

Bachelor's Thesis



**Czech
Technical
University
in Prague**

F3

**Faculty of Electrical Engineering
Department of Cybernetics**

Portfolio Optimization in Cryptocurrency Markets

Ondřej Komín

**Supervisor: Ing. Matej Uhrín
January 2022**

I. Personal and study details

Student's name: **Komín Ondřej** Personal ID number: **483762**
Faculty / Institute: **Faculty of Electrical Engineering**
Department / Institute: **Department of Cybernetics**
Study program: **Open Informatics**
Specialisation: **Artificial Intelligence and Computer Science**

II. Bachelor's thesis details

Bachelor's thesis title in English:

Portfolio Optimization in Cryptocurrency Markets

Bachelor's thesis title in Czech:

Optimalizace kryptoměnového portfolia

Guidelines:

1. Obtain relevant data available from multiple cryptocurrency exchanges [1,2].
2. Propose predictive models to identify mispriced assets within the data as a proof of concept. Predictive models should also estimate the uncertainty of the predictions.
3. Review existing approaches to portfolio optimization and implement the approach based on the Kelly criterion [3, 4]. Discuss the specifics of applying the Kelly criterion to optimize the portfolio of cryptocurrencies (e.g., the transaction costs of rebalancing the portfolio, higher uncertainty of price estimates, etc.)
4. Propose a robust backtesting approach to evaluate and compare the implemented combination of proposed predictive models and the Kelly strategy using the obtained data.
5. Discuss your findings and possible limitations of your proposed backtesting approach (e.g., the potential market reaction should the chosen solution be applied).

* Portfolio is a set of fractions that usually sum up to 1.0, representing 100% of investor's wealth.

Bibliography / sources:

[1] <https://docs.pro.coinbase.com/#get-historic-rates>

[2]

<https://github.com/binance/binance-spot-api-docs/blob/master/rest-api.md#klinecandlestick-dat>

[3] MacLean, Leonard C., Edward O. Thorp, and William T. Ziemba. The Kelly capital growth investment criterion: Theory and practice. Vol. 3. world scientific, 2011.

[4] Uhrín, Matej, et al. "Optimal sports betting strategies in practice: an experimental review." IMA Journal of Management Mathematics (2021).

Name and workplace of bachelor's thesis supervisor:

Ing. Matej Uhrín, Department of Computer Science, FEE

Name and workplace of second bachelor's thesis supervisor or consultant:

Date of bachelor's thesis assignment: **10.09.2021** Deadline for bachelor thesis submission: **04.01.2022**

Assignment valid until: **19.02.2023**

Ing. Matej Uhrín
Supervisor's signature

prof. Ing. Tomáš Svoboda, Ph.D.
Head of department's signature

prof. Mgr. Petr Páta, Ph.D.
Dean's signature

III. Assignment receipt

The student acknowledges that the bachelor's thesis is an individual work. The student must produce his thesis without the assistance of others, with the exception of provided consultations. Within the bachelor's thesis, the author must state the names of consultants and include a list of references.

Date of assignment receipt

Student's signature

Acknowledgements

I would like to thank my supervisor Ing. Matej Uhrín, for his patient guidance and constructive consultations that helped me greatly with this thesis. I would also like to thank my family and friends for their continuous support during my studies.

Declaration

I declare that the presented work was developed independently and that I have listed all sources of information used within it in accordance with the Methodical instructions for observing the ethical principles in the preparation of university theses.

Prague, January 4, 2022

Abstract

The aim of this work is to analyze the development of the price of cryptocurrencies and to propose models for price prediction, which will also be able to express the estimation uncertainty. In analyzing the problem, we showed the similarities between the random walk process and the development of the price of Bitcoin. This suggests that this is a challenging problem, as a random walk is unpredictable in principle. Overall, the most successful model tested proved to be a recurrent neural network with LSTM cells. We then examined this model more closely and found that the training dataset was able to detect large changes in price to some extent. However, we did not observe the ability to recognize large price fluctuations on the validation or test dataset. We also introduced the Kelly criterion, which is used to determine the ideal size of the trade, and tested how some of our models would perform when applying this criterion.

Keywords: portfolio optimization, cryptocurrencies, predictive models, LSTM, time series, Kelly criterion

Supervisor: Ing. Matej Uhrín

Abstrakt

Cílem této práce je analyzovat vývoj hodnoty kryptoměn a navrhnout modely k predikci ceny, které zároveň budou umět vyjádřit nejistotu odhadu. Při analýze problému jsme ukázali na podobnosti procesu náhodné procházky a vývoje ceny Bitcoinu. To naznačuje, že jde o náročný problém, neboť náhodná procházka je z principu nepředvídatelná. Celkově nejúspěšnějším vyzkoušeným modelem se ukázala být rekurentní neuronová síť s LSTM buňkami. Tento model jsme pak blíže zkoumali a zjistili jsme, že se na trénovacím datasetu dokázal do určité míry naučit rozpoznat velké změny v ceně. Schopnost rozpoznávat velké výkyvy ceny na validačním či testovacím datasetu jsme však nepozorovali. Dále jsme uvedli Kellyho kritérium, které slouží k určení ideální velikosti obchodu a otestovali jsme, jak by si vedly některé naše modely při aplikaci tohoto kritéria.

Klíčová slova: optimalizace portfolia, kryptoměny, prediktivní modely, LSTM, časové řady, Kellyho kritérium

Překlad názvu: Optimalizace kryptoměnového portfolia

Contents

1 Cryptocurrency markets	1	4.4.4 Dropout	20
1.1 Cryptocurrencies	1	4.4.5 Batch normalization	20
1.2 Evolution of cryptocurrency markets	1	4.4.6 Data transformation	20
1.3 Market dynamics	2	4.4.7 Feature selection	20
1.3.1 Price drivers	2	4.4.8 Uncertainty estimate	21
1.3.2 Cryptocurrencies connectedness	2	4.4.9 NN architecture	22
1.3.3 Risk	3	4.5 Results	22
1.3.4 Heavy tails	3	5 Portfolio optimization	29
1.3.5 Market efficiency	4	5.1 Kelly criterion	29
2 Problem description	5	5.2 Review of other portfolio optimization methods	30
2.1 Time series	5	5.2.1 Mean-variance portfolio	30
2.2 Stochastic process	5	5.2.2 Risk parity portfolio	31
2.3 Means of describing time series	6	5.3 Portfolio optimization and backtesting strategy based on Kelly criterion	31
2.4 Stationarity	6	5.4 Results of Kelly criterion optimization	32
2.4.1 Strict sense stationarity	6	5.4.1 Limitations of backtesting	33
2.4.2 Weak sense stationarity	6	6 Conclusion	35
2.5 Random walk	7	6.1 Future work	35
2.6 Problem description	7	7 Acronyms	37
2.7 Uncertainty	8	Bibliography	39
2.8 Simple returns	8	A Figures	43
2.9 Metrics	8		
2.9.1 Root mean squared error (RMSE)	8		
2.9.2 Mean average percentage error (MAPE)	9		
2.9.3 Mean directional accuracy (MDA)	9		
3 Dataset	11		
3.1 Data	11		
3.2 Filling the missing data	12		
4 Models	13		
4.1 Naive model	13		
4.2 Naive directional model	13		
4.3 ARIMA	13		
4.3.1 ACF	14		
4.3.2 PACF	14		
4.3.3 AR process	15		
4.3.4 MA process	15		
4.3.5 ARIMA(p, d, q) model	15		
4.3.6 Model characteristics	16		
4.4 LSTM	17		
4.4.1 Internal structure	18		
4.4.2 Early stopping	19		
4.4.3 L1 and L2 regularization	19		

Figures

1.1 Histogram of 1h BTC returns, estimate of actual probability density function (KDE) and comparison with Normal distribution	4
2.1 Scaled BTC price evolution compared to Gaussian random walks with $\mu = 0$ and $\sigma = 1$	7
3.1 Dataset split	12
4.1 ACF, PACF plot of 500 data points	16
4.2 ACF, PACF plot of 80 data points	17
4.3 RNN and its unrolled form, picture taken from [19]	18
4.4 Internal structure of LSTM cell, picture taken from [19]	18
4.5 Visual comparison of models predictions on the part of the test dataset	26
4.6 Analysis of predictions for different LSTM models on the part of the training dataset. Note that LSTM predicts small values, so the predictions are multiplied by 10. . .	27
5.1 Results of backtested strategies [1ex] LSTM-2L: 2 layer LSTM model, LSTM-1L MC: 1 layer LSTM model with MC sampling, fees are 0.1% .	32
A.1 Architecture of 1 layer LSTM model	43
A.2 Architecture of 2 layer LSTM model	44
A.3 Architecture of 3 layer LSTM model	45
A.4 Analysis of predictions for different LSTM models on the part of the validation dataset. Note that LSTM predicts small values, so the predictions are multiplied by 10. . .	46
A.5 Analysis of predictions for different LSTM models on the part of the test dataset. Note that LSTM predicts small values, so the predictions are multiplied by 10. . .	47

Tables

1.1 Comparison of tail probabilities for KDE of BTC 1h returns and Normal distribution with sample mean and variance. $P(R_t < -0.1)$ denotes the probability that the value of BTC drops more than 10%. [1ex] * real value is small number bigger than zero, the computed value is zero because of imprecision in calculation of floating point numbers	4
4.1 Results on training, validation and test datasets for tested models	25

Chapter 1

Cryptocurrency markets

This chapter will introduce cryptocurrencies, cryptocurrency markets, their behaviour, main characteristics, and interconnectedness with traditional asset classes.

1.1 Cryptocurrencies

Cryptocurrency is a digital asset stored on a public distributed ledger. It is typically not subject to a central authority and has rules that ensure overall functionality and enable consensus in the network.

The first cryptocurrency was created in 2009 by a group of people or a person with the pseudonym Satoshi Nakamoto to challenge the traditional financial sector and remove the need to trust banks and other institutions involved. Since the creation of Bitcoin, many other cryptocurrencies were introduced, and to this day, more than 5000 cryptocurrencies exist (coinmarketcap.com), according to coingecko.com, it is more than 7000.

1.2 Evolution of cryptocurrency markets

When Bitcoin was created, the possibilities of how to trade this cryptocurrency were sparse. The earliest exchanges happened on forums, but a great amount of trust from both sides was needed to close the deal successfully. The first cryptocurrency exchange launched in 2010, soon followed by other exchanges.

Total cryptocurrency market capitalization experienced two substantial increases and falls, the first in 2013, followed by a decrease after the biggest exchange of that time crashed, and second in 2017, peaking above 800B USD. The third upsurge is currently (December 2021) taking place, with market capitalization peaking above 2.9T USD.

1.3 Market dynamics

Most of the literature concerning market behaviour studies only Bitcoin or is concerned with a selection of the few most successful cryptocurrencies. However, we think that current literature may serve the purpose of illustrating the behaviour of cryptocurrencies and differentiating them from other asset classes.

1.3.1 Price drivers

[1] studies the main drivers of Bitcoin prices and concludes that the most significant driver in both the long-term and short-term is the interest, as analyzed from google and wiki trends. He also notes that standard fundamental factors such as usage, money supply and price level play a role in Bitcoin (BTC) price in the long run.

1.3.2 Cryptocurrencies connectedness

Co-movements between cryptocurrencies has been examined by [2] and [3]. The first paper analyses the interrelationship between Bitcoin and five major cryptocurrencies. His results are not the same for all cryptocurrency pairs. Covariance between one pair occurs more, for three pairs occur to a lesser extent, and between the last pair practically does not occur.

The second author suggests that interdependence between BTC and altcoin markets is more apparent in the short-term than in the long-term and observes a limited impact of BTC on altcoin prices in the long run. Somewhat on the contrary to the first author, [3] argues that Bitcoin and altcoin markets are still highly interdependent.

[4] and [5] investigated the contagion effect, another interconnectedness characteristics of cryptocurrencies. Contagion effect occurs when there is a big change in the price of one asset, and it is propagated to other assets. Such effect occurred during 2007-2008 financial crisis when the whole stock market “collapsed”.

Both authors compared periods before the 2017 crash and after the crash. Integration in both tested periods was confirmed, with higher integration after the crash. As a result of tighter integration, the former author suggests possible negative influence on diversification possibilities.

Although the authors disagree precisely when the transition to a more integrated market happened, both say that cryptocurrencies were less integrated before the crash. [5] points out that periods of high uncertainty in the market correspond to the strong interconnectedness of cryptocurrencies and vice versa. BTC is noted to be the main shock transmitter in the pre-crash period, whereas Ethereum (ETH) becomes the main transmitter in the post-crash period, indicating an increase in the importance of ETH.

1.3.3 Risk

Cryptocurrencies returns (defined in Section 2.8) generally suffer from heavy tails, as [6], [7], [8] observed. When compared with traditional currencies, cryptocurrencies are much riskier and volatile, but they also offer higher potential profits. We can also observe clustering of extreme events in Bitcoin, which holds for both negative and positive returns.

1.3.4 Heavy tails

In heavy-tailed distribution, events far from mean are much more likely to happen than in non-heavy tails distribution. Whether a distribution is heavy-tailed is typically shown by comparison with a normal distribution. For example, [9] describes heavy tails simply as “*the probability of extreme profits or losses is much larger than predicted by the normal distribution. Tail thickness varies from asset to asset*”. [10] defines upper tail as $\bar{F}(x) = P(X > x) = 1 - F(x)$ and adds that we are mainly interested in the behaviour of upper tail for large x . F is cumulative distribution function of given distribution. Describing behaviour with the upper tail is the convention in literature, but behaviour could be similarly described with lower tail.

For demonstration, we compare tail probabilities of Kernel Density Estimate (KDE) for 1h BTC returns with normal distribution with sample mean and variance in Figure 1.1. Probabilities for different selected events are shown in Table 1.1.

KDE – Kernel Density Estimate

KDE is a nonparametric way of estimating probability density of random variable from set of measurements/samples, in our case BTC 1h returns. Value of probability density function at each point is given as

$$\hat{p}(x_j) = \frac{1}{N} \left(\sum_{i=1}^N K_h^G(x_j, x_i) \right)$$

where K_h^G is Gaussian kernel defined as

$$K_h^G(x_j, x_i) = \frac{1}{h\sqrt{2\pi}} e^{-\frac{1}{2} \left(\frac{x_j - x_i}{h} \right)^2}$$

x_j is point being evaluated, x_i is one of the observed points and h is kernel’s bandwidth parameter. For selection of bandwidth we used scipy’s default method which implements Scott’s rule of thumb, [11].

$$h = n^{-\left(\frac{1}{d+4}\right)}$$

n – number of data points

d – number of dimensions.

Probabilities	KDE	Normal distribution
$P(R_t < -0.1)$	0.000131	3.614907e-26
$P(R_t < -0.05)$	0.002254	7.041860e-08
$P(R_t < -0.03)$	0.009181	0.000781
$P(R_t > 0.03)$	0.008128	0.000845
$P(R_t > 0.5)$	0.001891	7.980570e-08
$P(R_t > 0.1)$	0.000238	0*

Table 1.1: Comparison of tail probabilities for KDE of BTC 1h returns and Normal distribution with sample mean and variance. $P(R_t < -0.1)$ denotes the probability that the value of BTC drops more than 10%.

* real value is small number bigger than zero, the computed value is zero because of imprecision in calculation of floating point numbers

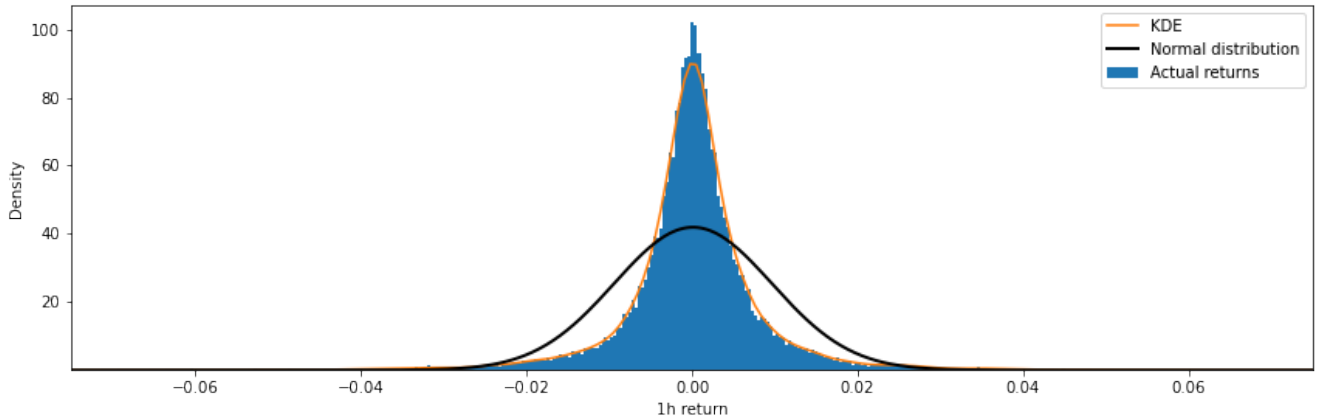


Figure 1.1: Histogram of 1h BTC returns, estimate of actual probability density function (KDE) and comparison with Normal distribution

1.3.5 Market efficiency

According to Efficient Market Hypothesis, efficient markets are such markets where all available information is already reflected in the price and an investor cannot gain an advantage over others, and this has been the subject of research by [12] and [13]. The efficiency of markets was varying in time, with volatility negatively affecting market efficiency. [13] notes that liquid market tends to be more efficient. Both authors point out that periods of market inefficiency in cryptocurrency markets are connected to major events, such as US elections and rapid price changes, like during the Mt.Gox crash and the end of 2017 crash.

Chapter 2

Problem description

We can approach the price prediction of cryptocurrencies in several ways. Here we show our view of the problem and indicate how we want to solve it. Specifically, we look at the data structure and show two main approaches, one of which we choose. We found inspiration in describing time series from the lectures of Kamil Dedecius [14].

2.1 Time series

Time series data is a sequence of observations/measurements, usually taken in regular intervals. They record the development of the observed process in time. Thus we are talking about discrete data.

As their name suggests, measurements are time-dependent. For their meaningful interpretation, we must maintain the time sequence of measurements. We can encounter a time series just about everywhere, such as next week's weather forecast, measured temperature records, river level monitoring, the current increase in new cases of Covid-19 disease, processor load monitoring, or in our case, the price evolution of one of the cryptocurrencies.

Time series

$$\{x_t \mid t = 1, 2, \dots, n\}$$

is a set of measurements in times t . We can also define time series more formally. Let (Ω, F, P) be probability space and let T be a set of indices interpreted as time. Time series is a set $\{X_t \mid t \in T\}$, where X_t are random variables from probability space (Ω, F, P) .

2.2 Stochastic process

Stochastic process in the context of time series is the underlying process from which the measurements/values were generated. Stochastic or also random process can be defined as a sequence of random variables X_t , t is from a suitable set of indices.

Stochastic processes can be divided into two groups, discrete processes and continuous processes. This division determines the index set, typically a subset of natural numbers or non-negative real numbers. When we will use

indices in formulas, we will stick to the notation $t \in \mathbb{N}$, representing a set of consecutive values.

2.3 Means of describing time series

We can describe the time series using a joint distribution of X_{t_1}, \dots, X_{t_n} , but that is not very practical and sometimes not even possible. That is why we describe the time series mainly through its moments - usually mean, variance, autocovariance, higher moments can be also considered

$$\mu_t = \mathbb{E}[X_t]$$

$$\sigma_t^2 = \text{var}(X_t)$$

$$\text{cov}(X_{t_1}, X_{t_2}) = \mathbb{E}[(X_{t_1} - \mu_{t_1})(X_{t_2} - \mu_{t_2})] = \gamma(t_1, t_2)$$

2.4 Stationarity

2.4.1 Strict sense stationarity

We call a process Strict sense stationary if joint distribution $F_X(X_{t_1}, \dots, X_{t_n})$ is same as $F_X(X_{t_1+h}, \dots, X_{t_n+h})$ for all h . In other words, shift by any time h does not have an impact on joint distribution, and the joint distribution only depends on times t_1, \dots, t_n for any n .

2.4.2 Weak sense stationarity

Strict Sense stationarity is in practice too strict to be used for any model. Instead, we define weak sense stationarity. A process is weak sense stationary if time series is invariant to time shifts within moments of probability distribution up to second order. i.e.,

$$\begin{aligned} \mathbb{E}[X_t] &= \mu \quad \forall t \\ \text{cov}(X_t, X_{t+\tau}) &= \gamma(\tau) \\ \mathbb{E}[X_t^2] &< \infty \quad \forall t \end{aligned}$$

$\gamma(\tau)$ indicates that covariance is not a function of time but a function of time shift.

Intuitively, if a process is stationary, it is easier for us to analyse it because its certain statistical properties do not change over time, [15]. If we mention stationarity, we mean weak stationarity unless stated otherwise.

2.5 Random walk

As an example of stochastic process we will take a look at the random walk. It is defined as

$$x_t = x_{t-1} + \epsilon$$

with ϵ typically being generated from set with values $\{-1, 1\}$ and $P(-1) = P(1) = 0.5$ or from Normal distribution with zero mean and unit variance (then it is called Gaussian random walk). Increments in random walk are therefore independent and identically distributed (iid). Figure 2.1 shows an example of random walk with epsilon generated from normal distribution with $\mu = 0$ and $\sigma = 1$. When we compare random walk with selected development of cryptocurrency prices, we can see some similarities between time series. In fact cryptocurrency prices behave very similarly to random walk process. Therefore trying to predict the price is a challenging task. We will describe similarities between those processes later in the context of autocorrelation function (ACF) and partial autocorrelation function (PACF) in the Section 4.3.6.

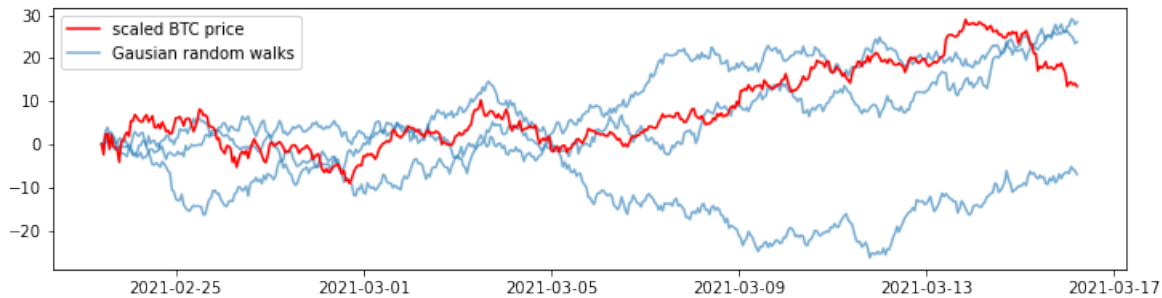


Figure 2.1: Scaled BTC price evolution compared to Gaussian random walks with $\mu = 0$ and $\sigma = 1$

2.6 Problem description

Given a sequence of data points, in our case, cryptocurrency prices, we can look at the problem from two perspectives. In the end, we would like to determine the ideal action based on prediction - BUY, SELL, possibly third action DO NOTHING). One approach would be to look at the problem as a classification task. Given measurements for the last n time steps up to time t , we can classify the change in price from time t to time $t + h$, h being a forecast horizon bigger than zero. Two categories could be possible: price increase, price decrease, or potentially three categories, to distinguish when the change is (too) small, e.g., decrease more than 1%, growth more than 1%, decrease or growth less than 1%. The disadvantage of this approach may be greater uncertainty about the exact change in price, as we only have an interval.

The second approach is to look at the problem as a regression problem. In the regression task, we predict the time series value in time $t + h$. Ideally, this gives us extra information about the price change direction and its exact size. We have applied the second approach, i.e., regression, concerning the form of data as a time series. A fixed forecast horizon of 1 was selected.

2.7 Uncertainty

Performing erroneous action in critical tasks, such as disease identification, autonomous driving or finance management, can have serious practical consequences for its users. It is therefore essential to be able to express uncertainty about prediction estimates. Models should be able to quantify the uncertainty of prediction estimates.

2.8 Simple returns

Simple returns or also used just as *returns* represent percentual gain or loss of given asset from time $t - h$ to t , h is time horizon. We can also look at the price prediction task equivalently as the return prediction task, and we use both perspectives, depending on the approach to the problem. P_t is price in time t . To explain returns on the example, if return R_t is 0.05, it means that the value of the underlying asset rose 5%.

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

2.9 Metrics

To evaluate the model's performance, we use three metrics to quantify the quality of predictions.

2.9.1 Root mean squared error (RMSE)

$$RMSE = \sqrt{\frac{\sum_{i=1}^N (x_t - \hat{x}_t)^2}{N}}$$

x_t is actual value at time t , \hat{x}_t is forecast value at time t . N is number of values.

RMSE "punishes" large errors more than small errors and is scale-dependent, meaning that RMSE of one time series can not be compared to RMSE of some other (or the same) model, but applied on different time series.

■ 2.9.2 Mean average percentage error (MAPE)

$$MAPE = \frac{\sum_{t=1}^N \left| \frac{x_t - \hat{x}_t}{x_t} \right|}{N}$$

MAPE is scale-independent, so the same model can be compared on different time series with different scales. MAPE is not defined when zero is in the ground truth values. Since the prices of cryptocurrencies are always positive, we can safely use this metric.

■ 2.9.3 Mean directional accuracy (MDA)

$$MDA = \frac{1}{N} \sum_{t=1}^N \mathbf{1}(\text{sign}(x_t - x_{t-1}) = \text{sign}(\hat{x}_t - x_{t-1}))$$

$\mathbf{1}$ is indicator function that yields 1 if signs of both expression are equal, 0 otherwise.

Indexing applies if we assume we have one actual value at index 0. MDA is the regression equivalent of accuracy in classification task in terms of the direction of predictions. Values range from 0 (all predicted directional changes are wrong) to 1 (all predicted directional changes are correct). Ultimately, we will be selecting a model with highest MDA. Other metrics will give us additional information on “how far” the estimates are from their real values.

Chapter 3

Dataset

Historical data for cryptocurrencies are publicly available in the form of time bars on multiple exchanges.

Time bars aggregate price movements during a given timeframe (e.g., 15min, 1h, 1d). Price movements are then identified by four pieces of information - open, high, low, close. Additional information, such as volume and volume-weighted average price (VWAP), is sometimes provided.

- Open - first transaction price
- High - maximum transaction price
- Low - minimum transaction price
- Close - last transaction price

This form of data is generally well accessible, but it has some disadvantages. Markets process information as it arrives, most certainly not in regular time intervals. For example, when markets react to new information, periods of higher activity occur. Conversely, periods of lower activity can occur when little or no new information has emerged. Therefore, time bars undersample in high activity periods and oversample in low activity periods.

Tick bars remove this disadvantage by not sampling in regular time intervals but by sampling a predefined number of trades. For example, 1000 trades constitute one tick bar. Tick bars have better statistical properties than time bars, such as correlation, heteroskedasticity and distribution of returns is closer to normal distribution, [16]. However, tick data are difficult to obtain and/or are behind a paywall, so we will use time bars instead and keep the disadvantages in mind.

3.1 Data

All presented models are trained and tested on the same time series, in order to make a quality comparison of different models against each other. We have chosen BTC data on 1h timeframe from Binance exchange, as the most liquid

exchange with the biggest traded volume, [17]. We split the data into three parts: train, validation and test (holdout) datasets. Train dataset is used to train a model, search for the best hyperparameters is conducted on validation dataset, and the model performance is then evaluated on test dataset. Figure 3.1 shows this split. Not all models need separate training and validation datasets. For example ARIMA uses walk-forward optimization. In that case, only the validation dataset is used to search for the best hyperparameters.

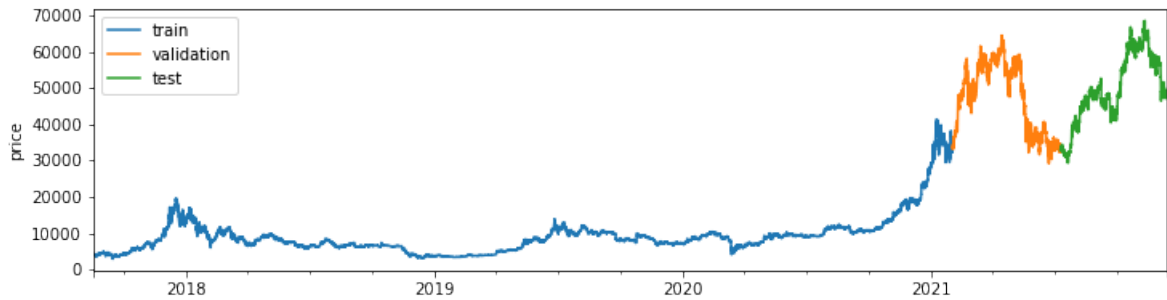


Figure 3.1: Dataset split

3.2 Filling the missing data

Occasionally, some data from a downloaded dataset are missing, either because of an exchange outage or because no trade has taken place within a given timeframe. The first thought that comes to mind is to interpolate the missing data. Nevertheless, that would be a mistake, as it leaks information from the future and gives the model a hint as to which direction the price will move. The best solution for this situation is to apply forward filling, i.e., missing values at times $t, \dots, t + n$ will be filled with the last known value from time $t - 1$.

Chapter 4

Models

In this chapter we will introduce and describe used prediction models.

4.1 Naive model

Naive forecast, or else known as persistence forecast is one the simplest models for regression task. Naive forecast makes assumptions that conditions persist (hence the name persistence forecast) and predicts that the next value will be the same as the last value. It should be not hard to outperform this model regarding the MDA metric. However, it may be hard to outperform the naive model in terms of RMSE metric regarding the similarity of data with random walk, [18].

4.2 Naive directional model

Given that our goal is to predict the price change and most of the changes are bigger than zero in absolute value, we propose another naive model that considers this. We will call it *naive directional* model. We have noticed that on the training dataset, most of the returns for the next step is the opposite of the previous time step. Therefore, the model's predicted change in price between times t and $t + 1$ is the opposite as it was from the time $t - 1$ to time t . The predicted price \hat{x}_{t+1} will be calculated with the use of previous price x_t as

$$\hat{x}_{t+1} = x_t(1 - R_t)$$

Both naive models will serve as baseline models for performance evaluation.

4.3 ARIMA

AutoRegressive Integrated Moving Average model is still a relatively simple model, yet it can be very effective. ARIMA describes autocorrelation in the data to predict the behaviour. Parts in this section were inspired by excellent lectures of Kamil Dedecius [15] on time series.

4.3.1 ACF

Autocorrelation function (ACF) is a correlation between time series and its own shifted copy.

Autocorrelation coefficient of a stochastic process:

$$\rho_{XX}(t_1, t_2) = \frac{\mathbb{E}[(X_{t_1} - \mu_{t_1})(X_{t_2} - \mu_{t_2})]}{\sigma_{t_1}^2 \sigma_{t_2}^2} = \frac{\text{cov}(X_{t_1}, X_{t_2})}{\sigma_{t_1}^2 \sigma_{t_2}^2}$$

For weakly stationary process, where μ and σ are invariant to time shifts:

$$\rho_{XX}(t_1, t_2) = \rho_{XX}(t_1 - t_2) = \rho_{XX}(\tau) = \frac{\mathbb{E}[(X_t - \mu)(X_{t+\tau} - \mu)]}{\sigma^2} = \frac{\gamma(\tau)}{\sigma^2}$$

$\tau = t_1 - t_2$ is lag

Values of autocorrelation function range between 0 and 1, just as correlation between random variables.

4.3.2 PACF

Partial autocorrelation function of lag k is autocorrelation between X_t and X_{t+k} without the influence of values $X_{t+1}, X_{t+2}, \dots, X_{t+k-1}$ in between. In other words PACF shows direct influence of X_t on X_{t+k} . ACF on the other hand shows both direct and indirect influence of previous values.

Partial autocorrelation coefficient

Let X and Y be random variables and between them is some linear dependency. However, both X and Y are also influenced by another n -dimensional random variable, i.e., random vector, Z . If we want to measure the direct influence between X and Y , we have to get rid of Z . We do that by finding the regression lines

$$\begin{aligned}\hat{X} &= a + b^T * Z \\ \hat{Y} &= c + d^T * Z\end{aligned}$$

Variables \hat{X} and \hat{Y} are the best linear approximation of X and Y , and at the same time, variables $e_X = X - \hat{X}$ and $e_Y = Y - \hat{Y}$ are cleaned from the influence of Z because e_X and e_Y are residuals not explained by Z . Partial autocorrelation coefficient is Pearson's correlation coefficient of e_X and e_Y .

$$r_{XY.Z} = \frac{E[e_X e_Y] - E[e_X]E[e_Y]}{\sigma_{e_X} \sigma_{e_Y}}$$

$r_{XY.Z}$ - partial correlation coefficient of X , Y cleaned from the influence of Z

Partial autocorrelation function

PACF of positive lag τ is partial autocorrelation coefficient between X_t and $X_{t+\tau}$ with notation $\alpha(\tau) = r_{X_t X_{t+\tau} \cdot \{X_{t+1}, \dots, X_{t+\tau-1}\}}$

■ 4.3.3 AR process

Values of autoregressive (AR) process depend on previous values, and AR process of order p , with the notation AR(p), has the following form:

$$X_t = c + \phi_1 X_{t-1} + \dots + \phi_p X_{t-p} + \epsilon_t$$

ϕ_i are AR coefficients

ϵ_t is white noise, e.g., Gaussian white noise: $\epsilon \sim iid \mathcal{N}(0, \sigma^2)$

c is constant term

Roots of characteristic equation $1 - \phi_1 z - \dots - \phi_p z^p = 0$ must lie outside the unit circle in order for AR process to be weakly stationary. ADF statistical test is used to test the presence of unit root in time series. Identification of AR order is possible with the help of a graph of PACF.

Autoregressive model enables us to describe the AR process with the same formula that generated those values.

■ 4.3.4 MA process

Values from the Moving average (MA) process are propagated random values through time up to q steps into history. MA process of order q , with the notation MA(q), is described as:

$$X_t = c + \epsilon_t + \theta_1 \epsilon_{t-1} + \dots + \theta_q \epsilon_{t-q}$$

ϵ_t is white noise

c is constant term

MA models use the propagation of ϵ terms to simulate random shocks and their propagation, for example, on financial markets. Charting ACF can help us determine the MA order.

■ 4.3.5 ARIMA(p, d, q) model

ARMA (without I) combines the AR and MA models, but ARMA requires time series to be stationary, which is not always the case. Stationarity can be achieved by differencing time series.

ARIMA(p, d, q) thus combines properties of MA(p) model to model shocks, properties of AR(q) model to predict values with the use of historical realisations and also enables making data stationary (I term – differencing of order d). ARIMA therefore works directly with the prices of cryptocurrencies and also predicts them.

We define ARIMA(p, d, q) model:

$$\Phi(1 - B)^d X_t = \Theta \epsilon_t$$

Where

$$BX_t = X_{t-1}$$

$$\Phi = 1 - \sum_{i=1}^p \phi_i B^i$$

$$\Theta = 1 + \sum_{i=1}^q \theta_i B^i$$

B is called backshift operator.

Confidence interval for one step prediction is:

$$\hat{x}_{t+1} \pm q \sigma_r$$

\hat{x}_{t+1} - point estimate

σ_r - standard deviation of residuals

$q = F^{-1}(\alpha)$ - quantile function of $\mathcal{N}(0, 1)$

α - significance level for confidence intervals

4.3.6 Model characteristics

In the Figure 4.1 are plots of ACF and PACF of 500 selected data points. We can see significant values for PACF at lag 1. This points out to AR(1) characteristic with big ϕ_1 coefficient. Asymptotically declining ACF confirms AR process. However, if we recall the random walk process (Section 2.5), we can notice that random walk is a special case of AR(1) model: $X_t = \phi_1 X_{t-1} + \epsilon_t$. If ϕ_1 is close to 1, we would most likely identify the process as a random walk. Thus, in the long run, it is not suitable to predict data with ARIMA model.

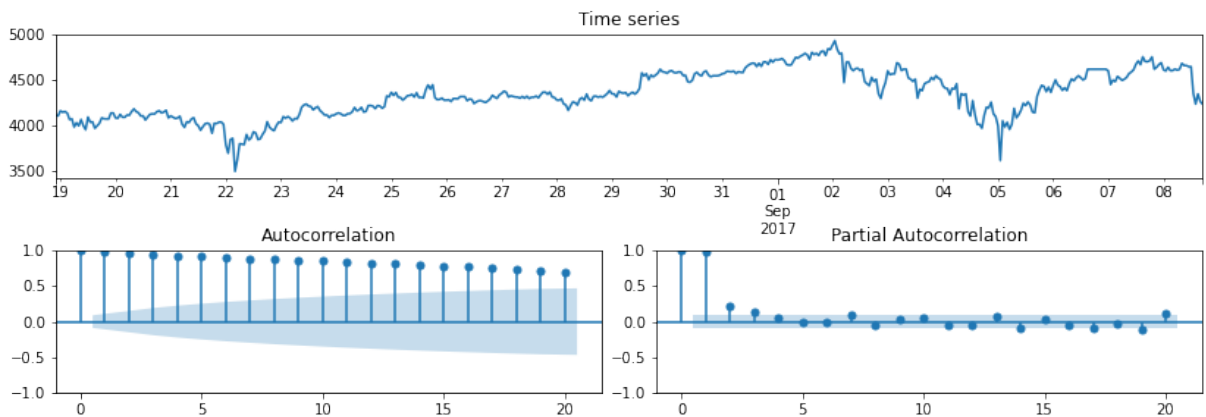


Figure 4.1: ACF, PACF plot of 500 data points

We will look at another example of PACF, ACF plot in the figure 4.2, now with only 80 data points. With fewer data points, we can see significant value for PACF at lag 1 and 3, resembling random walk to much lower degree. It is more common for our data that MA and/or AR characteristics occur for fewer data points than for many points. However, we must also note that AR (or MA) characteristics could be present due to the small number of data points.

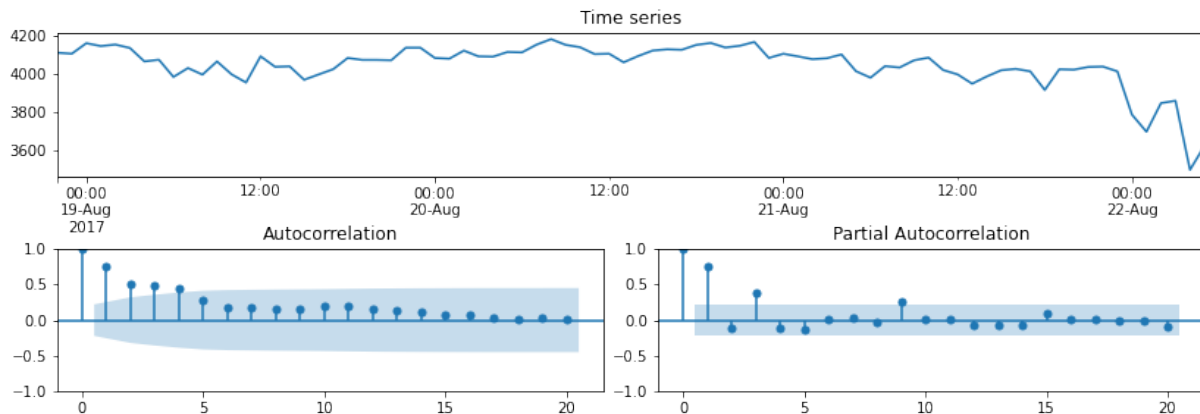


Figure 4.2: ACF, PACF plot of 80 data points

To verify if the data can be predicted with ARIMA model, we have applied walk-forward optimization with fixed rolling window size for training. Training period should be quite small because longer periods resemble random walk, as we have shown above. We used package `pmdarima` and its function for automatic order selection for ARIMA model to automate the process. The function uses statistical tests to determine the best order of differencing and then searches for the most likely parameters.

4.4 LSTM

Long short-term memory (LSTM) is a member of recurrent neural network (RNN) family. RNNs are good at processing sequential data. They are used in speech recognition, text generation, language translation. They can even be used to describe the content in the picture.

The ability to process data sequences is possible due to their property to “remember” important information from earlier data.

This is achieved by linking the output to the input of RNN cell itself. The diagram in the Figure 4.3 illustrates such a link and also shows that passing a data sequence can be unrolled to a sequence of single passes with connections between RNN cells to propagate information.

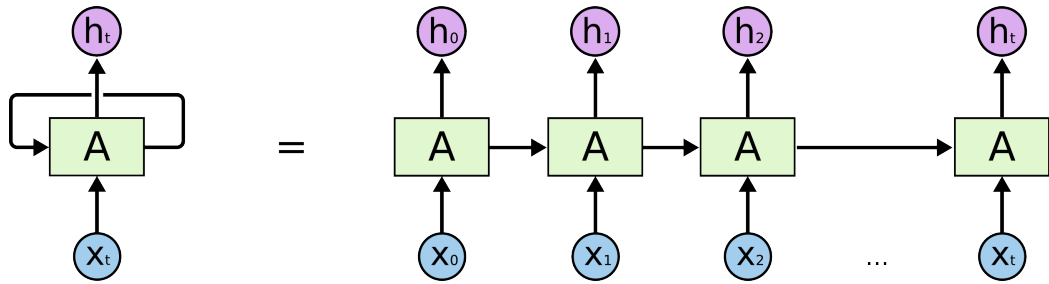


Figure 4.3: RNN and its unrolled form, picture taken from [19]

While early simple RNN cells were in practice not very successful in handling long-term dependencies and were suffering from the vanishing gradient problem, LSTMs are more successful and are now widely used.

Multiple studies have been conducted on this topic, [20], [21], [22], [23] and [24] to name some of them. These studies generally claim that LSTM is a good model to predict cryptocurrency prices or prices of stocks. However, we note that very few of them have an accuracy metric and other focus only on RMSE or another distance measure. Also a comparison with a naive model is missing in the studies, so it is difficult to make an opinion on how LSTM stands against some benchmark. We fill this gap and refine the results of the LSTM model with the comparison with naive baselines.

4.4.1 Internal structure

Now that we have introduced the LSTM, we will show its internal structure, an illustration of which we can see in the Figure 4.4

LSTM has two internal states that it receives and sends on. The first is cell state C_t . The second such state is hidden state h_t , which we can view as working memory, whereas cell state can be seen as long-term memory.

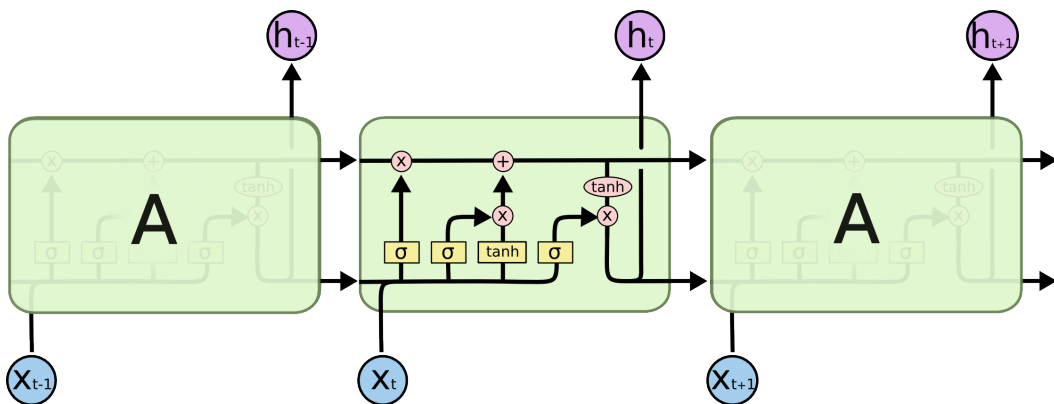


Figure 4.4: Internal structure of LSTM cell, picture taken from [19]

Single forward pass of LSTM cell can be described as

$$\begin{aligned}
 f_t &= \sigma(W_f x_t + U_f h_{t-1} + b_f) \\
 i_t &= \sigma(W_i x_t + U_i h_{t-1} + b_i) \\
 \tilde{C}_t &= \tanh(W_c x_t + U_c h_{t-1} + b_c) \\
 C_t &= f_t \odot C_{t-1} + i_t \odot \tilde{C}_t \\
 o_t &= \sigma(W_o x_t + U_o h_{t-1} + b_o) \\
 h_t &= o_t \odot \tanh(C_t)
 \end{aligned}$$

f_t is forget function, i_t is input function, \tilde{C}_t are candidate cell state values, C_t is new cell state, o_t is output function and h_t is new hidden state. \odot is symbol for piece-wise multiplication, W , U are corresponding matrices.

Various modifications of LSTM exist, such as with peepholes, but [25] experimentally showed that LSTM's main strengths came from forget gate and output activation function, and none of the variants he tested could significantly improve LSTM's capabilities.

4.4.2 Early stopping

When to stop training the model is an important question. One option is to treat the number of epochs as a hyperparameter and optimize it. However, the number of epochs needed to find the best model can vary with different sets of hyperparameters and even with different training sessions of the same model if training contains a non-deterministic part, such as dropout. Early stopping is a regularization method that solves this issue and provides a way to find the desired number of training epochs and prevent overfitting of the model.

Model overfits when the observed metric, typically loss or accuracy on the training dataset keeps improving, but deteriorates on the validation set.

Early stopping monitors given metric and when it starts to get worse, it stops training. Sometimes, the validation loss worsens before it starts improving again, resulting in an even better model. In this case, it may be beneficial to wait and overcome the period of worsened performance. Early stopping with patience implements this approach. [26]

4.4.3 L1 and L2 regularization

L1 and L2 regularization techniques are used to prevent overfitting and ensure better generalization of the model. This techniques adds penalty term to original loss function L such as *mean squared error*, marked as $Error(y, \hat{y})$.

L1 regularization

$$L = Error(x, \hat{x}) + \lambda \sum_{i=1}^k |w_i|$$

L2 regularization

$$L = Error(x, \hat{x}) + \lambda \sum_{i=1}^k w_i^2$$

Effects of L1 are that individual weights w_i are encouraged to be small, which results in a sparse weight matrix and is itself a mean of feature selection, whereas L2 penalizes large weights, forcing weights to be small in general. λ is a hyperparameter determining the influence of L1/L2 on loss. [27]

4.4.4 Dropout

Dropout is a regularization technique that helps to avoid overfitting. It randomly set weights to zero and by that it forces the network to train its different parts.

Applying dropout between recurrent connections in RNNs adds noise to the signal, and the information that would be propagated will be lost. [28] proposes a better version of dropout for RNNs called Variational Dropout. The main difference is that Variational Dropout maintains the same dropout mask, whereas the “standard” dropout samples its mask for each forward pass in NN.

4.4.5 Batch normalization

It is used as a regularization method, especially for deep networks and using them can decrease training time and improve model performance. Batch normalization normalizes data within mini-batches and for each mini-batch modifies features values, so they have zero mean and standard deviation of one. [29]

4.4.6 Data transformation

Inputs to the network were arranged in the following manner. Input for each prediction has a form of $(\mathbf{f}_{t-m}, \mathbf{f}_{t-m+1}, \dots, \mathbf{f}_{t-1})$, where \mathbf{f}_t is a vector of features in time t and m length of sequence or how back into history the model directly sees. Each feature is then scaled into range $(-1, 1)$.

4.4.7 Feature selection

The selection of relevant features is important for the model’s predictive powers. Besides historical logarithmic returns $\ln(R_t + 1)$, the model is also fed with a set of technical indicators: Simple Moving Average, Exponential Moving Average, Bollinger Bands (specifically bandwidth and distance between price and bands of Bollinger Bands indicator), Momentum, Relative Strength Index and Rolling Relative Volume. LSTM’s target values are logarithmic returns for the next time step, that can be converted to simple return as $\ln(R_t + 1)$.

Although LSTM predicts return, we can convert it to a price prediction task as $\hat{P}_{t+1} = P_t(1 + \hat{R}_{t+1})$. Where \hat{P}_{t+1} is predicted price calculated with the knowledge of last observed price P_t and predicted return \hat{R}_{t+1} .

■ 4.4.8 Uncertainty estimate

For LSTM models, we implement two approaches to estimate prediction uncertainty.

■ Standard deviation of residuals

The same approach to quantifying uncertainty used in ARIMA model can be used for LSTM. We can measure the standard deviation of residuals σ_r from the training dataset and use this parameter to approximate a confidence interval of predictions on unseen data.

If we recall the evolution of BTC price in the Figure 3.1, we can see that price moves in a wide range, from around 3000 USD up to more than 60000 USD. This disproportion in prices makes it difficult to estimate standard deviation that would suitably cover the whole interval. Instead, we calculate standard deviation on time series of returns residuals, as they are much more stable.

Given that the training dataset is sufficiently large, we could alternatively estimate confidence intervals directly from KDE of returns residuals. Because as we have shown, normal distribution is not very similar to KDE of returns (the same applies for returns residuals), at least not when we are concerned with tail risk.

■ Monte-Carlo Dropout Uncertainty Estimation

Dropout, as a regularisation technique, is only active during training. When dropout is active both during training and prediction, we can view NN as an ensemble of many models given how the dropout's individual weights are switched off or remain active. Such a network may then generate predictive distribution by repeatedly sampling values from the same input.

This approach is used by the Monte-Carlo Dropout Uncertainty Estimation (MCDUE) used in [30], which estimates predictive mean $\hat{\mu}_{pred}$ and predictive variance $\hat{\sigma}_{pred}^2$ from such a distribution. For the successful use of MCDUE, the model's predictions must meet the property that a larger variation in the estimate is associated with a larger error.

The pseudo-code for predictive distribution sampling is shown below in Algorithm 1. We have also implemented the MCDUE model, but unfortunately, the variation of predictions has a low correlation with the magnitude of the error (Pearson correlation coefficient = 0.3 on training dataset), so we cannot rely on such estimations of uncertainty. Nevertheless, we present a model

with this approach in the results table to compare the effect of active dropout during predictions on metrics.

Algorithm 1: Sampling of predictive distribution

```

Input : Trained model, input data  $X$  and number of samples  $N$ .
Output: Mean  $\mu_{pred}$  and standard deviation of data distribution
           $\sigma_{pred}$ .
/* Model must have active dropout during prediction
   phase. */
list = []
for  $i \leftarrow 0$  to  $N - 1$  do
  |  $prediction = model.predict(X)$ 
  |  $list.append(prediction)$ 
end
 $\sigma_{pred} = std(list)$ 
 $\mu_{pred} = mean(list)$ 
return  $\mu, \sigma$ 

```

4.4.9 NN architecture

Using the techniques and methods mentioned above, we have tried four different model architectures with a different number of LSTM layers. Three models predict only point predictions and they have one, two and three LSTM layers. The fourth model has one layer and leverages MC dropout to estimate the predictive distribution and uncertainty. The loss function is the mean squared error, and the used optimizer is ADAM. Model architectures can be found in Appendix A.

4.5 Results

Results of tested models are summarized in Table 4.1. All metrics are calculated on predicted prices.

LSTM models performed consistently well on all subsets of our dataset. RMSE of the naive model was outperformed only on the training dataset. This confirms that beating the naive model in terms of RMSE is challenging, as outlined earlier. LSTM models mainly dominate MDA, and MAPE metric was only in the domain of LSTM models.

We have also observed a decrease in directional accuracy on the test dataset for all LSTM models. This suggests that the ability to generalize outside of training data or outside of validation data, on which the hyperparameters of models were optimized, is limited.

ARIMA models turned out to be unsuitable for this problem, as both MAPE and RMSE were significantly worse than for other models. However, we must

note that the models were selected based on the MDA criteria. The enormous error for ARIMA models can be explained by a low amount of training data points. Sometimes, the model was badly trained and overshot significantly, which resulted in skewed metrics. Most other ARIMA models had better RMSE metrics, such as 480 for the authentication dataset, but with a lower MDA. We note that results of all ARIMA models were worse than results of other models.

Models described in the Table 4.1:

LSTM-1L Model with one LSTM layer

LSTM-2L Model with two LSTM layers

LSTM-3L Model with three LSTM layers

LSTM-1L MC Model with one LSTM layer that applies MCDUE during inference

ARIMA ARIMA model that was trained on 11 last data points and was retrained after every 10 predictions.

ARIMA without persistence We forced this model not to use ARIMA(0, 0, 0). This setting corresponds to the persistence forecast and does not contribute to MDA metric improvement. The model was trained on the last 15 data points and was retrained after every 5 predictions.

Naive – persistence forecast model

Naive directional predicted return will be the opposite of the last return

Further, we visualise price predictions for some models in the Figure 4.5 and analyse predictions of returns of the LSTM models in the Figure 4.6.

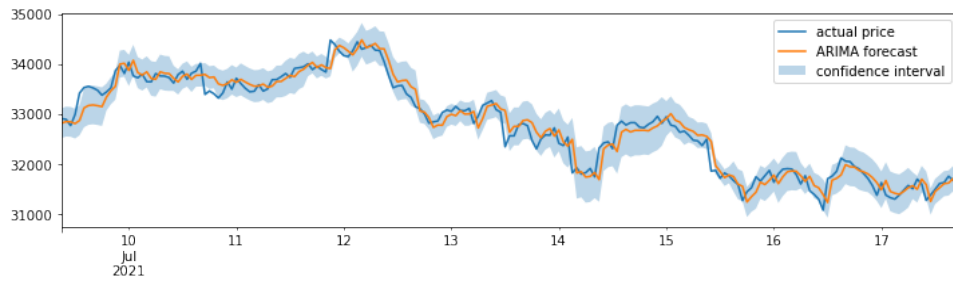
We can see that LSTM-1L's predicted returns are very smooth and look like a moving average. Even though there are signs of fitting to significant values, we can not see much of a pattern in predictions. Interestingly, LSTM-1L MC, the model with the same architecture but a different way of prediction, seems to predict some patterns quite well. We have also shown standard deviation as a way of estimating uncertainty. We can speculate that with bigger volatility of actual returns, predicted standard deviation is also getting bigger, but only to some probably not significant degree. This supports the conclusion not to use MCDUE uncertainty estimate.

Models with more LSTM layers seem to be more confident in predictions of bigger returns. However, it can also be a sign of overfitting. If we look at the predictions on the validation and test datasets in the Figures A.4 and A.5, we can not see that the models generalize well, and it is another evidence for overfitting.

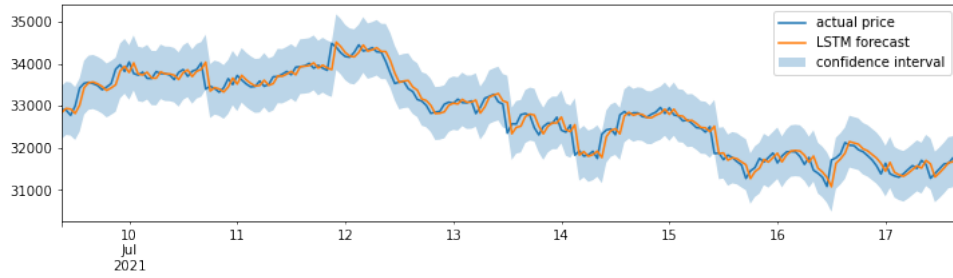
Model	Training			Validation			Test		
	MAPE	RMSE	MDA	MAPE	RMSE	MDA	MAPE	RMSE	MDA
LSTM-1L	0.00538	120.72	0.53	0.00688	456.33	0.5439	0.00478	347.63	0.5299
LSTM-2L	0.00541	119.62	0.5373	0.00691	457.71	0.5453	0.00485	350.82	0.5232
LSTM-3L	0.00528	119.83	0.5328	0.0069	454.47	0.5439	0.00483	350.43	0.51
LSTM-1L MC	0.00537	120.69	0.5115	0.00702	468.86	0.5173	0.00477	346.41	0.5003
ARIMA	0.0134	1524.17	0.5168	0.11725	11107.78	0.5161	0.11331	11090.26	0.5155
ARIMA without persistence	0.00762	532.52	0.517	0.03927	5524.2	0.5263	0.03994	5974.83	0.5063
naive	0.00543	120.58	0.0017	0.00693	456.19	0.0016	0.00478	345.82	0.0019
naive directional	0.00763	166.97	0.5456	0.01008	641.23	0.5129	0.00684	496.32	0.5165

Table 4.1: Results on training, validation and test datasets for tested models

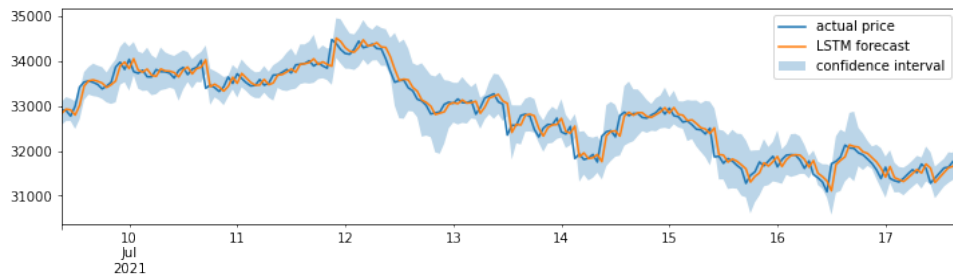
4. Models



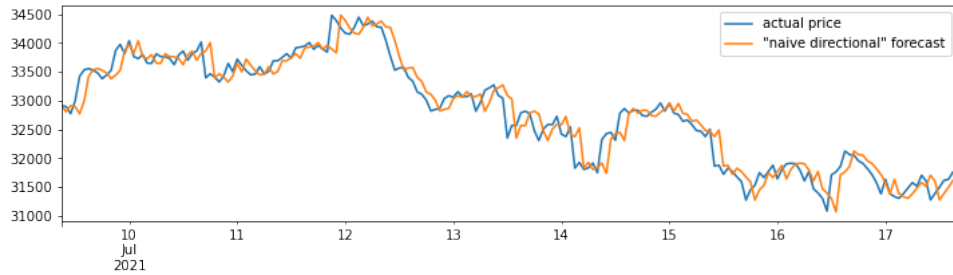
(a) : ARIMA without persistence



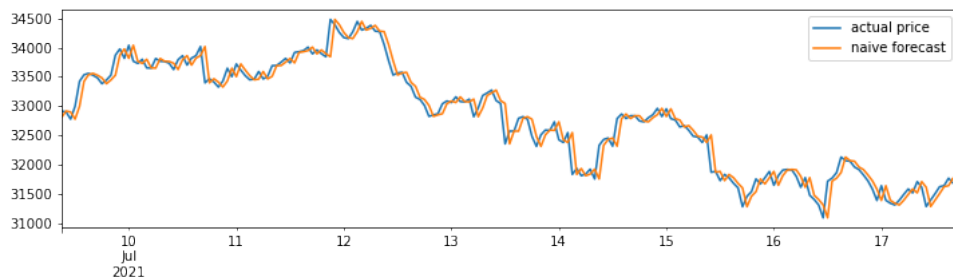
(b) : LSTM-2L, confidence interval of returns is $\hat{R}_t \pm 1.96\sigma_r$, \hat{R}_t is predicted return



(c) : LSTM-1L MC, confidence interval of returns is set as $\hat{R}_t \pm 10\hat{\sigma}_{pred}$



(d) : Naive directional



(e) : Naive

Figure 4.5: Visual comparison of models predictions on the part of the test dataset

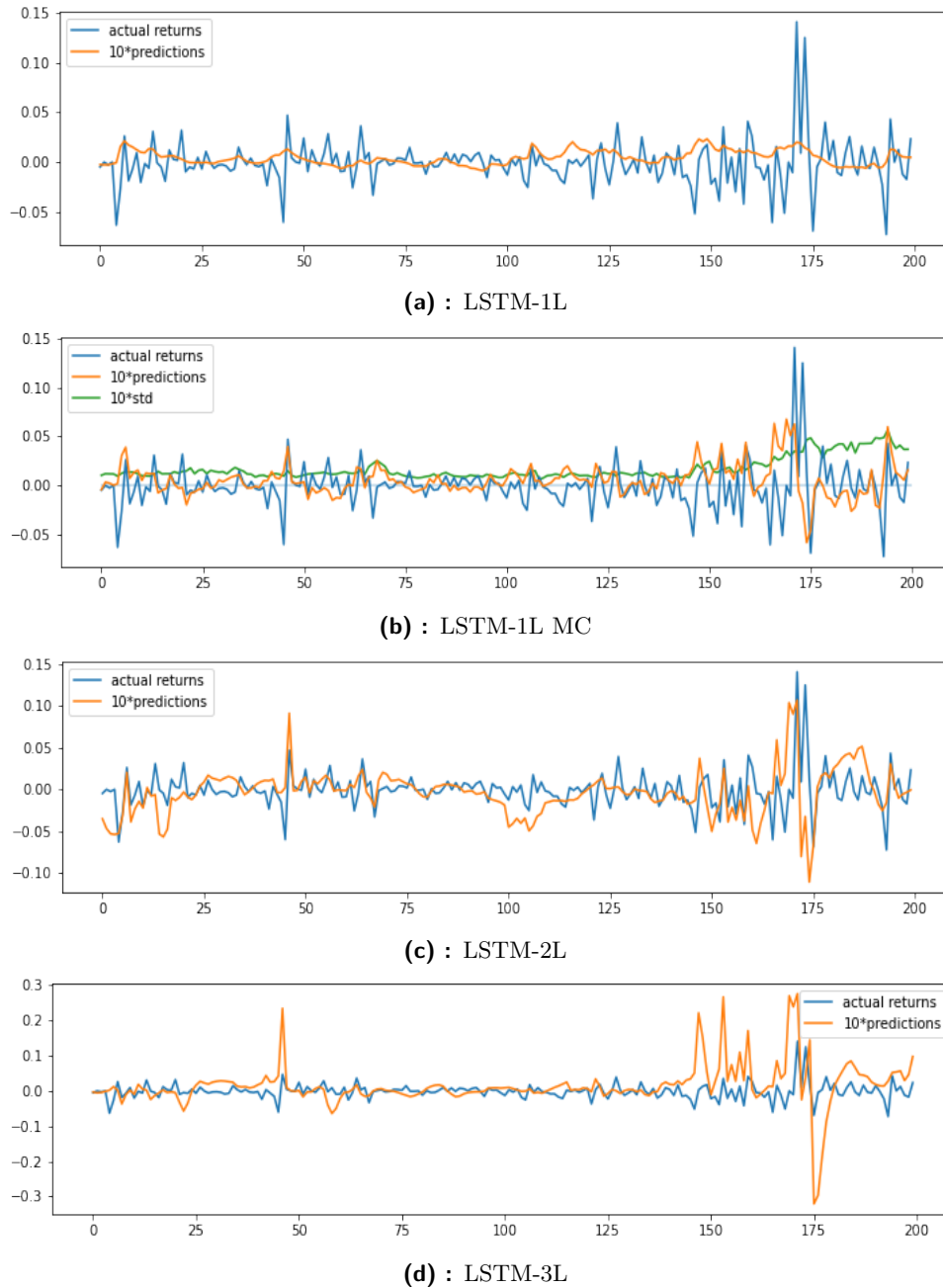


Figure 4.6: Analysis of predictions for different LSTM models on the part of the training dataset. Note that LSTM predicts small values, so the predictions are multiplied by 10.

Chapter 5

Portfolio optimization

In this chapter, we will introduce Kelly criterion, backtest our models using this criterion to determine the size of an investment and review other portfolio optimization methods.

A **portfolio** is a set of financial investments, in our case, a set of cryptocurrencies. The investor chooses in what assets he wants to invest.

5.1 Kelly criterion

As we have defined it, the prediction problem can be viewed as a series of bets, either that the price of a cryptocurrency will grow or decline. Kelly criterion is defined with the assumption that we know underlying probabilities of games outcomes, assumption that we play the same game repeatedly and that number of bets goes to infinity.

Given some initial amount of money M_0 , our wealth at time t is

$$M_t = M_0(1 + f)^S(1 - f)^F$$

f is a fraction of our bank that we bet every time, S is a number of successful bets and F is a number of unsuccessful bets. Naturally, we would like to maximize our wealth. Kelly criterion determines the optimal size for a bet and maximizes wealth in the long run.

$$G(f) = \ln\left(\frac{X_n}{X_0}\right)^{\frac{1}{n}} = \frac{S}{n}\ln(1 + f) + \frac{F}{n}\ln(1 - f)$$

is a measure of exponential rate of increase per trial. Kelly criterion maximizes wealth by maximizing growth rate $g(f)$, where

$$g(f) = \mathbb{E}\left[\ln\left(\frac{X_n}{X_0}\right)^{\frac{1}{n}}\right] = \mathbb{E}\left[\frac{S}{n}\ln(1 + f) + \frac{F}{n}\ln(1 - f)\right] = p*\ln(1 + f) + q*\ln(1 - f)$$

$g(f)$ is maximal for $f^* = p - q$, p is the probability of win, q is a probability of loss. This is the case when for every unit wager player either wins a unit or loses a unit of his money. This formula can be extended to uneven payoff

games. Suppose that for every unit wager player loses a units and wins b units. Then $g(f)$ is maximal when

$$f^* = \frac{p}{a} - \frac{q}{b}$$

In cryptocurrency markets, it is rarely the case that we are certain about the probability that the bet will be successful. Kelly criterion is therefore used as an estimate. Fractional Kelly criterion can be used to mitigate the risk of overbetting, which potentially leads to ruin. Fractional Kelly bets some fraction of Kelly criterion, e.g., 1/2 for Half Kelly.

Kelly criterion can be estimated with the help of Taylor polynomial as

$$f = \mathbb{E}[R_t] - \frac{1}{2} \text{Var}[R_t]$$

We will use the approximated Kelly criterion since our estimate of uncertainty corresponds better to this approximated criterion, i.e., predicted return will be $\mathbb{E}[R_t]$ and estimated variance $\text{Var}[R_t]$. [31], [32]

5.2 Review of other portfolio optimization methods

In this section we briefly introduce two other portfolio optimization approaches.

5.2.1 Mean-variance portfolio

Harry Markowitz proposed this method of portfolio optimization in 1952, [33]. He uses the variance of assets as a proxy of risk and then minimizes risk for a given return. For n assets, we search for optimal weights vector $\mathbf{w} = (w_1, \dots, w_n)$, w_i is weight of i -th asset in portfolio, such that the variance of the portfolio is minimal.

$$\begin{aligned} & \underset{\mathbf{w}}{\text{minimize}} && \frac{1}{2} \mathbf{w}^\top \Sigma \mathbf{w} \\ & \text{subject to} && \mathbf{m}^\top \mathbf{w} \geq \mu_b \\ & && \mathbf{1}^\top \mathbf{w} = 1 \end{aligned}$$

Σ is correlation matrix of assets returns, μ_b is a minimal return of a portfolio that we want to achieve, and \mathbf{m} is a vector of expected returns associated with individual assets. We therefore minimize variance of portfolio $\frac{1}{2} \mathbf{w}^\top \Sigma \mathbf{w}$ for desired return μ_b , and we also want weights of individual investments to sum to one.

Mean-variance portfolio optimization has some disadvantages, such as that variance is not a good measure of risk. This framework can be modified to work with other risk measures, e.g., [34], and to be able to solve additional

constraints, for example, long positions only portfolio.

Also, mean-variance portfolios concentrate funds in only a few assets, ignoring the risk of diversification, and this optimization method is sensitive to parameter estimation errors for Σ and \mathbf{m} . Risk parity portfolios tackle this problem. [35]

5.2.2 Risk parity portfolio

Risk parity portfolio focuses on allocation of risk rather than allocation of capital, as mean-variance portfolios do. It is also less sensitive to parameter estimation errors. Risk parity portfolios performed well during the 2007-2008 financial crisis, which suggested the success of this allocation strategy compared to other portfolios. [35]

5.3 Portfolio optimization and backtesting strategy based on Kelly criterion

We can assume that a rational investor wants to maximize the value of his portfolio. This section shows an example of such portfolios. To determine the size of a bet, we use approximated Kelly criterion outlined in the previous section.

Our minimal portfolio consists of two cryptocurrencies, namely Bitcoin (BTC) and Tether (USDT). Tether is a cryptocurrency whose value is backed by the value of the real currency (also called fiat), in this case by US Dollar. For every model's prediction, we calculate the fraction of the portfolio that we should bet on BTC and the rest (1-fraction) is kept in USDT.

The value of the portfolio in each moment is calculated as

$$M_t = M_0(1 + R_1^p)(1 + R_2^p)\dots(1 + R_t^p)$$

where M_0 is the portfolio's initial value, R_t^p is the portfolio's return from time $t - 1$ to time t . R_t^p can be calculated as $f_{t-1}R_t$, where R_t is a return of BTC and f_{t-1} is a fraction of the portfolio that is invested in BTC.

Multiple portfolios backtests are performed. We use LSTM-2L as our prediction model, with standard deviation derived from residuals on the training dataset, and LSTM-1L MC, which uses variance from its predictive distribution. Different strategies are tested. The buy and hold strategy does what its name suggests, buy at the beginning and hold the position. It can be viewed as a benchmark.

Some strategies do not use the Kelly criterion. Those have 'fixed bet x%' in their name, also suggesting the size of the position. Thus, the strategy 'LSTM-2L fixed bet 10%' always bets 10% of the portfolio value. A transac-

tion fee of 0.1% is applied on strategies that have ‘fees’ in their name to get closer to real market conditions and better assess the model’s performance.

Results of backtest are visualised in the Figure 5.1

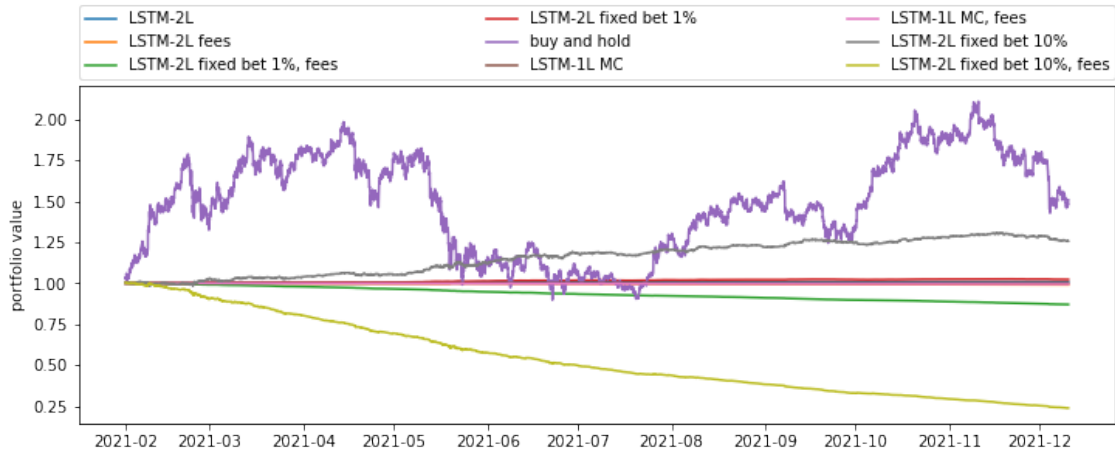


Figure 5.1: Results of backtested strategies

LSTM-2L: 2 layer LSTM model, LSTM-1L MC: 1 layer LSTM model with MC sampling, fees are 0.1%

5.4 Results of Kelly criterion optimization

Backtesting without fees resulted in slight profits for all LSTM models, the biggest profit had strategy that was betting 10% of its portfolio value each time.

However, introducing transaction fee resulted in almost continuous loss of all portfolio’s value. It is worth noting that the fastest loss of value has the portfolio betting 10%, the same portfolio which had biggest gains. Buy and hold strategy had biggest gains overall and we did not manage to beat it with our models, especially when the fees were applied.

Several factors explain the inability of LSTM models to generate a profit. We see the first reason in the model’s predictions and uncertainty. All the predictions are relatively small, and the uncertainty is high, which is why Kelly criterion chooses small fractions to bet. The second reason is the fees and holding period for the position. With frequent trading, we’re exposed to fees many times over, and even small ones add up over time. And the last reason is undoubtedly the quality of the predictions, when we look at the directional accuracy of the model, it is not very high for any model.

■ 5.4.1 Limitations of backtesting

In our backtest, we performed actions (BUY, SELL), provided that the market would not be affected by this. But this may not always be the case. If we traded in large amounts, we would likely affect the price to our detriment. What is a large amount depends on the circumstances. It will vary if we trade BTC, which has one of the largest cryptocurrency volumes traded, or if we trade some unknown, unpopular cryptocurrency.

If we plan to trade large volumes, it is possible to incorporate this disadvantage into the backtest and simulate each trade with a certain loss, also called slippage.

Chapter 6

Conclusion

LSTM has proven to be a good model compared to other models. A detailed examination of LSTM predictions has shown that the network is able to learn some patterns in a time series. However, we saw this capability as very limited for out-of-sample data.


Our prediction uncertainty estimates have shown us that the network is very uncertain. This complicated our work in the part of optimizing the cryptocurrency portfolio. Determining the size of a bet with approximated Kelly criterion is based on an estimate of the prediction uncertainty and the magnitude of the prediction itself. The combination of uncertainty and small predicted returns made Kelly bet tiny parts of the portfolio.

However, by backtesting, we experimentally verified that the predictions of the LSTM model were more correct than wrong. As a result, we have made a profit if we do not include transaction fees. When we counted them, many small fees added up, and none of the LSTM models was profitable.

6.1 Future work

We observed an interesting phenomenon when comparing the predictions of the one layer LSTM model and the same architecture model, only using MC sampling. The patterns of predictions from the model using MC sampling were more similar to patterns of real returns. Exploring this property could be a topic for further development.

Also, involving more features, such as market sentiment, could help the model improve its prediction capabilities. Another way to explore the model's predictive capabilities could be to consider only a certain subset of predictions, e.g., only large predicted returns.



Chapter 7

Acronyms

BTC Bitcoin

ETH Ethereum

KDE Kernel Density Estimate

ACF Autocorrelation Function

PACF Partial Autocorrelation Function

MDA Mean Directional Accuracy

RMSE Root Mean Squared Error

MAPE Mean Absolute Percentage Error

AR Autoregressive

MA Moving-Average

ARMA Autoregressive Moving-Average

ARIMA Autoregressive Integrated Moving-Average

RNN Recurrent Neural Network

LSTM Long Short-Term Memory

NN Neural Network

MC Monte-Carlo

MCDUE Monte-Carlo Dropout Uncertainty Estimation



Bibliography

- [1] Ladislav Kristoufek. “What are the main drivers of the Bitcoin price? Evidence from wavelet coherence analysis”. In: *PloS one* 10.4 (2015), e0123923.
- [2] Walid Mensi et al. “Time frequency analysis of the commonalities between Bitcoin and major Cryptocurrencies: Portfolio risk management implications”. In: *The North American Journal of Economics and Finance* 48 (2019), pp. 283–294. ISSN: 1062-9408. DOI: <https://doi.org/10.1016/j.najef.2019.02.013>. URL: <https://www.sciencedirect.com/science/article/pii/S1062940818303322>.
- [3] Pavel Ciaian, Miroslava Rajcaniova, and d’Artis Kancs. “Virtual relationships: Short- and long-run evidence from BitCoin and altcoin markets”. In: *Journal of International Financial Markets, Institutions and Money* 52 (2018), pp. 173–195. ISSN: 1042-4431. DOI: <https://doi.org/10.1016/j.intfin.2017.11.001>. URL: <https://www.sciencedirect.com/science/article/pii/S1042443117302858>.
- [4] Paulo Ferreira and Éder Pereira. “Contagion Effect in Cryptocurrency Market”. In: *Journal of Risk and Financial Management* 12.3 (2019). ISSN: 1911-8074. URL: <https://www.mdpi.com/1911-8074/12/3/115>.
- [5] Nikolaos Antonakakis, Ioannis Chatziantoniou, and David Gabauer. “Cryptocurrency market contagion: Market uncertainty, market complexity, and dynamic portfolios”. In: *Journal of International Financial Markets, Institutions and Money* 61 (2019), pp. 37–51. ISSN: 1042-4431. DOI: <https://doi.org/10.1016/j.intfin.2019.02.003>. URL: <https://www.sciencedirect.com/science/article/pii/S1042443118303469>.
- [6] Andrew Phillip, Jennifer S.K. Chan, and Shelton Peiris. “A new look at Cryptocurrencies”. In: *Economics Letters* 163 (2018), pp. 6–9. ISSN: 0165-1765. DOI: <https://doi.org/10.1016/j.econlet.2017.11.020>. URL: <https://www.sciencedirect.com/science/article/pii/S0165176517304731>.

- [7] Konstantinos Gkillas and Paraskevi Katsiampa. “An application of extreme value theory to cryptocurrencies”. In: *Economics Letters* 164 (2018), pp. 109–111. ISSN: 0165-1765. DOI: <https://doi.org/10.1016/j.econlet.2018.01.020>. URL: <https://www.sciencedirect.com/science/article/pii/S0165176518300284>.
- [8] Joerg Osterrieder and Julian Lorenz. “A statistical risk assessment of Bitcoin and its extreme tail behavior”. In: *Annals of Financial Economics* 12.01 (2017), p. 1750003.
- [9] Stoyan V Stoyanov et al. “Fat-tailed models for risk estimation”. In: *The Journal of Portfolio Management* 37.2 (2011), pp. 107–117.
- [10] Markus Haas and Christian Pigorsch. “Financial Economics, Fat-Tailed Distributions.” In: *Encyclopedia of Complexity and Systems Science* 4.1 (2009), pp. 3404–3435.
- [11] *Scipy - gaussian KDE*. https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.gaussian_kde.html. [Online; accessed 15-November-2021].
- [12] Akihiko Noda. “On the evolution of cryptocurrency market efficiency”. In: *Applied Economics Letters* 28.6 (2021), pp. 433–439.
- [13] Khamis Hamed Al-Yahyaee et al. “Why cryptocurrency markets are inefficient: The impact of liquidity and volatility”. In: *The North American Journal of Economics and Finance* 52 (2020), p. 101168. ISSN: 1062-9408. DOI: <https://doi.org/10.1016/j.najef.2020.101168>. URL: <https://www.sciencedirect.com/science/article/pii/S1062940820300656>.
- [14] Kamil Dedecius. “Úvod do problematiky, exponenciální vyhlazování”, FIT, CTU in Prague. <https://gitlab.fit.cvut.cz/dedecam/mi-scr/blob/master/prednasky/1.ipynb>. [Online; accessed 9-November-2021]. 2021.
- [15] Kamil Dedecius. “Vlastnosti časových řad”, FIT, CTU in Prague. <https://gitlab.fit.cvut.cz/dedecam/mi-scr/blob/master/prednasky/1.ipynb>. [Online; accessed 10-December-2021]. 2021.
- [16] Marcos Lopez De Prado. *Advances in financial machine learning*. John Wiley & Sons, 2018.
- [17] *Cryptocurrencies exchanges comparison*. <https://coinmarketcap.com/rankings/exchanges/>. [Online; accessed 8-May-2021].
- [18] Rob J Hyndman and George Athanasopoulos. *Forecasting: Principles and Practice*. [Online; accessed 20-October-2021]. 2018. URL: <https://otexts.com/fpp2/stationarity.html#stationarity>.
- [19] Christopher Olah. *Understanding LSTM Networks*. <https://colah.github.io/posts/2015-08-Understanding-LSTMs/>. [Online; accessed 14-November-2021]. 2015.
- [20] Bruno Spilak. “Deep neural networks for cryptocurrencies price prediction”. MA thesis. Humboldt-Universität zu Berlin, 2018.

- [21] Patrick Jaquart, David Dann, and Christof Weinhardt. “Short-term bitcoin market prediction via machine learning”. In: *The Journal of Finance and Data Science* 7 (2021), pp. 45–66.
- [22] Omnia Kelany, Sherin Aly, and Mohamed A Ismail. “Deep Learning Model for Financial Time Series Prediction”. In: *2020 14th International Conference on Innovations in Information Technology (IIT)*. IEEE, 2020, pp. 120–125.
- [23] Ferdiansyah Ferdiansyah et al. “A lstm-method for bitcoin price prediction: A case study yahoo finance stock market”. In: *2019 International Conference on Electrical Engineering and Computer Science (ICECOS)*. IEEE, 2019, pp. 206–210.
- [24] Murtaza Roondiwala, Harshal Patel, and Shraddha Varma. “Predicting stock prices using LSTM”. In: *International Journal of Science and Research (IJSR)* 6.4 (2017), pp. 1754–1756.
- [25] Klaus Greff et al. “LSTM: A search space odyssey”. In: *IEEE transactions on neural networks and learning systems* 28.10 (2016), pp. 2222–2232.
- [26] Jason Brownlee. *A Gentle Introduction to Early Stopping to Avoid Overtraining Neural Networks*. <https://machinelearningmastery.com/early-stopping-to-avoid-overtraining-neural-network-models/>. [Online; accessed 25-November-2021]. 2018.
- [27] Kurtis Pykes. *Fighting Overfitting With L1 or L2 Regularization: Which One Is Better?* <https://machinelearningmastery.com/early-stopping-to-avoid-overtraining-neural-network-models/>. [Online; accessed 4-December-2021].
- [28] Yarin Gal and Zoubin Ghahramani. “A theoretically grounded application of dropout in recurrent neural networks”. In: *Advances in neural information processing systems* 29 (2016), pp. 1019–1027.
- [29] Jay Whang and A Matsukawa. “Exploring Batch Normalization in Recurrent Neural Networks”. In: *Stanford Center for Professional Development*. Available online: https://jaywhang.com/assets/batchnorm_rnn.pdf (accessed on 28 September 2020) ().
- [30] Evgenii Tsymbalov, Maxim Panov, and Alexander Shapeev. “Dropout-based active learning for regression”. In: *International conference on analysis of images, social networks and texts*. Springer, 2018, pp. 247–258.
- [31] Uhrín Matej et al. “Optimal sports betting strategies in practice: an experimental review”. In: *IMA Journal of Management Mathematics* (2021).
- [32] Edward O Thorp. “The Kelly criterion in blackjack sports betting, and the stock market”. In: *Handbook of asset and liability management*. Elsevier, 2008, pp. 385–428.

- [33] Harry Markowitz. “Portfolio Selection”. In: *The Journal of Finance* 7.1 (1952), pp. 77–91. ISSN: 00221082, 15406261. URL: <http://www.jstor.org/stable/2975974>.
- [34] Diana Roman, Kenneth Darby-Dowman, and Gautam Mitra. “Mean-risk models using two risk measures: a multi-objective approach”. In: *Quantitative Finance* 7.4 (2007), pp. 443–458. DOI: 10.1080/14697680701448456. eprint: <https://doi.org/10.1080/14697680701448456>. URL: <https://doi.org/10.1080/14697680701448456>.
- [35] Daniel P. Palomar. *Risk parity portfolio*. https://palomar.home.ece.ust.hk/ELEC5470_lectures/slides_risk_parity_portfolio.pdf. [Online; accessed 12-October-2021]. 2020.

Appendix A

Figures

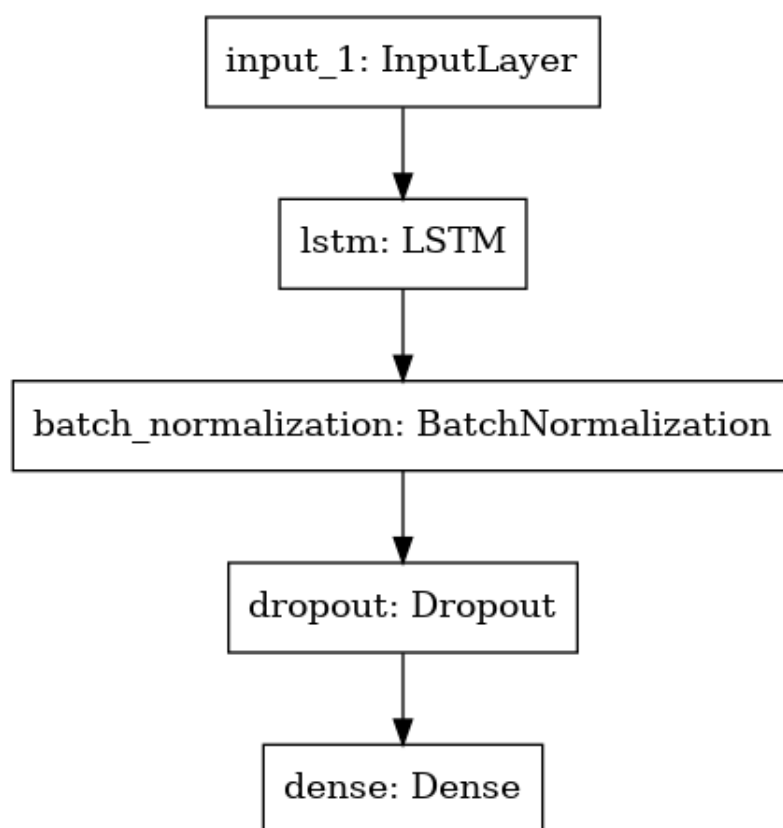


Figure A.1: Architecture of 1 layer LSTM model

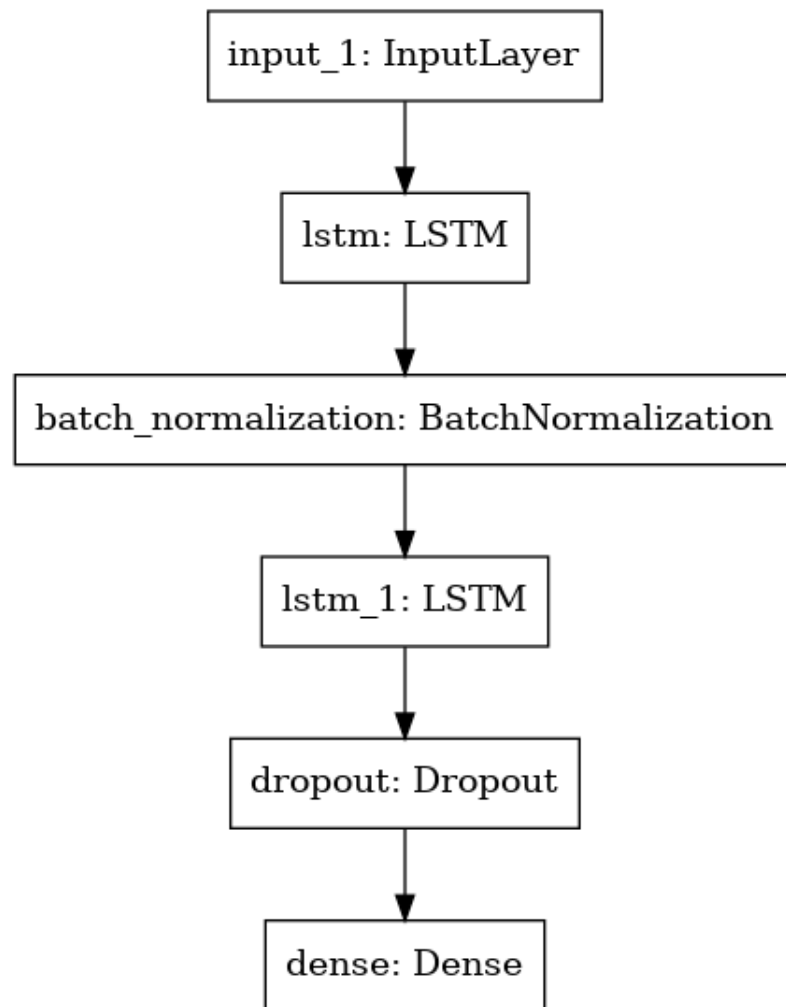


Figure A.2: Architecture of 2 layer LSTM model

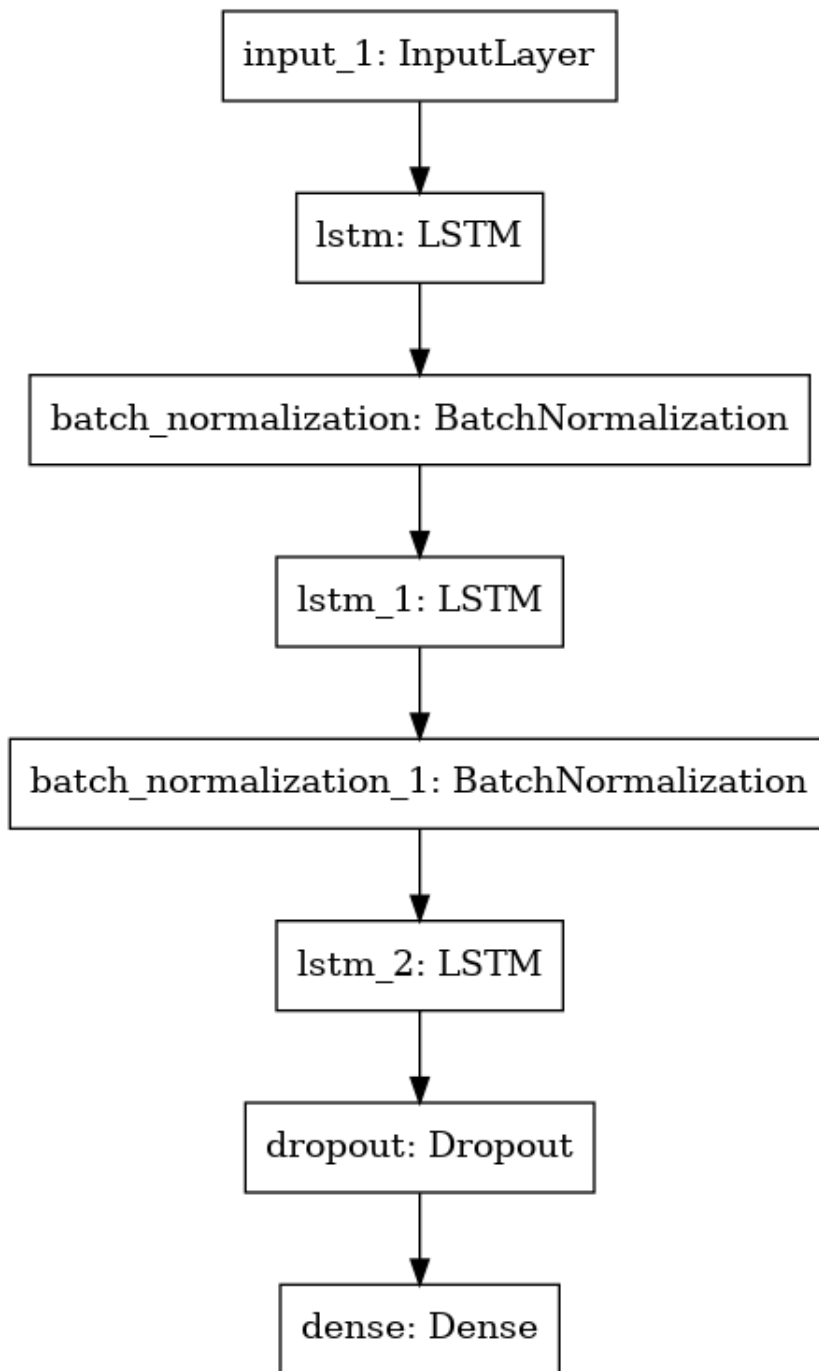
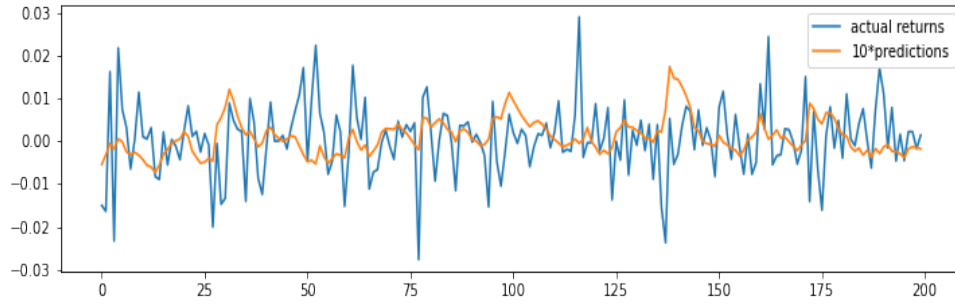
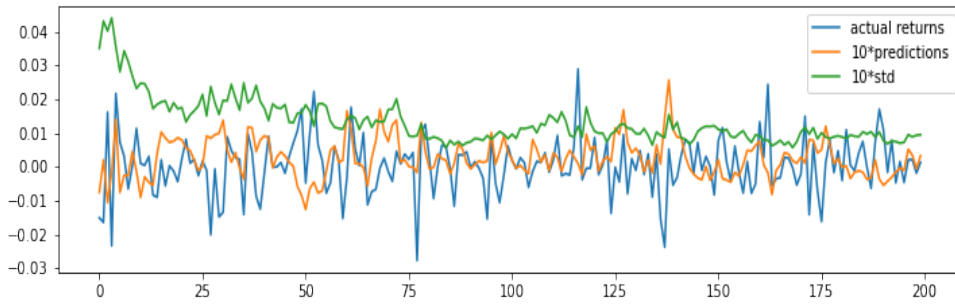


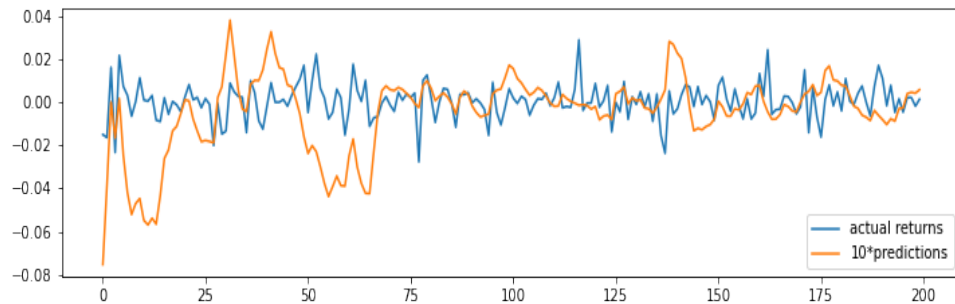
Figure A.3: Architecture of 3 layer LSTM model



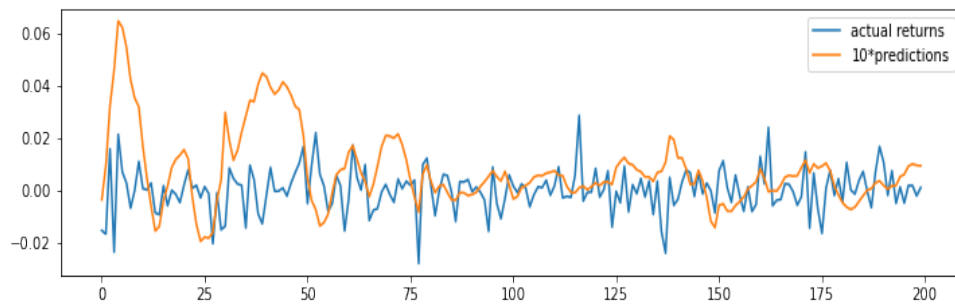
(a) : LSTM-1L



(b) : LSTM-1L MC

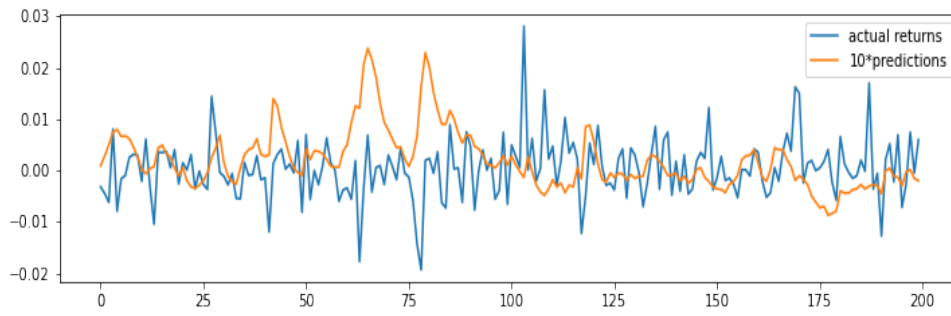


(c) : LSTM-2L

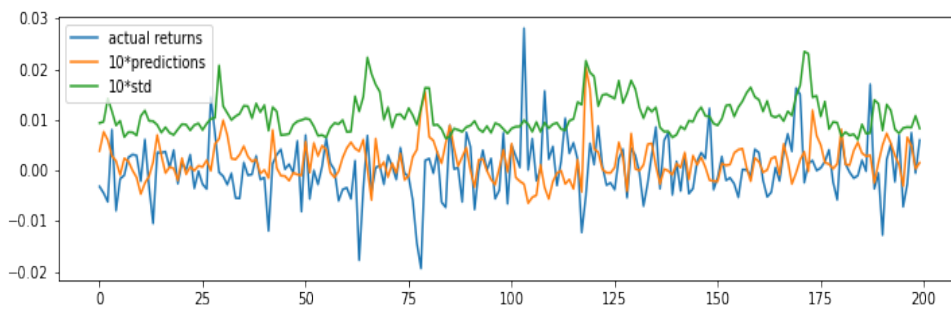


(d) : LSTM-3L

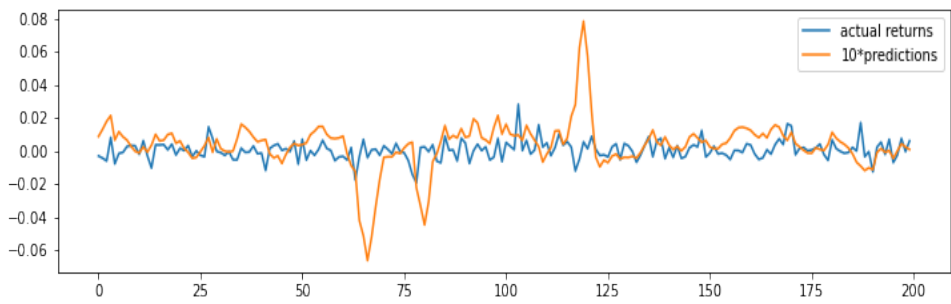
Figure A.4: Analysis of predictions for different LSTM models on the part of the validation dataset. Note that LSTM predicts small values, so the predictions are multiplied by 10.



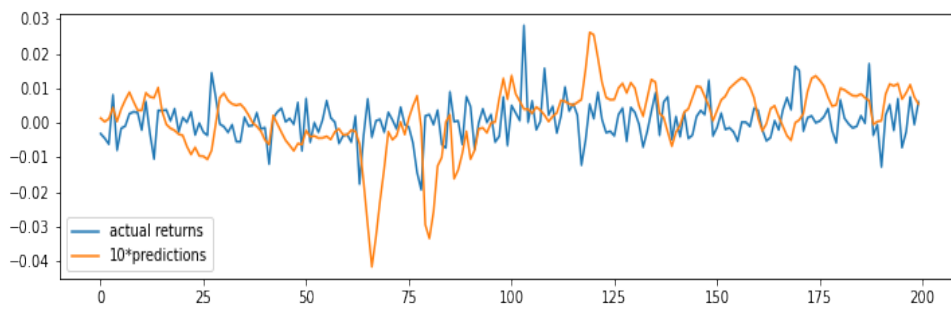
(a) : LSTM-1L



(b) : LSTM-1L MC



(c) : LSTM-2L



(d) : LSTM-3L

Figure A.5: Analysis of predictions for different LSTM models on the part of the test dataset. Note that LSTM predicts small values, so the predictions are multiplied by 10.