distributed (4), we estimate the variances by taking the unbiased sample variance:

$$\hat{\sigma}_t^2 = \frac{1}{N-1} \sum_{i=0}^{N-1} \varepsilon_{t-i}^2. \tag{18}$$

By estimating the variance, the σ_t^2 's become different than those in example 1.1. This in turn changes the initial weights belonging to the generated data. Using trial and error, we took a sample size of $N = 100$. The training and evaluation results can be found respectively in Table 4 and 5. The model has small errors and a CoD of 0.9964, implying a near to perfect fit. We thus conclude that even when the variances are unknown and have to be estimated, machine learning is able to find parameters such that it fits the data well.

	True parameters	Estimated parameters
α_0	0.04	0.039999
α_1	0.3	0.299999
β_1	0.6	0.600004
Moments and Kurtosis		
$\alpha_1 + \beta_1$	0.9	0.900003
$3\alpha_1^2 + 2\alpha_1\beta_1 + \beta_1^2$	0.99	0.990004
κ	57	57.022329

Table 1: Training results ex. 1.1.

Evaluation results	
MAE	0.000106
RMSE	0.006391
RAE	0.0005
RSE	0.000137
CoD	0.999863

Table 2: Evaluation results ex. 1.1.

Fold number	MAE	RMSE	RAE	RSE	CoD
0	0.000048	0.000056	0.000219	0	1
1	0.000046	0.00005	0.000223	0	1
2	0.00041	0.012781	0.002042	0.001501	0.998499
3	0.000047	0.000054	0.000191	0	1
4	0.000048	0.00006	0.000239	0	1
5	0.000047	0.000057	0.000197	0	1
6	0.000049	0.000082	0.000198	0	1
7	0.000047	0.000053	0.000206	0	1
8	0.000047	0.000051	0.000225	0	1
9	0.000047	0.000053	0.000239	0	1
Mean	0.000084	0.00133	0.000398	0.00015	0.99985
Standard Deviation	0.000115	0.004024	0.000578	0.000475	0.000475

Table 3: Cross validation evaluation results ex. 1.1.

Estimated parameters for N = 100	
α_0	0.002763
α_1	0.002625
β_1	0.988881
Moments and Kurtosis	
$\alpha_1 + \beta_1$	0.991506
$3\alpha_1^2 + 2\alpha_1\beta_1 + \beta_1^2$	0.983098
κ	3.002446

Table 4: Training results ex. 1.2.

Evaluation results for N = 100	
MAE	0.005175
RMSE	0.013079
RAE	0.037819
RSE	0.003589
CoD	0.996411

Table 5: Evaluation results ex. 1.2.

In the previous examples we looked at a dataset of which we knew that it followed a GARCH model. In the first example, the dataset featured the true innovations as well as the true variances. The second example featured the true innovations and used estimated variances. The first model retrieved the initial parameters almost flawlessly and the second model found other parameters that were able to fit the data well. Both models were concluded to be a good fit for the data.

Now that we have concluded that machine learning is able to retrieve the true parameters of a GARCH(1,1) model with known parameters and that it can fit the data well even when the variances have to be estimated, we next concentrate on an example concerning a real financial dataset. The following dataset will require estimates for the variance and more cleaning and transforming of the data. Additional figures associated to this example can be found in the Appendix and they will be referred to when needed.

aex data

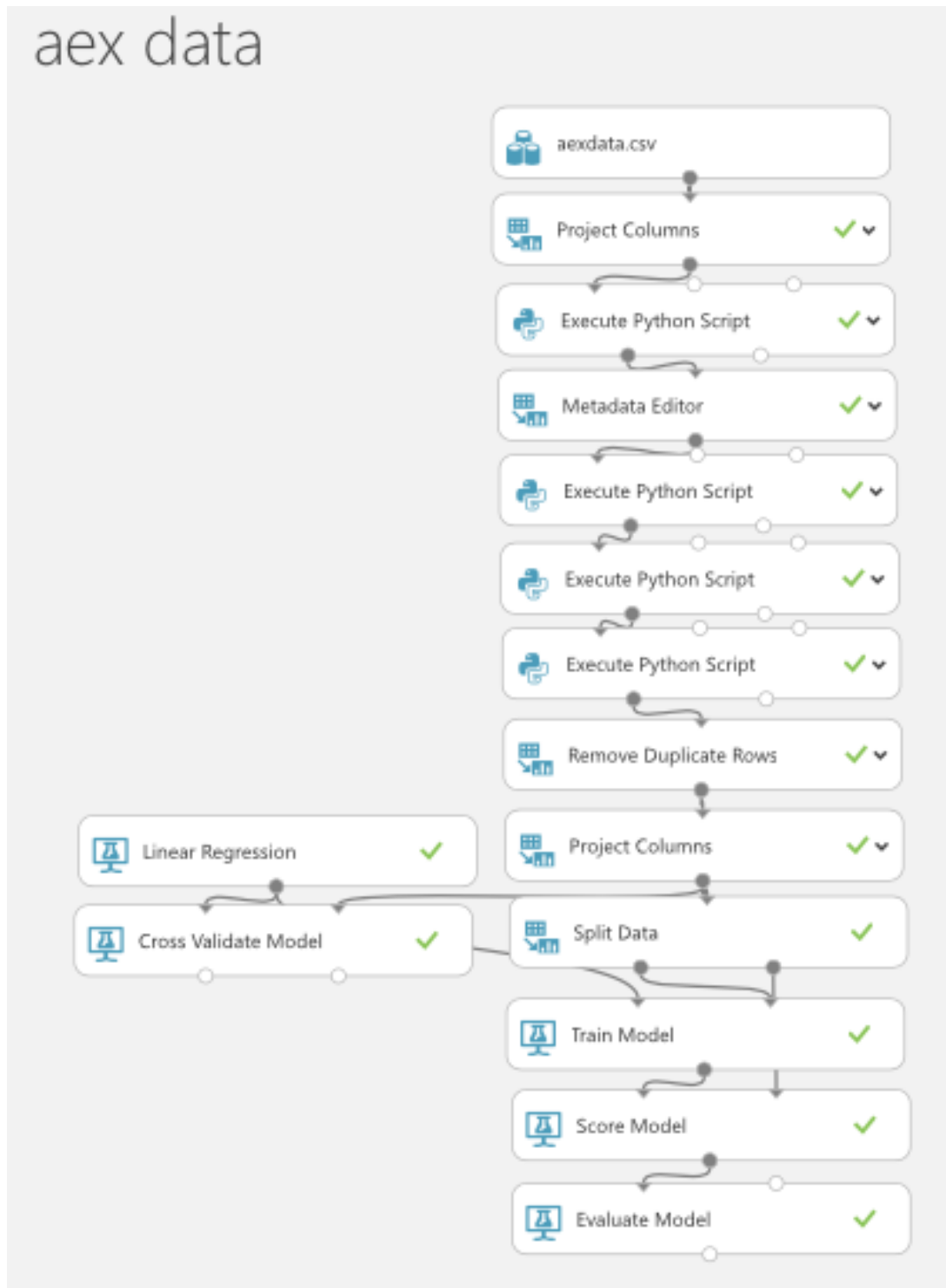


Figure 4: Workflow of example 2.

Example. 2

In this example we look at the daily returns on the closing values of the AEX. We will apply machine learning to try to find parameters for the GARCH(1,1) model such that it fits the data well. The workflow for this example is presented in Figure 4.

The dataset we use contains the daily data of the AEX between 1992 and 2015 and consists out of the following features: Date, Open, High, Low, Close, Volume, Adj. Close. Of these features, we only need the 'Close' values because they allow us to calculate the daily returns (1). Using (3) and the fact that returns are almost unpredictable (see section 2), we find that:

$$\varepsilon_t = r_t.$$

In other words, we find that the innovations are equal to the returns and hence the squared innovations are equal to the squared returns. Furthermore, as the variances are unknown, we will need to estimate them from our data. Remember that the innovations are assumed to be conditionally normally distributed (4). Therefore, we estimate the variances by taking the unbiased sample variance as defined in (18) with a sample size of $N = 25$. The size of the sample is chosen by trial and error.

After cleaning and transforming the data (see Listing 2), we end up with a dataset that contains the squared innovations, the estimated variances and the new variances. Analogously to example 1.1, a regression model equivalent to (17) is applied to this data. The training and evaluation results are depicted respectively in Table 6 and 7. Notice that the RAE is quite high, which implies that the errors in the prediction are relatively large. This can also be seen when comparing the new variances to the associated predicted labels in Figure 7. Despite these differences however, we have a CoD of approximately 0.91, implying a pretty good fit. About 90% of the data can be explained by our model. Taking a closer look at the cross evaluation results (see Table 8) reveals more about where these errors came from. Namely from fold 5 and 6.² We see that most folds have more or less the same evaluation results, but that those in fold 5 and 6 are significantly different. They have far higher error values than the other folds, some even 2-9 times higher, and a CoD of respectively 0.23 and 0.06. The reason these are so significantly different, could originate from the economic circumstances belonging to the period these folds cover. When we for instance take a closer look at our initial data (see Fig. 5), we see two rather big drops between "Jan 1 '00" and "Jan 1 '10". It is possible that the period around these drops were covered by fold 5 and 6 and that they were so substantially different that our model

²Remember that the training and evaluation datasets were equal of size. Hence folds 0-4 were used for training and that 5-9 were used for evaluation.



Figure 5: AEX Closing values 1992 and 2015.

could not yet explain this particular data. In turn, this would then cause higher error statistics. However, in order to better understand the origin of the errors, the data and the results should be examined more carefully. Unfortunately, this is outside the scope of this thesis. Further explanations for the model to not completely be able to explain the data include erroneous estimated variances and/or an underlying probability distribution that is non-Gaussian. Feasible ways to improve our model are discussed in the following section.

Summarizing the previous example, we created a regression model for data of a GARCH(1,1) model with unknown parameters. The estimated parameters were able to explain approximately 90% of the data and the model is therefore considered to be a reasonable good fit. The remaining, unexplained 10% is possibly due to economic circumstances, but this should be examined more carefully. Furthermore it is also possible that our model makes the wrong assumptions concerning the variances or the underlying probability distribution and therefore is not able to explain the data better than it already does. We will now continue to the discussion, where we summarize and review the work we have done in this thesis and suggest possible future work.

Estimated parameters for N = 25	
α_0	0.000042
α_1	0.004498
β_1	0.894212
Moments and Kurtosis	
$\alpha_1 + \beta_1$	0.89871
$3\alpha_1^2 + 2\alpha_1\beta_1 + \beta_1^2$	0.807720
κ	3.000631

Table 6: Training results ex. 2.

Evaluation results	
MAE	0.000037
RMSE	0.000098
RAE	0.234893
RSE	0.088352
CoD	0.911648

Table 7: Evaluation results ex. 2.

Fold number	MAE	RMSE	RAE	RSE	CoD
0	0.000042	0.000197	0.124095	0.013007	0.986993
1	0.000042	0.000194	0.111238	0.012674	0.987326
2	0.00006	0.000331	0.129891	0.027524	0.972476
3	0.000036	0.000155	0.118837	0.011788	0.988212
4	0.000031	0.000111	0.143316	0.01186	0.01186
5	0.000065	0.000911	0.272315	0.765646	0.234354
6	0.000064	0.000976	0.312885	0.937289	0.062711
7	0.000056	0.000271	0.111049	0.015018	0.984982
8	0.000039	0.00016	0.122133	0.012437	0.987563
9	0.000043	0.000201	0.115911	0.013487	0.986513
Mean	0.000048	0.000351	0.156167	0.182073	0.817927
Standard Deviation	0.000012	0.000319	0.073153	0.355144	0.355144

Table 8: Cross validation evaluation results ex. 2.

7 Discussion

In this thesis we have presented the theoretical basis of the GARCH and machine learning (linear) regression model, whereafter we applied machine learning to estimate the parameters of a GARCH(1,1) model. The standard procedure for estimating these parameters however, is the maximum likelihood (ML) procedure. The estimates are ML estimates if the underlying probability density function (pdf) is Gaussian and are called quasi-ML if this is not the case. The pdf is generally assumed to be Gaussian, which we did as well, but assuming the wrong pdf can cause errors. Furthermore, we have seen that when a random variable exactly follows a GARCH(1,1) process as defined by (4) and (6), a machine learning model is able to find the ML estimator. So when there is a theoretical optimum, a machine learning model can find it (example 1.1). We also found that when the true variance is unknown and has to be estimated, a machine learning model is still able to find estimated parameters such that it fits the data well (example 1.2).

The problem with most data however, is that the underlying pdf is not always known and therefore a theoretical optimum cannot be found using (quasi) ML procedures. As we have assumed a Gaussian pdf throughout this thesis, we might be able to find a better estimator in example 2 by assuming a different pdf. Other possible improvements can possibly be accomplished by tweaking some parameters (e.g. the number of samples used for estimating the variance) or by applying a different machine learning model. The linear model that we used is just one out of several. To give an example of an alternative model, there are machine learning models that use so-called Support Vector Machines (SVMs) and “The benefits of the SVM in regression (also known as a support vector regressor; SVR) lies in not assuming that there is a probability density function (pdf) over the return series [...]” [5]. Therefore, these models can give better estimates when the underlying pdf is unknown or not Gaussian [5]. Future work could include applying one of these models, more tweaking of parameters and/or doing a more in-depth study on the data, possibly leading to a better understanding of the data. This in turn could cause a practitioner to make correcter assumptions on the model, which could eventually lead to better results.

Appendix

```
1 def azureml_main(frame1):
2     import pandas as pd
3     import os.path
4     from math import log
5
6     # what asset will we be looking at?
7     # construct dataframe from txt-file (from yahoo finance)
8     asset = 'aex'
9     pathName = "/Users/mathijsdelepper/Documents/MATLAB"
10    fileName = asset+".txt"
11    filePath = os.path.join(pathName, fileName)
12    frame1 = pd.read_csv(filePath)
13
14    # all we need are the closing values
15    # data from yahoo finance is listed (top-bottem) from recent
16    # to less recent. Therefore we reverse the list.
17    frame1 = frame1['Close']
18    lijstje = frame1.tolist()
19    lijstje.reverse()
20    frame1 = pd.DataFrame(lijstje)
21
22    # create dataframe with returns
23    frame1['Returns'] = [1.0]*len(frame1)
24
25    for x in range(len(frame1)-1):
26        Return = frame1[0][x+1]/frame1[0][x]
27        logreturn = log(Return)
28        frame1['Returns'][x] = logreturn
29
30    # create dataframe with squared returns
31    frame1['Squared>Returns'] = frame1['Returns']**2
32
33    # create dataframe with (new) variance where the variance is
34    # a sample variance (from a normal distribution) from a
35    # sample with 25 elements.
36    frame1['Variance'] = [0.0]*len(frame1)
37    frame1['New_Variance'] = frame1['Variance']
38
39    N = 25
40    i = 0
41    for x in range(len(frame1) - 1):
42        if(x>N):
43            while(i < N):
44                frame1['Variance'][x] += frame1['Squared>Returns'][x-i]
```

```

45         i += 1
46         frame1['Variance'][x] = frame1['Variance'][x]/(N-1)
47         i = 0
48
49
50     for x in range(len(frame1) - 2):
51         frame1['New_Variance'][x] = frame1['Variance'][x+1]
52
53     # create .csv files for matlab / azure ml
54     returnlist = frame1['Returns']
55     squaredreturnlist = frame1['Squared>Returns']
56     variancelist = frame1['Variance']
57     new_variancelist = frame1['New_Variance']
58
59     everything = [squaredreturnlist, variancelist, new_variancelist]
60     dataframe_full = pd.concat(everything, axis = 1)
61
62     dataframe_full = dataframe_full.drop(dataframe_full.index[:(N+1)])
63
64     returnlist.to_csv('returnlist_'+asset+'.csv', index = False, header
65     = True)
66     squaredreturnlist.to_csv('squaredreturnlist_'+asset+'.csv', index =
67     False, header = True)
68     variancelist.to_csv('variancelist_'+asset+'.csv', index = False,
69     header = True)
70     new_variancelist.to_csv('new_variancelist_'+asset+'.csv', index =
71     False, header = True)
72     dataframe_full.to_csv(asset+'_info_from_python.csv', index = False,
73     header = True)
74
75     return frame1

```

Listing 2: Joined Execute Python Scripts Example 2.


	Innovation_sqrd	Variance	New_variance	Scored Labels
view as  				
	0.018722	0.36036	0.26183	0.261833
	1.060076	0.30487	0.54092	0.540944
	0.81355	0.37946	0.51174	0.511741
	0.327287	0.30663	0.32217	0.322164
	0.159121	0.19749	0.20623	0.20623
	0.185813	0.18998	0.20973	0.209731
	0.002769	0.18469	0.15165	0.151644
	0.123173	0.32351	0.27106	0.271058
	0.302126	0.41311	0.3785	0.378504
	0.001756	0.19979	0.1604	0.160401
	0.164601	0.25197	0.24056	0.240562
	0.257171	0.23985	0.26106	0.261061
	0.07131	0.13783	0.14409	0.144091

Figure 6: Part of the predicted labels ex. 1.1.







	Squared_Returns	Variance	New_Variance	Scored Labels
view as  				
	0.000055	0.000147	0.000149	0.000173
	0.000096	0.000149	0.000149	0.000176
	0.000001	0.000149	0.000147	0.000175
	0.000008	0.000147	0.000129	0.000173
	0.000001	0.000129	0.000129	0.000158
	0.000005	0.000129	0.000108	0.000158
	0.000144	0.000108	0.000106	0.000139
	0.000055	0.000106	0.000089	0.000137
	0.000046	0.000089	0.000081	0.000122
	0.000002	0.000081	0.000077	0.000114
	0.000006	0.000077	0.000069	0.000111
	0.000002	0.000069	0.000071	0.000104
	0.000155	0.000071	0.000056	0.000106

Figure 7: Part of the predicted labels ex. 2.

References

- [1] J. van Bergen (2015), “Efficient Market Hypothesis: Is the Stock Market Efficient?”, available at <http://www.investopedia.com/articles/basics/04/022004.asp>
- [2] T. Bollerslev (1986), “Generalized autoregressive conditional heteroskedasticity” *Journal of Econometrics* 31, 307-327
- [3] R. Engle (1982), “Autoregressive Conditional Heteroskedasticity with Estimates of the Variance of U.K. Inflation”, *Econometrica*, 41, 987-1008.
- [4] R. Engle (2003), “Risk and Volatility: Econometric Models and Financial Practice”, *Nobel Lecture*
- [5] F. Pérez-Cruz, J. A Afonso-Rodríguez and J. Giner (2003, “Estimating GARCH models using support vector machines”, *Quantative Finance Volume 3*, 1-10
- [6] Unknown (2003), “Volatility”, available at <http://www.investopedia.com/terms/v/volatility.asp>