University of Warsaw
Faculty of Economic Sciences

Working Papers
No. 2/2019 (287)

ROBUSTNESS OF SUPPORT VECTOR MACHINES IN ALGORITHMIC TRADING ON CRYPTOCURRENCY MARKET

Maryna Zenkova
Robert Ślepaczuk

Warsaw 2019

WORKING PAPERS 2/2019 (287)

Robustness of Support Vector Machines in Algorithmic Trading on Cryptocurrency Market

Maryna Zenkova, Robert Ślepaczuk*

Quantitative Finance Research Group, Faculty of Economic Sciences, University of Warsaw
** Corresponding author: robert.slepaczuk@gmail.com*

Abstract: This study investigates the profitability of a algorithmic trading strategy based on training SVM model to identify cryptocurrencies with high or low predicted returns. A tail set is defined to be a group of coins whose volatility-adjusted returns are in the highest or lowest quantile. Each cryptocurrency is represented by a set of six technical features. SVM is trained on historical tail sets and tested on the current data. The classifier is chosen to be a nonlinear support vector machine. Portfolio is formed by ranking coins using SVM output. The highest ranked coins are used for long positions to be included in the portfolio for one reallocation period. The following metrics were used to estimate the portfolio profitability: %ARC (the annualized rate of change), %ASD (the annualized standard deviation of daily returns), MDD (the maximum drawdown coefficient), IR1, IR2 (the information ratio coefficients). The performance of the SVM portfolio is compared to the performance of the four benchmark strategies based on the values of the information ratio coefficient IR1 which quantifies the risk-weighted gain. The question on how sensitive the portfolio performance is to the parameters set in the SVM model is also addressed in this study.

Keywords: machine learning, support vector machines, investment algorithm, algorithmic trading, strategy, optimization, cross-validation, overfitting, cryptocurrency market, technical analysis, meta parameters

JEL codes: C4, C45, C61, C15, G14, G17

Disclaimer: The views presented in this text are those of the authors and do not necessarily represent those of Circus Consulting Group nor LHF project.

1. Introduction

The method of support vectors was developed by Vladimir Vapnik in 1995 and was first applied to the task of classification of texts by Joachims in 1998. In its original form, the algorithm solved the problem of distinguishing objects of two classes. The method has gained immense popularity due to its high efficiency. Many researchers used it in their work in the classification of texts. The approach proposed by Vapnik to determine to which of the two predefined classes the sample should belong adheres to the principle of structural minimization of risk.

The results of the classification of texts using the support vector method are among the best, in comparison with other machine learning approaches. However, the learning speed of this algorithm is one of the lowest. The method of SVM requires a large amount of memory and exercise a significant computational load for the computer to perform the training. Summing up, the simplicity combined with state of the art performance on many learning problems (classification, regression, and novelty detection) has contributed to the popularity of SVM.

SVM is also quite popular algorithm for building trading systems. It was used mostly to predict stock or index price movement whether it will go up or down. However, hardly any papers reveal how SVM performs on cryptocurrency market where the asset price volatility is much higher than on traditional markets.

The goal of this paper is to apply SVM algorithm to build an investment strategy for cryptocurrency market and investigate its profitability. The research hypothesis is that the strategy based on SVM algorithm is able to outperform the benchmark strategies in terms of return-risk relation. Similarly to Ślepaczuk et al. (2018) we will use the following metrics to estimate the portfolio profitability: %ARC (the annualized rate of change), %ASD (the annualized standard deviation of daily returns), MDD (the maximum drawdown coefficient), IR1, IR2 (the information ratio coefficients). The assessment of the research hypothesis will be based on the value of the IR1 which quantifies the return-risk ratio. The research questions addressed in this study are formulated around the sensitivity analysis results, namely what is the sensitivity of the portfolio performance to the main parameters set in the SVM model. Four parameters were selected in the sensitivity analysis. These parameters are number of cryptocurrencies kept in the portfolio, reallocation period, percentage value of the transaction costs, training data size TS. The parameters that were set as fixed are the following: length of historical data taken to calculate technical features, lambda λ used to

calculate exponential moving average for returns, meta parameters C and γ, length of training data and long positions only assumption.

The main idea and methodology concepts were adopted from the research paper „Nonlinear support vector machines can systematically identify stocks with high and low future returns" by Huerta et. al. (2013) and Kość et. al. (2018) "Momentum and contrarian effects on the cryptocurrency market".

SVM is implemented to build the trading strategy in the following way. Training set is basically a tail set which is defined to be a group of coins whose volatility-adjusted price change is in the highest or lowest quantile, for example the highest and the lowest 25 coins. Each coin is represented by a set of technical features. A classifier is trained on historical tail sets and tested on current data. The classifier is chosen to be a nonlinear support vector machine. The SVM is trained once per reallocation period. If portfolio is reallocated once per week, SVM is trained once per week accordingly. Portfolio is formed by ranking coins using the classifier output. The highest ranked coins are used for long positions and the lowest ranked potentially can be used for short sales. The data cover the period from 01/01/2015 to 08/01/2018.

The structure of the paper can be summarized as follows. After literature review in the theoretical part a short introduction into support vector machines is provided. There are three concepts such that the maximal margin classifier, the support vector classifier, and the support vector machine. Second section focuses on data, methodology and strategy implementation. Third part provides empirical results in comparison with benchmark investment strategies. Sensitivity analysis is the final part of the thesis.

1.1 Literature review

The theoretical background for SVM is based mainly on James et al. (2017) and Hastie et al. (2017). The nonlinear SVM classifier was chosen to run the strategy in this paper because it worked well in multiple previous applications. Additionally, it was proven to be convenient to use and fast to train (Vapnik (1999) and Muller et al. (2001)).

There are many methods in machine learning (for example neural networks) that might work as good as SVM, but the simplicity of the mathematical functions, and the theory that guides the training of the model as a convex optimization problem (Boyd & Vandenberghe, 2004) make SVMs a good option. An important trait of convex optimization problems is a guarantee that there is an only optimal model to fit the data.

In the previous literature the application of SVMs to financial data has been mostly dedicated to the prediction of future direction of stock price index. For example, the study of Kim (2003) examines the feasibility of applying SVM in financial forecasting by comparing it with back-propagation neural networks and case-based reasoning. The experimental results proved that SVM provides a promising alternative to stock market prediction.

Another example is the paper of Van Gestel et al. (2001) where SVM was used for one step ahead prediction of the 90-day T-bill rate in secondary markets and predict the daily closing price return of the German DAX30 index. The SVMs were used for regression instead of classification, and the feature vector was based on lagged returns of the index, bond rates, S&P500, FTSE and CAC40. That paper also showed that a rolling approach to select the most optimal meta parameters demonstrated better performance. The rolling approach is based on the selection of the meta parameters via using all so far available historical information.

Additional examples of SVM regression for futures index prediction are found in Tay and Cao (2001, 2002) and Cao and Tay (2003), where also was proven that SVMs provide a promising alternative to the neural network for financial time series forecasting. As demonstrated in the experiment, SVMs forecast significantly outperformed the BP network in the CME-SP, CBOT-US, CBOT-BO and MATIF-CAC40 futures and slightly better in the EUREX-BUND.

In the work of Huang et al. (2005) they investigated the predictability of financial movement direction with SVM by forecasting the weekly movement direction of NIKKEI 225 index. The empirical results showed that SVM outperforms the other classification methods such as Linear Discriminant Analysis, Quadratic Discriminant Analysis and Elman Backpropagation Neural Networks.

Kim (2003) also used SVM as a classification method to predict the direction of the markets movement. The paper emphasized the importance of the meta parameter assumptions and how the prediction performances of SVMs are sensitive to the value of these parameters.

Huang et al. (2004) developed an interesting approach, which uses fundamental data to predict credit rating. SVMs are applied as a successful classifier of the ratings for the companies. The research revealed the fact that different markets and sectors possess distinct factors for classification.

The most related work that contributed to the idea of this paper is Huerta et. al. (2013) which investigates the profitability of a trading strategy based on training SVMs to identify stocks with high or low predicted returns. This is the only paper found in the open internet space

which applies SVMs for running an investment strategy for stock market. Even though our paper is dedicated to testing SVMs in an investment strategy for cryptocurrencies, many recommendations and assumptions from Huerta et. al. (2013) had been adopted.

Another paper that contributed to the methodological and strategy implementation part is the work of Kość et. al (2018) which investigates the momentum and contrarian effects on cryptocurrency markets. The performance of investment portfolios was benchmarked against: (1) equally weighted and (2) market-cap weighted investments as well as against the B&H strategies based on (3) S&P500 index and (4) BTCUSD price. According to the results, the cryptocurrency market clearly demonstrates the existence of strong contrarian effect.

Regarding the application of SVMs on cryptocurrency market, there is no such extensive research that has been conducted if compared to the research for stock market. For example, Chen et. al. (2018) checks the accuracy of predictor models for price changes in ethereum, and the performance of SVMs has not turned to be the best.

Many of the published papers using SVMs for financial data mention the influence of the meta parameter selection on the performance of the model. It is a key issue to make sure that the selection of meta parameters is free from forward looking bias. In order to avoid this widespread problem, the meta parameters must be chosen based only on historical information.

There was always an important point for discussion concerning the choice of the type of SVM: linear versus nonlinear. Linear SVMs are fast to train and run, but this type of SVMs tend to underperform on complex datasets with many training examples. As was proven in Huerta et. al. (2013) linear SVMs resulted in inferior returns if compared to those yielded by nonlinear SVMs.

2. Theoretical background regarding SVM

Support Vector Machines (SVM) is one of the most popular algorithms for classification. It is based on the assumption that in a multidimensional space there exists a hyperplane that may separate the data into classes.

SVM can be generalized out of a simple classifier which is called the maximal margin classifier. It is only feasible to apply linear classifier to the data sets which are linearly separable. Unfortunately, most of data sets have classes which cannot be separated by a linear boundary. The maximal margin classifier was extended to the support vector classifier, which in turn can be applied to a wider range of data sets. Support vector machines is a further

extension of the support vector classifier which already can be applied for fitting data with non-linear class boundaries.

If we deal with non-linear class boundaries, the problem can be solved by extending the dimension via using quadratic, cubic or higher-order polynomial functions of the features. For example, rather than applying a support vector classifier with p features space, there can be used a support vector classifier fit via using $2p$ features space:

$$X_1,\ X_1^2,\ X_2,\ X_2^2, \ldots, X_p,\ X_p^2. \tag{2.1}$$

The optimization problem takes the following form:

$$\text{Maximize } M$$

$$\beta_0, \beta_{11}, \beta_{12} \ldots, \beta_{p1}, \beta_{p2}, \varepsilon_1, \ldots, \varepsilon_n, M$$

$$\text{subject to } y_i\ (\beta_0 + \sum_{j=1}^{p} \beta_{j1}\ x_{ij} + \sum_{j=1}^{p} \beta_{j2}\ x_{ij}^2) \geq M\,(1-\varepsilon_i), \tag{2.2}$$

$$\varepsilon_i \geq 0, \quad \sum_{i=1}^{n} \varepsilon_i \leq C, \quad \sum_{j=1}^{p} \sum_{k=1}^{2} \beta_{jk}^2 = 1.$$

So, SVM is an enlargement of the support vector classifier that realizes through extending the feature space via using kernels that are considered to be a specific efficient computational approach. A kernel is a kind of a function that is able to quantify the similarity of a pair of observations. For example, one may consider the following expression:

$$K(x_i, x_{i'}) = \sum_{j=1}^{p} x_{ij} x_{i'j}, \tag{2.3}$$

that will just returns the support vector classifier. The equation (2.3) is called a linear kernel due to the fact that the support vector classifier is linear in the predictors; the linear kernel basically quantifies the similarity of two observations applying Pearson correlation. However, it is possible to choose another form for (2.3). For example, one may substitute every output of $\sum_{j=1}^{p} x_{ij} x_{i'j}$ with the instance obtained via the following expression:

$$K(x_i, x_{i'}) = (1+ \sum_{j=1}^{p} x_{ij} x_{i'j})^{\mathrm{d}}. \tag{2.4}$$

The expression stated above is a polynomial kernel with degree d, where d is a positive integer. Applying such a kernel instead of the linear kernel as shown in (2.4), the algorithm demonstrates much more flexible decision boundary. Rather than in the initial feature space, it basically results in running a support vector classifier with a higher-dimensional space using polynomials of degree d. The resulting classifier is called a support vector machine when a non-

linear kernel is applied such as shown in (2.4). So, the non-linear function takes the form of the following equation:

$$f(x) = \beta_0 + \sum_{i=1}^{N} \alpha_i \, y_i \;\; K(x, \; x_i), \tag{2.5}$$

where

- x_i is all the vectors of the training set.
- N is the number of training examples used to fit the SVM parameters.
- α_i is a scalar, that is a real number, that takes values between 0 and C. C may be deemed as a budget defined by the number allowing the margin to be violated by the n observations. If C equals zero, it means that there is no budget for violations to the margin.
- y_i identifies whether the feature vector x_i of the object i belongs to the tail set with class + or class −.
- β_0 is obtained by training the SVM, and is a scalar that shifts the output of the SVM by a constant.
- $K(x, \; x_i)$ is the kernel function, that takes two vectors as inputs and produces a single scalar value which is positive.

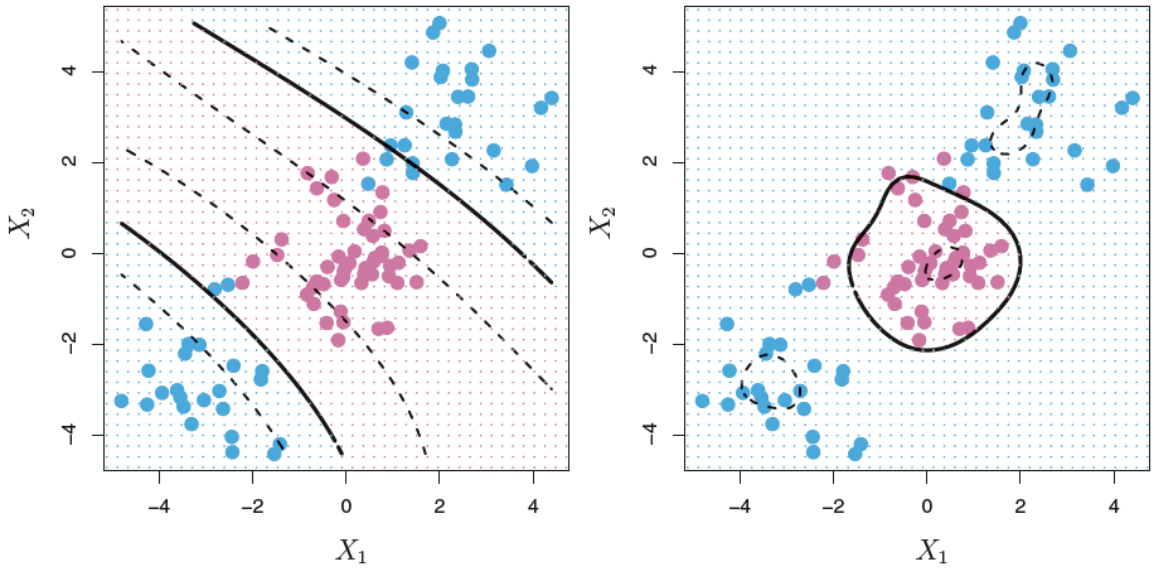
The polynomial kernel that was outlined in (2.4) serves as just one instance of a possible non-linear kernel. There exist ample alternatives. One very wide-spread alternative is the radial kernel, that has the following form:

$$K(x_i, x_{i'}) = \exp(-\gamma \; \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2). \tag{2.6}$$

In (2.6), γ (gamma) is a tuning meta parameter, that is a positive constant. Intuitively, the gamma parameter defines how far the influence of a single training example reaches, with low values meaning ‘far’ and high values meaning ‘close’.

Figure 2.1 demonstrates two examples for non-linear data where SVM is run with a polynomial kernel. So, looking on Figure 2.1, in both cases, either kernel is able to capture the decision boundary, however SVM with a radial kernel demonstrates far better fitting.

Figure 2.1. Left panel: non-linear data divided by an SVM with a polynomial kernel of degree 3. Right panel: the same non-linear data divided by an SVM with a radial kernel



Source: James, G., Witten, D., Hastie, T., Tibshirani, R. An Introduction to Statistical Learning with Applications in R.

Advantages of SVM model are such that: it is very efficient when groups are fully or almost fully separable, can work very well when there are more independent variables than observations, can be adjusted to work well with unbalanced datasets, has only just several parameters to tune and is partially immune to outliers. Disadvantages are as follows: SVM is very slow for large number of observations and it is not the most efficient when separation is low. To conclude, the type of SVMs used in the paper is the non-linear radial kernel.

3. Data, methodology and strategy implementation

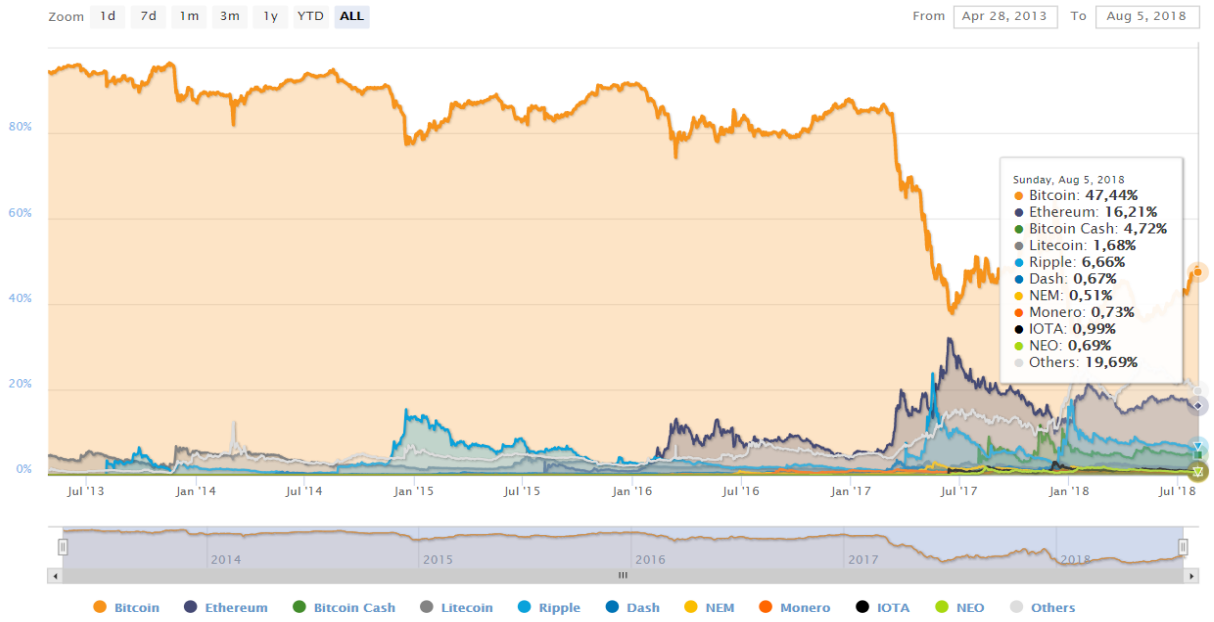
3.1. Data and filtering

The data were downloaded from coinmarketcap.com. As of 05/08/2018 there were 1732 coins listed on the website. Data include OHLC prices, Volume and MarketCap starting from 27/12/2013 and ending by 05/08/2018 as decided by authors.

As of 05/08/2018 the number of all listed cryptos totals 1732, while as of 01/12/2017 there were ~ 1500 cryptos. In such a way, despite the current downturn on cryptocurrency market, the number of cryptocurrencies is constantly growing.

The total market capitalization of the cryptocurrency market reached its peak in Jan'18 totaling almost 800 billion USD, what is equal ~ 2.5% of S&P 500 Total Market Cap[1]. Nowadays the total market cap of the cryptocurrency market is undergoing 4 times decline if compared to the peak at the end of the year. The recent decreasing trend may reflect investors' sentiment, which has been impacted by negative news concerning new market regulations and ICOs frauds. Additionally, there are some claims that the highs of 2017 may have been manipulated and artificially inflated.

Figure 3.1. Percentage of total market capitalization by dominance



Source: https://coinmarketcap.com/charts/
Note: the chart presents the dominance of the cryptocurrencies with the largest market capitalization over the period from 28/04/2013 to 05/08/2018.

As can be noticed on the Figure 3.1 till Mar'17 the market was dominated by Bitcoin (90% of the total market cap of cryptocurrency market). In 2017-2018 Bitcoin is losing its position and allows other assets to share the market. As of 05/08/2018 the share of Bitcoin comprises 47,44% of the total market, Ethereum – 16,21%, Ripple – 6,66%, Others – 19,69%.

14-day moving average of volume was calculated for each asset, and those which do not meet the filter threshold of 100 USD are excluded from the further usage. Such application of filtering ensures that the investment portfolio meets the minimum liquidity requirement.

[1] Estimates based on data provided by: www.coinmarketcap and World Bank: data.worldbank.org/indicator/CM.MKT.TRAD.CD. USA stock market cap for 2017 equals 32.121 trillion USD.

Additional filter is applied regarding the price history of an asset. Only those cryptocurrencies that have 91 days and more of history for the close price are qualified to constitute the data set.

After that, there was created a ranked set of 100 cryptocurrencies by the largest market cap. This set of 100 cryptocurrencies with the largest market cap will be referred further in the thesis to as the Top100.

The starting date for the simulation period was taken as 01/10/2014 so to ensure the availability of at least 3-month historical data to create training set and start running the strategy on 01/01/2015. The ranking with 100 largest cryptocurrencies (Top100) was calculated for each day of the simulation. From the set of 1438 unique assets only 475 have been qualified to enter the Top100 ranking for at least one day, as well as having taken into consideration the condition of the 14-day moving average of daily volume being higher than 100 USD.

3.2. Performance statistics

The following measures were used to provide descriptive statistics for the data and further these measures were applied in evaluation of portfolios' efficiency:

- ARC (the annualized rate of change):

$$\text{ARC} = \left(1 + \frac{P_T}{P_0}\right)^{\frac{365}{T}} - 1 \tag{3.1}$$

where P_T stands for the portfolio value after the T-th period.

- ASD (the annualized standard deviation of daily returns):

$$\text{ASD} = \sqrt{\frac{365}{T} \sum_{t=1}^{T} (r_1 - \bar{r})^2}, \tag{3.2}$$

where $r_t = \frac{P_t}{P_{t-1}} - 1$ and $\bar{r}$ is an average return calculated as the simple mathematical average of a series of returns generated over one year.

- MDD (the maximum drawdown coefficient):

$$\text{MDD}(T) = \max_{\tau \in [0,T]} (\max_{\tau \in [0,T]} P_t - P_\tau) \tag{3.3}$$

- IR1, IR2 (the information ratio coefficients) quantify the risk-weighted gain:

$$\text{IR1} = \text{ARC} / \text{ASD} \tag{3.4}$$

$$\text{IR2} = \text{sign(ARC)ARC}^{\,2} / (\text{ASD*MDD}) \tag{3.5}$$

Table 3.1 presents descriptive statistics for the 10 largest and 10 smallest cryptocurrencies as of 01/01/2018. As can be seen from Table 3.1, the values of %ARC, %ASD, IR and IR2 have huge difference across the cryptocurrencies. Quite a lot of new assets such as loom-network, cybermiles, nuls bibox-token and odem appeared just a couple of months ago, but already took the position in Top100. For example, odem behaves as an investment "star" demonstrating abnormal return and the lowest drawdown. This crypto was built on the Ethereum blockchain and stands for On-Demand Education Marketplace.

The values for the maximum drawdown are also relatively large. If odem is not taken into consideration, other cryptocurrencies in the set have %MDD in the range from 50% to 99%. For comparison purposes − the S&P500 index has noted only ~ 14% drawdown in the same simulation horizon (see Table 4.1. in the next section).

Table 3.1. Descriptive statistics for 10 largest and 10 smallest cryptocurrencies by MarketCap in TOP100 as of date 2018-08-01

The largest 10 cryptocurrencies in TOP100 as of 2018-08-01

Name	%ARC	%ASD	%MDD	IR1	IR2	Date of start	Volume, mUSD	MarketCap, USD
bitcoin	118	75.8	69.7	1.6	2.6	2014-10-01	1888	43839225862
ethereum	437.7	145.2	84.3	3	15.7	2015-08-20	323	17124399552
ripple	226.7	164.6	87.1	1.4	3.6	2014-10-01	499	13468236361
bitcoin-cash	83.5	198.5	84.4	0.4	0.4	2017-08-05	699	6672807179
eos	516	253	87.9	2	12	2017-07-14	78	5220519698
stellar	228.6	178.8	82.6	1.3	3.5	2014-10-01	301	4634665748
litecoin	111.1	119.2	79.1	0.9	1.3	2014-10-01	80	3709789644
cardano	698.6	263.8	89.3	2.6	20.7	2017-10-14	32	2624893338
iota	48.4	188.4	82.9	0.3	0.1	2017-06-26	3059	2460207729
tether	-5.4	45.7	49.9	-0.1	0	2015-03-15	140	2233258238

The smallest 10 cryptocurrencies in TOP100 as of 2018-08-01

Name	%ARC	%ASD	%MDD	IR1	IR2	Date of start	Volume, mUSD	MarketCap, USD
loom-network	649.7	217.4	80	3	24.3	2018-04-21	2.8	98040413
gas	365.8	265.4	89.6	1.4	5.6	2017-08-09	2.6	91875052
tenx	-97.6	247	99.2	-0.4	-0.4	2017-07-10	8.1	91154578
nxt	34.2	158.1	95.6	0.2	0.1	2014-10-01	2.8	90165499
cybermiles	-44.8	202.2	87.4	-0.2	-0.1	2018-05-04	7.1	88375828
nuls	5083.5	382.2	79.1	13.3	854.8	2018-03-22	4.1	88067102
byteball	160	236.8	91.2	0.7	1.2	2017-01-09	0.56	86950232
bibox-token	53.2	265.7	87.9	0.2	0.1	2018-06-08	67.5	83456610
odem	89471	229.6	40.7	389.7	856638	2018-08-01	0.135	82906522
electroneum	-92.5	243	95.3	-0.4	-0.4	2017-11-15	0.552	81946739

Legend: %ARC − annualized rate of return as in (3.1), %ASD − annualized standard deviation in percent as in (3.2), %MDD − maximum drawdown of capital in percent as in (3.3), IR1, IR2 − information ratios as in (3.4) and (3.5) accordingly, "Date of Start" − date of first appearance in Top100.

3.3. Construction of the training data

One of the key issues in this investigation is how to form the tail sets that constitute the positive and negative classes of the training data. As a metric there were chosen returns divided by a volatility estimate. This option creates an ordered list of coins with volatility-adjusted returns. The volatility is estimated here by an exponential moving average:

$$\begin{aligned} \hat{\sigma}_0^2 &= 0 \\ \hat{\sigma}_1^2 &= \lambda\hat{\sigma}_0^2 + (1-\lambda)r_0^2 \\ \hat{\sigma}_2^2 &= \lambda\hat{\sigma}_1^2 + (1-\lambda)r_1^2 \\ &\vdots \\ \hat{\sigma}_{t+1}^2 &= \lambda\hat{\sigma}_t^2 + (1-\lambda)r_t^2 \end{aligned} \tag{3.6}$$

The volatility-adjusted return is the ratio of the daily return and the estimation of the volatility according to formula 3.6:

$$adj\ R_t = \frac{R_t}{\hat{\sigma}_t} \tag{3.7}$$

where R_t is a daily return, $\hat{\sigma}_t$ is the exponential moving average estimate of the volatility calculated as shown in (3.6) with λ set to 0.94.

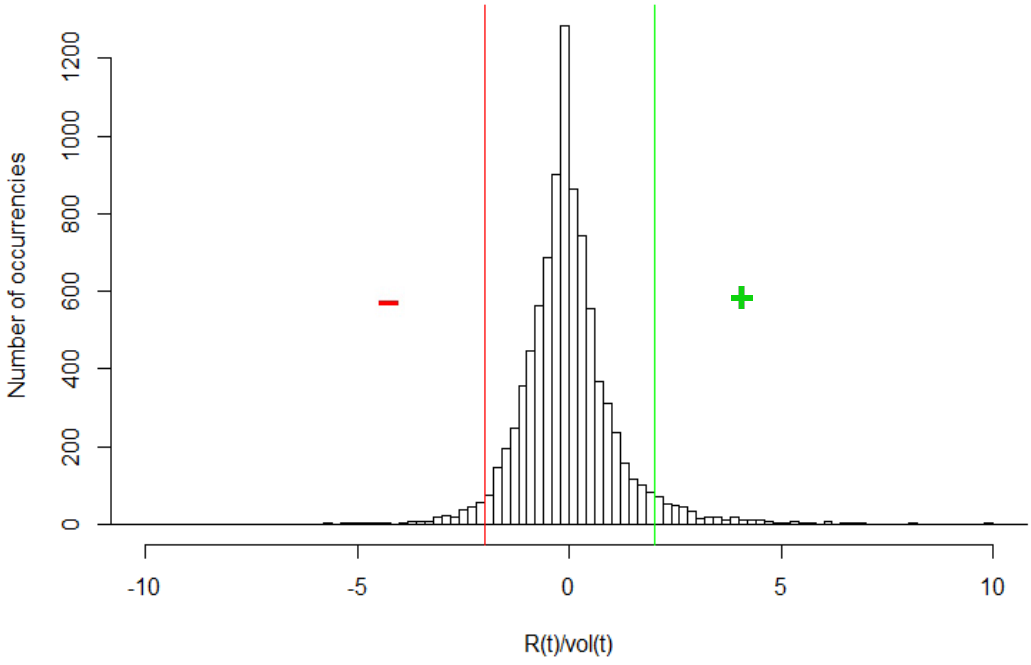
It was shown in Huffman et. al. (2011) that computational load is higher when the volatility is calculated using standard deviations. Moreover, the results revealed no significant difference between these two methods for volatility estimation. Therefore, the way to calculate the volatility-adjusted return as presented in (3.7) is the preferred option.

In order to run calculations for each reallocation period the volatility-adjusted returns for 3-month period (91 days) for the assets in Top100 for each day were calculated. Long data were filtered by the condition to ensure the presence of 91 days of history. Then for each day in the range of these 91 days there were calculated discrete daily returns. Using these daily returns, estimate of the volatility (an exponential moving average) was calculated according to the algorithm presented in (3.6).

As can be seen, first sigma is 0. Lambda was set to 0.94 by default as a value more frequently used for exponential moving average computations. Then the volatility adjusted returns are calculated according to (3.7). As a result, for each day for each coin in Top100 the volatility-adjusted returns are calculated for 3-month period (91 days).

For illustration purposes, the distribution of volatility-adjusted returns with 3-month history from 01/10/2014 to 12/31/2014 is shown in Figure 3.2. As an example, the positive tail sets are the B most positive volatility-adjusted returns, and the negative tail sets are the B most negative. B can be 5, 10 or 15 coins in the data set. The values for the training size will be provided later in this chapter in terms of sensitivity analysis. The vertical lines represent the cut-off indicating how many assets will be used for SVM training. The + and − regions are the ones used for that. In such a way, to train SVM for 01/01/2015 there will be used 3-month data from 01/10/2014 to 12/31/2014 with volatility-adjusted returns, for 01/02/2015 − data from 02/10/2014 to 01/01/2015.

Figure 3.2. Distribution of volatility-adjusted returns with 3-month history from 01/10/2014 to 12/31/2014 with cut-off lines to identify the number (training size) of the assets with the highest (green) and the lowest (red) volatility-adjusted returns to use for SVM training



Source: own work.
Note: the distribution of 91-days volatility-adjusted returns where the positive tail sets are the most positive volatility-adjusted returns, and the negative tail sets are the most negative. The vertical dotted lines represent the cut-off. The + and – regions are the ones used for SVM training.

Further in the paper TS will be denoted as the length for training set, or in other words the number of volatility-adjusted returns used in designing the training data sets.

3.4. Selection of technical features

Each coin on day t should be characterized by a vector of technical features $x_i(t)$. Features are chosen based on their popularity in the academic literature. The list of features used is presented in Table 3.2.

Table 3.2. Technical indicators used for creation of feature set to train SVMs

Feature	Full name	Parameters
MOM, n days	Momentum for close prices, n days	n = 10 days
ΔV, n days	Volume change n days	n = 10 days
RSI	Relative Strength Index	n = 10 days
FI	Force Index	N/A
Williams %R	Williams Percent Range	n = 10 days
PSAR	Parabolic stop and reversal system	Acceleration factor by default set to 2% increasing by 2% with a maximum of 20%

Note: the table contains the list of six technical features used for running SVM and parameters set for calculations purposes.

Momentum has been one of the well-recognized phenomenon described in the academic literature; see Jegadeesh et al. (2012) and Rouwenhorst (2002). They stated that stocks with high (low) returns over periods of three to 12 months keep having high (low) returns over subsequent three to 12 month periods. So, Momentum for close prices of cryptocurrencies was included in the feature set.

Another feature, the volume change, is an indicator catching underreactions and overreactions in stock price movements. If a price movement happens with large volume, the price change is more significant than if it occurs with low volume; see Chordia et al. (2002). To capture this effect, a percentage change of the daily trading volume was included in the feature data set.

RSI is a momentum indicator that captures the magnitude of the latest price changes in order to estimate whether the market is overbought or oversold. Thus, it is predominantly used to identify overbought or oversold market conditions. RSI is calculated as follows:

$$RSI = 100 - 100 / (1 + RS), \quad (3.8)$$

where RS is an average gain of up-trending periods during the certain time period divided by an average loss of down-trending periods during the defined time period. In such a way, RSI provides a relative estimation of the strength of an asset's recent price performance. RSI outcomes range from 0 to 100. The default time period for relating up-trending periods to down-trending periods is 2 weeks. Traditional interpretation of RSI is that RSI outcomes of 70 or

above show that an asset is getting overbought or overvalued. RSI being 30 or below interpreted as pointing out oversold or undervalued conditions. Sudden significant price changes may provide false buy or sell signals. Therefore, it is better to use it with amendments to its application or together with other reliable technical indicators.

FI captures the market power behind the change in the asset price. FI's value may be both positive or negative, what depends on the upward or downward change in the asset price. The three inputs required for the formula are: close price, open price and trading volume. Analysts often use FI along with the moving average to make predictions for an asset's future performance. The formula for FI is as follows:

$$FI = (\text{Close price} - \text{Open price})*\text{Volume}. \tag{3.9}$$

Williams%R is a type of momentum indicator that ranges between 0 and -100 and measures overbought and oversold market conditions. The Williams%R is commonly applied to define entry and exit points for trading. It can be treated as a technical analysis oscillator. It compares an asset's close price to the high-low range over a specific period, by default over 14 days. The formula to calculate this indicator is as follows:

$$\%R = (\text{highest high} - \text{close price}) / (\text{highest high} - \text{lowest low}) * (-100) \tag{3.10}$$

The Williams%R became popular as an indicator because of its ability to signal for market swings at least one or two points in the future. Predictions of market reversals are very valuable for market participants, so an asset is overbought when the indicator is above -20, and is oversold when the indicator is below -80. Overbought and oversold periods can persist, should the price keep on rising or falling.

The parabolic SAR is a technical indicator applied to define the price direction of an asset, as well as to measure how the price direction is changing. It is generally used to put trailing price stops, thus it may be treated as a stop-loss system.

PSAR is calculated independently for each trend in the price. When the price is in an uptrend, PSAR goes below the price and converges upwards towards it. By the same logic, on a downtrend, PSAR goes above the price and converges downwards. At each step within a trend, PSAR is calculated one period in advance. PSAR value for tomorrow is calculated using data available today. PSAR values are calculated as follows:

$$PSAR = PSAR_{n+1} = PSAR_n + (AF * (EP - PSAR_n)) , \tag{3.11}$$

where EP is the highest high for a long-term trend and the lowest low for a short-term trend, which is updated each time a new EP is achieved; AF is the default of 2% increasing by 2% each time a new EP is achieved, with a maximum of 20%.

In the work of Pistole (2010), moving average rules, RSI method and PSAR technique were compared with the buy&hold strategy for the S&P500 index. Interestingly, PSAR indicator is far more successful in positioning long or short on the S&P500 index. Results revealed that by applying PSAR as the buy and sell signal, the strategy significantly outperforms the market, as well as buy&hold and other strategies discussed in that paper.

The above mentioned, six technical indicators were considered and included in feature set mostly based on the academic literature. All the indicators were calculated in R. RSI, SAR, Williams %R were implemented using the corresponding built-in functions from TTR package.

3.5. Portfolio construction and training protocol

The data sets which should be prepared in advance are long data containing all the information available and the Top100 market cap ranking. These data sets will be used for running the SVM strategy. A step-by-step loop was written in R to implement the strategy.

Volatility-adjusted returns were calculated for the assets in Top100 for the 3 previous months for each day, more specifically over the period from $date_{[t]-92}$ to $date_{[t]-1}$. Then these returns were ranked in descending order. For each day there should be 100 coins with volatility-adjusted returns. These volatility-adjusted returns are serving as a class for SVMs.

The length of training set is introduced as the parameter TS. To construct the training set TS is used to denote the number of volatility-adjusted returns. For example, TS on the level of 25 means that 25 coins with the highest values and 25 coins with the lowest values of volatility-adjusted returns are taken to form the training data. The assets with the highest values form the class + and the assets with the lowest values form the class −. Then for these 50 coins the six technical features are calculated (MOM, V, IF, PSAR, Will%R, RSI) for each day for the period of the last 91 days. The period over which the training set is constructed to perform one SVM test on the reallocation day ($date_{[t]}$) is from $date_{[t]-92}$ to $date_{[t]-1}$.

When training set is ready, SVM is applied in order to tune the meta parameters C (cost) and γ (gamma). The best C and γ are used later to test SVM for the assets on the reallocation day. The choice of meta parameters are explained in the section 3.7.

The first reallocation day is assumed to be 01/01/2015. The reallocation day is the day on which the portfolio composition is changed. The change depends on the SVM output. The assets with the greatest values of the SVM output are included in the portfolio on the reallocation day. SVM output is the value provided by the function as in (2.5)

On the reallocation day, we liquidate the assets that are not recommended by SVM output and they leave the portfolio. The assets that remain in the portfolio have their weights reallocated. The time period which is between two sequential reallocation days is called the reallocation period (RE).

Then testing set is prepared. SVM is tested on the data on the $date_{[t]}$. Even though, the system is trained for a particular train data size TS, all the assets which are on the list of Top100 on the reallocation day are considered for testing. The technical features are calculated for them and SVM is tested on the whole data set. SVM output provides the function value for each asset in the range varying around -1 and 1 (due to assigning 1 for positive class and -1 for negative class). The output is ranked and those assets with the highest values are qualified to become "buy" candidates and to be included in the portfolio. In such a way, every time as a portfolio is reconsidered on the reallocation day, a fresh SVM model is trained and tested in order to capture the changes on the market. It is worth noting that the function svm() scales the data by default.

To summarize, the training protocol is such that SVM is trained over tail sets for the period of time [t − 92, t −1] and tested for the period [t]. As the data from coinmarketcap.com are available on the daily basis and without breaks for weekends, the above mentioned protocol is applied over the whole simulation period from 01/01/2015 to 08/01/2018.

In order to estimate the portfolio performance the following methodology was used. The gross rate of return $R_{0,T}^{(P)}$ for a given portfolio P in the period $t \in [0, \ldots, T]$ is calculated as:

$$R_{0,T}^{(P)} = \Pi_{t=0}^{T} \left(1 + \sum_{i=1}^{N} w_{i,t}\, r_{i,t} - \Delta W_t^R * TC\right) - 1 \qquad (3.12)$$

where N is the total number of cryptos; T is total time horizon for the investment; $r_{i,t}$ is the accruing daily rate of return of the i-th asset on day t; $w_{i,t}$ is the weight of the i-th asset in the whole portfolio Π on day t; ΔW_t^R is the total portfolio turnover rate in percent on day t; TC is the total transaction costs in percent.

The weights are being calculated according to the following formula

$$w_{i,t} = \left(1 + r_{i,t}\right) w_{i,t-1} \qquad (3.13)$$

It is assumed that $w_{i,t}$ sums up to unity for each reallocation day. On each reallocation day $t = t_R$ the weights are reallocated in the following way:

$$w_{i,t_R} = \begin{cases} \frac{1}{N} & \longrightarrow \text{weights for SVM and equally weighted portfolio} \\ \frac{MC_{i,t}}{\sum_i^N MC_{i,t}} & \longrightarrow \text{weights for market-cap weighted portfolio} \end{cases} \quad (3.14)$$

The portfolio composition changes on the reallocation day.

In order to understand the logic of the formula for calculation of the turnover ratio of the portfolio three cases were taken into consideration. First one is that assets leave the ranking. Second one is that assets enter the portfolio, and third one, assets keep staying in the portfolio, just with new weights. To account for this change in portfolio composition the turnover ratio of the portfolio was calculated as follows:

$$\Delta W_{t_R}^R = \sum_{i=1}^N \left| w_{i,t} - w_{i,t_R} \right| \quad (3.15)$$

The above value can be of any in the range from zero (the composition of the portfolio is not changed if compared to the previous reallocation day) to 200% (the composition of the portfolio is entirely changed, all assets left the portfolio and new ones entered).

The values of transaction costs on the cryptocurrency markets can be between 0.2% and 2.0% of the transaction value depending on the asset's type and the liquidity. Transaction costs for the base and the benchmark portfolios were assumed to be 1%. To provide the fair calculation for the total portfolio reallocation cost, the product of the portfolio turnover ratio $\Delta W_{t_R}^R$ and the total percent transaction cost TC is taken.

The implementation of a portfolio loop in R was done with the package Performance Analytics. That vignette gives some simple examples of computing portfolio returns using asset prices as well as weights framework.

3.6. Benchmark portfolios

To construct the benchmark portfolios there was used the Top100 market cap ranking. To estimate the efficiency of the main SVM strategy, similarly like in Kość et. al. (2018), the benchmark portfolios were chosen as follows:

- The benchmark equally weighted portfolio (further denoted as EqW) is constructed as an investment with equal weights in all cryptocurrencies which are qualified for Top100 on the reallocation day. As for a base case, the reallocation period is set to one week.

Transaction costs constitute the 1% of the portfolio value. These assumptions are the same as for the SVM strategy.

- The benchmark market-cap weighted portfolio (further denoted as McW) is built as an investment with market-cap weights in cryptocurrencies which are qualified for in TOP100 on the reallocation day. It means that on reallocation day the investment in an asset takes the weight as the ratio of the market capitalization of an asset to the whole market capitalization for all assets in Top100. The reallocation period and the transaction costs are the same as for the equally-weighted portfolio.
- Buy&hold strategy is also considered as a benchmark portfolio. Just to get the full insight on the performance of all the portfolios two buy&hold strategies were run, one is for bitcoin (BTC B&H), and the second is for S&P500 index (S&P B&H). It is worth noting that buy&hold strategy on S&P500 index is a widely used benchmark to compare the portfolios. Both buy&hold strategies have the same simulation period as the former benchmark strategies.

3.7. Selection of meta parameters

The C (cost) and γ (gamma) values are the meta parameters of the SVM model. One of the problem that frequently arise is overffiting, which is quite common in machine learning (Cawley et. al. 2010). That is why the question of choice of meta parameters is a major one. The meaning of meta parameters was described in the theoretical part in the section 2. As a short reminder, the general meaning of the C (cost) and γ (gamma) is as follows.

High values for C causes the cost of misclassification to be large, therefore SVMs are forced to classify the input data more severely and the problem of overfitting may arise. Small values for C means lower variance and higher bias. Small values for C makes the cost of misclassification low, thus providing more “space” for the model to make a mistake by misclassifying the case. The objective is to find the balance between “not too severe” and “not too relaxed”.

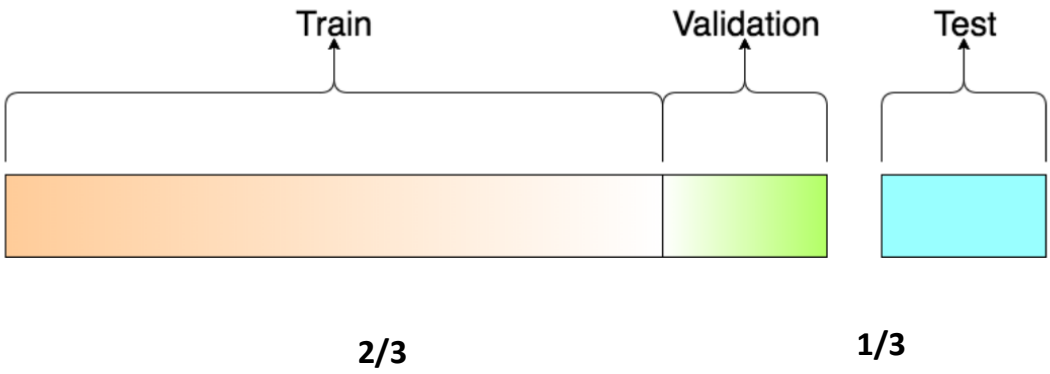
When the value for gamma is small, the constraint for the model is too high and it cannot catch the complexity or curvature of the input data. In other words, gamma explains how strong the influence of a single training observation.

As in a standard classification problem, dataset is divided into training and testing sets which are mutually exclusive sets. Further in order to perform tuning of meta parameters the training set is separated into again training set and validation set. The visualization of

the data set splits is presented on Figure 3.3. Therefore, in order to perform the tuning only second training and validation sets are used.

The partition of the training set into again training and validation sets is performed with the help of certain sampling method. Potentially, there can be a number of partitions into validation and training sets over which tuning is performed.

Figure 3.3. Visualization of the data set splits and their proportions.



Note: the figure represents in which proportion data is split to run the tuning of eters.

In the package e1071 tune.control() option available within built-in function tune.svm() is presented. Tune.control offers three sampling methods on how the train data set may be split. These methods are "cross", "fix" and "bootstrap". Due to the fact that performing cross-validation or bootstrap over the full data set has very high computational load and takes unreasonably long time for the user to get the results, sampling method “fix” was used to tune the parameters. So, sampling method set to “fix” means that a single split into train/validation set is used and the train set contains a fix part of the supplied data. By default the proportion of the train set is 2/3.

As the algorithm of SVM is a standard classification task which has only two parameters, grid search method is considered quite effective. To conduct greed search over the parameters function tune.svm() from the package e1071 was used. A sequence of parameters for cost and gamma was created as vector (0.5, 1, 2, 4). Each pair of the parameters from that sequence are tested and those values of cost and gamma providing the lowest prediction error for the model on the validation subsets are chosen.

In such a way, every time SVM is tested on the reallocation $date_{[t]}$, first the training set is used to tune the parameters and chose the optimal ones. Then these best cost and gamma are used for testing SVM. In such a way, there can be different set of optimal parameters for each reallocation period. So, there is its own set of optimal parameters for each reallocation period.

3.8. Summary for strategy implementation: step-by-step actions

This section summarizes step-by-step actions to implement the SVM strategy.

Here are the one-time actions conducted before running the loop to generate strategy results:

1. Web-scrapping: the data were scrapped from the website coinmarketcap.com. starting from 27/12/2013 and ending by 05/08/2018. They were not provided in any "friendly" format so we had to scrap the from html source of the website.
2. Filtering: 14-day moving average of volume was calculated for each asset, and those which did not meet the filter threshold of 100 USD were excluded from the further usage. Additionally, only those cryptocurrencies that have 91 days and more of history for the open price were qualified to constitute the long data set.
3. Top100 ranking: a ranked set of 100 cryptocurrencies by the largest market cap was created for each day for the whole long data set.

A step-by-step loop written in R to implement the strategy is run as many times as the number of reallocation days in the period from 01/01/2015 to 01/08/2018. Step-by-step actions in the loop are as follows:

1. Preparation of the training set: class for SVM. On $date_{[t]}$ volatility-adjusted returns are calculated for the assets in Top100 for the 3 previous months for each day, more specifically over the period from $date_{[t]-92}$ to $date_{[t]-1}$. The returns are further ranked in descending order. For each day there should be 100 coins with volatility-adjusted returns calculated for them. Volatility-adjusted returns are serving as a class for SVMs. The assets with the positive values of volatility-adjusted returns are assigned the class + and the assets with the negative values are assigned the class −.
2. Preparation of the training set: technical features. We select the assets with the highest and the lowest volatility-adjusted returns from the spectrum defined by %TS assumption and then calculate for them six technical features (MOM, V, IF, PSAR, Will%R, RSI) for each day for the period of the last 91 days.
3. Meta parameters tuning. SVM is applied on the training set in order to tune the meta parameters C (cost) and γ (gamma). The best C and γ are used later to test SVM to predict the class for the assets in the testing set on the reallocation day.
4. Preparation of the testing set. On $date_{[t]}$ all the assets which are on the list of Top100 on the reallocation day are considered for testing. Six technical features are calculated (MOM, V, IF, PSAR, Will%R, RSI) for 100 cryptocurrencies from Top100 on $date_{[t]}$.

5. SVM is run using prepared training and testing sets. The output of the SVM for 100 assets from Top100 on $date_{[t]}$ is ranked and those assets with the highest values are qualified to become "buy" candidates.
6. "Buy" candidates are then kept in the portfolio for the reallocation period (for example, it is 1 week for the base case).
7. Calculation of the net portfolio value for one reallocation period taking into consideration transaction costs.

Once the loop is finished, performance statistics are calculated for the net value of the portfolio. In such a way, every reallocation period a fresh SVM model is trained and tested with its own optimal meta parameters.

The parameters which are deemed to be fixed in the model are as follows:

- the number of periods for volatility-adjusted returns calculated as daily returns.
- lambda λ (set to 0.94) used to calculate exponential moving average for returns.
- length of historical data taken to calculate technical features described in Table 3.2.
- meta parameters C and γ.
- length of training data set to 91days.
- long positions only assumptions.

Four parameters were chosen to participate in the sensitivity analysis. These parameters are:

- the number of cryptocurrencies kept in the portfolio (N).
- reallocation period (RE)
- percentage value of the transaction costs (%TC).
- training data size (%TS).

4. Empirical results

4.1. Performance of the SVM strategy in comparison to benchmark strategies

The performance statistics for SVM and benchmark strategies are presented in Table 4.1.

Table 4.1. Descriptive statistics of SVM strategy compared with the benchmark strategies

	N	RE	%TC	V	%ARC	%ASD	%MDD	**IR1**	IR2	%MT
S&P B&H	-	-	-	-	13,6	15,5	14,2	**0,9**	0,8	-
BTC B&H	-	-	-	-	147,4	76,8	69,7	**1,9**	4,1	-
EqW	100	1 w	1	100	425,8	96,2	81,7	**4,4**	23,1	10,8
McW	100	1 w	1	100	141,9	74,9	73,1	**1,9**	3,7	6,3
SVM	25	1 w	1	100	173,6	103,1	83,1	**1,7**	3,5	143,7

Legend: McW – market cap weighted strategy, EqW – equally weighted strategy, N - number of currencies to be invested/used to construct portfolio, RE – the width of the reallocation period between the portfolio reallocation days, %TC - the total transaction costs taken as the percentage of the total transaction value of the portfolio, V – the threshold value (USD) of the 14-day moving average of daily volume, %ARC − annualized rate of return, %ASD − annualized standard deviation in percent, %MDD − maximum drawdown of capital in percent, IR1, IR2 − information ratios, %MT – the mean portfolio turnover ratio in percent.

Buy&hold strategy on bitcoin (BTC B&H) demonstrates more than 10 times larger %ARC than buy&hold on S&P500 index (S&P B&H), however both the risk and maximum drawdown in terms of %ASD and %MDD are approximately 5 times larger. The resulting values of IR1 and IR2 are 2 and 5 times larger for BTC B&H respectively.

%ARC of B&H BTC is higher than the market cap weighted strategy (McW), although the difference is not relatively high. It can be explained intuitively as the predominant part of the McW portfolio consists of bitcoin. The values of information ratios IR1/IR2 are also comparable what means that the amount of return per unit of risk is the same as bitcoin dominates the McW portfolio.

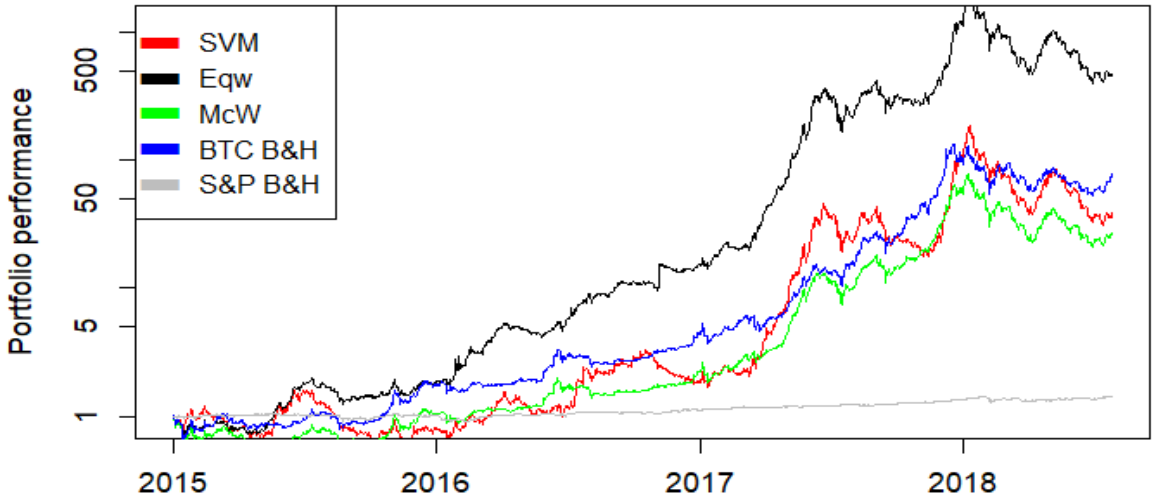
EqW outperforms all the benchmark strategies and also SVM strategy. It gives the highest values of ARC and IR1/IR2 demonstrating abnormal returns. Therefore, sorting the performance of portfolios according to IR1, the strategy with the highest value is EqW. EqW outperforms SVM more than two times as well as the other benchmark strategies. The sequence of other strategies according to IR1 is as follows: BTC B&H, McW and SVM is on the fourth place outperforming only S&P B&H.

The main hypothesis that the investment strategy based on SVMs algorithm outperforms benchmark strategies can be rejected based on these results. SVM portfolio with the long positions only gained the fourth result according to IR1 after EqW, McW and BTC B&H. Additionally, it is the most risky one according to the value of %ASD and %MDD, meaning

that SVMs algorithm selects the cryptocurrencies which are relatively volatile. Additionally, the mean portfolio turnover for SVM strategy is 14 times larger than for EqW strategy that is the reason for high transaction costs and consequently lower portfolio net value. Overall, one may invest equal weights into Top100 cryptocurrencies, incur no additional costs of implementing more sophisticated strategy and yet get abnormal returns on cryptocurrency market in comparison to simple B&H or more sophisticated strategies.

Plots of the equity lines and drawdowns for SVM strategy in comparison to the benchmark strategies can be found on Figure 4.1 and Figure 4.2 respectively.

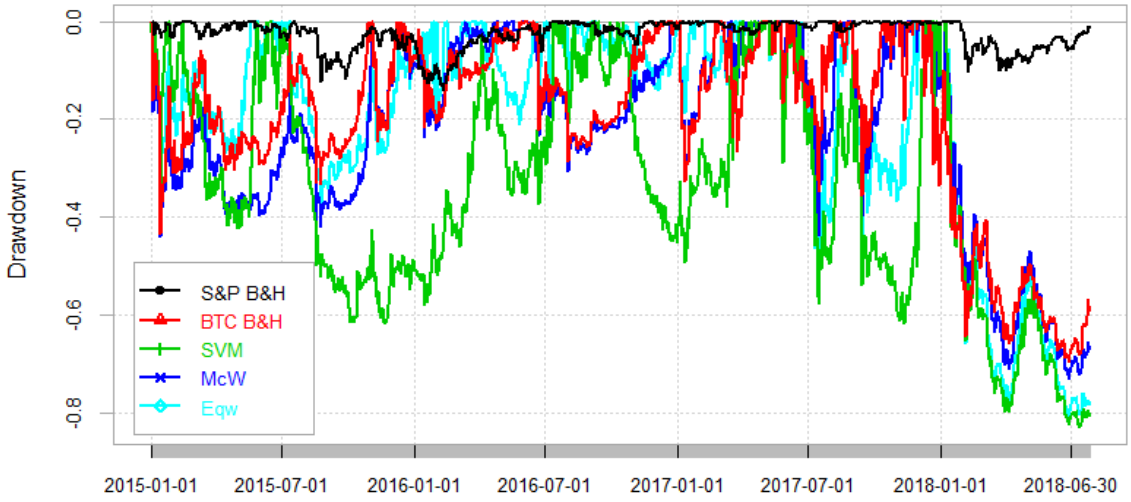
Figure 4.1. Equity line of the SVM strategy in comparison with the benchmark strategies



Note: the graph shows equity lines of SVM strategy and four benchmark strategies over the period from 01/01/2015 to 01/08/2018. EqW with equity line drawn in black outperforms all the benchmark strategies and also SVM strategy.

As can be seen on Figure 4.2, SVM and EqW strategies reaches the "deepest" drawdown if compared to other strategies. Conversely, S&P B&H demonstrates the most stable returns.

Figure 4.2. Drawdowns of the SVM strategy in comparison with the benchmark strategies
Note: the graph shows the drawdown lines for SVM strategy and four benchmark strategies over the period from



01/01/2015 to 01/08/2018. SVM strategy drawn in green reaches the "deepest" drawdown line if compared to other benchmark strategies.

4.2. Sensitivity analysis

The research questions of this study were formulated around the sensitivity analysis, namely how sensitive are the results of portfolio performance to the model parameters. The sensitivity analysis of SVM strategy is performed for the following four parameters:

- number of cryptocurrencies kept in the portfolio N = 5, 10, 15, 20, 25, VAR. VAR means that any number (between 0 and 100) of cryptocurrencies selected by SVM output for buying are included in the portfolio. This number varies from one reallocation period to another.
- reallocation period RE: 3d (3 days), 1w (7 days), 1m (30 days).
- percentage value of the transaction costs TC: 0.5%, 1.0%, 2.0%.
- training data size TS: ~ 25%, ~ 50%, ~100%.

The parameters that were set as fixed are the following:

- length of historical data taken to calculate technical features: 10d (10 days).
- lambda λ used to calculate exponential moving average for returns: 0.94.
- meta parameters C and γ are being chosen for each reallocation period via tuning algorithm. These parameters are sequenced as follows: (0.5, 1, 2, 4). There can be different set of optimal parameters for each reallocation day and it is not overseen. The choice of meta parameters is described in the section 2.6.

- length of training data: 3 months (91 days).
- long positions only assumptions.

The fixed parameters are kept constant according to the assumptions of the author. Only four parameters are chosen to participate in the sensitivity analysis: reallocation period RE, percentage value of the transaction costs TC, number of cryptocurrencies kept in the portfolio N and training data size TS. Descriptive statistics for SVM strategy and the performance of the portfolios are presented in Table 4.2. At the end of this table we additionally attach the best selected set of parameters for the SVM strategy which demonstrated the performance of the strategy with the highest IR1.

The base case for SVM strategy is presented in Table 4.2 together with benchmark strategies. As a reminder, the parameters for the base case are as follows: N = 25, RE = 1w, TC = 1%, TS ~ 50%.

Table 4.2. Descriptive statistics for SVM strategy (sensitivity analysis). Descriptive statistics for the benchmark strategies have been placed above for convenient comparison

Benchmark Strategies						
Name	%ARC	%ASD	%MDD	IR1	IR2	%MT
S&P B&H	13,6	15,5	14,2	0,9	0,8	
BTC B&H	147,4	76,8	69,7	1,9	4,1	6,3
EqW	425,8	96,2	81,7	4,4	23,1	10,8
McW	141,9	74,9	73,1	1,9	3,7	6,3
SVM	**173,6**	**103,1**	**83,1**	**1,7**	**3,5**	**143,7**

Parameters					SVM Strategy					
N	Position	%TS	RE	%TC	%ARC	%ASD	%MDD	IR1	IR2	%MT
25	long only	50	*3d*	1	19,4	108,7	90,6	0,2	0,0	115,3
25	**long only**	**50**	***1w***	**1**	**173,6**	**103,1**	**83,1**	**1,7**	**3,5**	**143,7**
25	long only	50	*1m*	1	224,2	101,5	86,0	2,2	5,8	148,8
5	long only	50	1w	1	-21,8	142,2	95,1	-0,2	0,0	189,3
10	long only	50	1w	1	89,3	131,7	85,0	0,7	0,7	176,8
15	long only	50	1w	1	207,2	115,7	82,0	1,8	4,5	166,2
20	long only	50	1w	1	215,9	110,0	82,3	2,0	5,1	154,3
25	**long only**	**50**	**1w**	**1**	**173,6**	**103,1**	**83,1**	**1,7**	**3,5**	**143,7**
VAR	long only	50	1w	1	326,4	92,6	57,6	3,5	20,0	105,6
25	long only	*100*	1w	1	177,9	103,3	85,1	1,7	3,6	144,3
25	**long only**	***50***	**1w**	**1**	**173,6**	**103,1**	**83,1**	**1,7**	**3,5**	**143,7**
25	long only	*25*	1w	1	210,6	103,6	85,5	2,0	5,0	160,5
25	long only	50	1w	*0,5*	368,8	110,2	76,5	3,3	16,1	155,4
25	**long only**	**50**	**1w**	***1***	**173,6**	**103,1**	**83,1**	**1,7**	**3,5**	**143,7**
25	long only	50	1w	*2*	29,6	110,9	88,1	0,3	0,1	154,9

Best performance of SVM strategy with selected set of parameters										
N	Position	%TS	RE	%TC	%ARC	%ASD	%MDD	IR1	IR2	%MT
VAR	long only	50	1m	1	392.43	88.97	53.45	4.41	32.38	105.9

Legend: McW – market cap weighted strategy, EqW – equally weighted strategy, N – number of currencies to be invested/used to construct portfolio, %TS – training data size, RE – the width of the reallocation period between the portfolio reallocation days, %TC - the total transaction costs taken as the percentage of the total transaction value of the portfolio, %ARC − annualized rate of return, %ASD − annualized standard deviation in percent, %MDD − maximum drawdown of capital in percent, IR1, IR2 − information ratios, %MT – the mean portfolio turnover ratio in percent.

If reallocation period is changed from 1 week to 3 days, the performance becomes much worse, IR1 and %ARC drops several times. Such poor performance can be explained by high transaction costs, even though %MT is lower for RE 3d. This can be explained by the fact that when the reallocation period is 3 days, the change of assets in the portfolio is more dynamic if compared to 7 day reallocation. The transaction costs for 3 day reallocation are higher because we reallocate the portfolio 1.87 (=2.33*115.3/143.7) times more often. Similar situation, but in opposite direction occurs when we change reallocation period from 1w to 1m. the results significantly improve in terms of %ARC, IR1 and IR2. So, the length of reallocation period significantly impacts the portfolio performance.

Analyzing the sensitivity to parameter N, we can see that the worst performance is noticed when we keep only 5 coins in the portfolio during a reallocation period. The lower the number of coins in the portfolio, the higher is the portfolio turnover. Consequently, with N changing from VAR to 5, and accordingly higher %MT the statistics demonstrates decreasing figures for %ARC, IR1 and IR1 and increasing values of %ASD and %MD. The best results are observed for the varying number of cryptocurrencies in the portfolio (meaning any number advised by SVM output for buying are kept in the portfolio). Additionally, statistics are very sensitive to the parameter N.

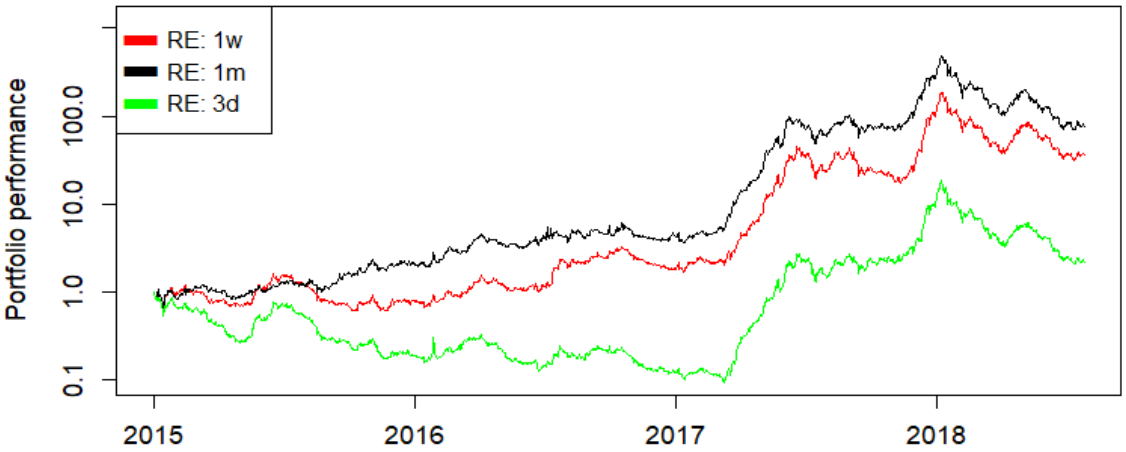
If we change the training size (%TS) through 25%, 50% and 100% (for example, for 50% it is 25 coins from class + and 25 coins from class −), it does not exercise significant impact on the performance of our strategies, but we still observe that the best results are observed for the smallest training size.

Performance of the portfolios heavily depends on the magnitude of transaction costs but it is rather straightforward. For %TC equaled 0.5 the annual return is substantially higher than for %TC equaled 1.

Therefore, the common feature for the sensitivity analysis is that the shorter the reallocation period and the lower the number of cryptocurrencies, the lower is

the performance for the strategy measured by IR1 and IR2. The performance of the portfolios heavily depends on the magnitude of transaction costs and relatively to a lesser extent depends on the change of training size. Addressing the research questions stated in the beginning, the strategy results are significantly sensitive to the three out of four chosen parameters. So, the model does not provide robust results.

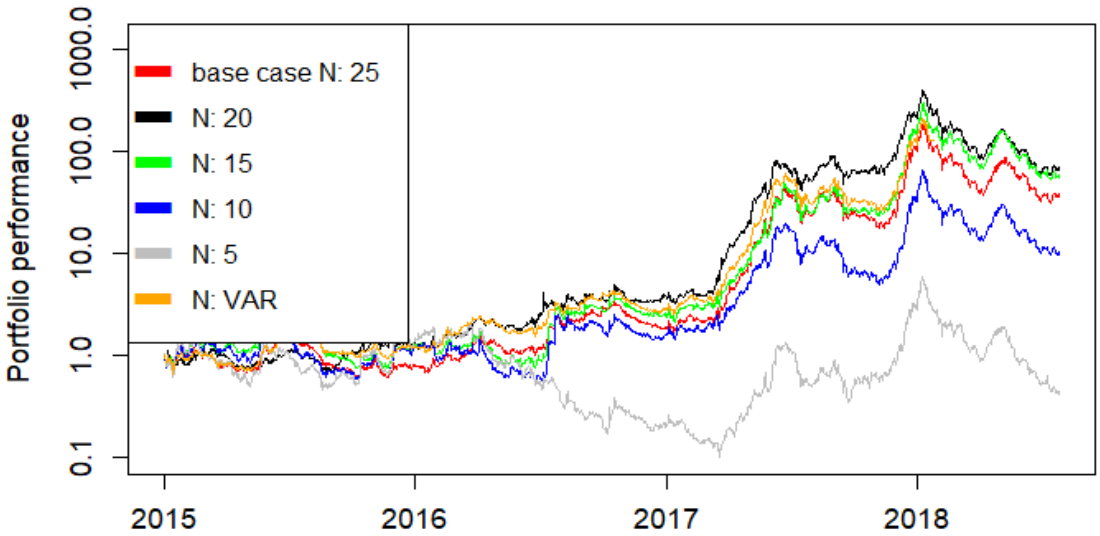
Figure 4.3. Equity lines of the SVM strategy with changing reallocation period RE: 1 week (base case), 1 month and 3 days



Note: the graph shows the equity lines of the SVM strategy with changing reallocation period RE over the period from 01/01/2015 to 01/08/2018. The length of reallocation period significantly impacts the portfolio performance.

Figure 4.3 and Figure 4.4 presents respectively equity lines for the SVM strategies with the varying parameters such as reallocation period RE and number of assets N kept in the portfolio.

Figure 4.4. Equity lines of the SVM strategy with changing number of assets N in the portfolio: 25, 20, 15, 10, 5 and VAR

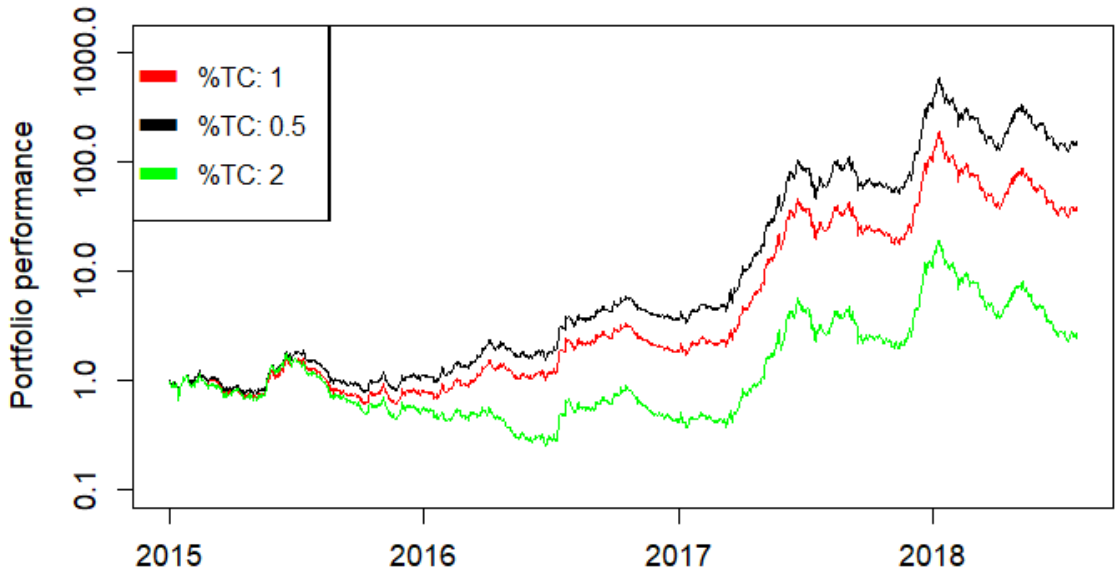


Note: the graph shows the equity lines of the SVM strategy with changing number of assets N in the portfolio over the period from 01/01/2015 to 01/08/2018. The worst performance is noticed when only 5 coins are kept in the

portfolio during a reallocation period. The lower the number of coins in the portfolio, the higher is the portfolio turnover.

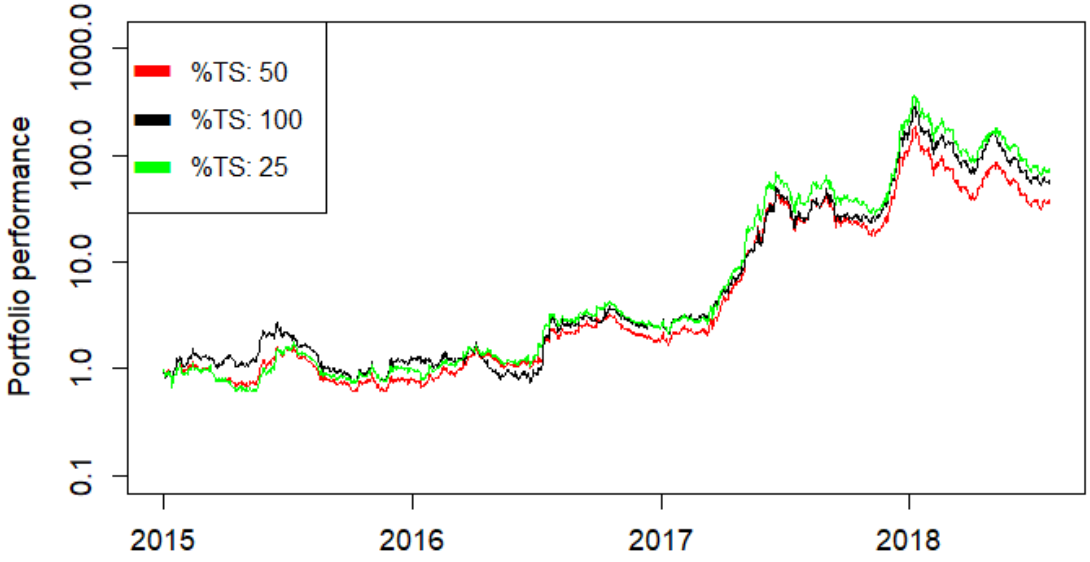
Equity lines with the varying parameters such as transaction costs %TC and length of training set TS are presented on the Figure 4.5 and Figure 4.6 respectively.

Figure 4.5. Equity lines of the SVM strategy with various transaction costs (%TC equaled 2%, 1% and 0.5%)



Note: the graph shows the equity lines of the SVM strategy with changing transaction costs %TC in the portfolio over the period from 01/01/2015 to 01/08/2018. Performance of the portfolios heavily depends on the magnitude of transaction costs, what can be obviously seen from the behavior of the equity lines.

Figure 4.6. Equity lines of the SVM strategy with changing length of training set TS: 25%, 50% and 100%



Note: the graph shows the equity lines of the SVM strategy with changing length of training set %TS in the portfolio over the period from 01/01/2015 to 01/08/2018. As lines are evolving very close to each other, one may conclude that the change of the parameter %TS does not exercise significant impact on the portfolio statistics.

5. Conclusions

The main aim of this paper was to apply SVM algorithm to build an investment strategy for cryptocurrency market and investigate its profitability. The research hypothesis was that the strategy based on SVM algorithm is able to outperform the benchmark strategies in terms of return-risk relation. The results of this investigation were reported for the period between 2015-01-01 and 2018-08-01. The main hypothesis that the investment strategy based on SVMs algorithm outperforms benchmark strategies is rejected based on IR1 values.

The main methodology concepts were based on the research paper „Nonlinear support vector machines can systematically identify stocks with high and low future returns" by Huerta et. al. (2013) and Kość et. al. (2018) "Momentum and contrarian effects on the cryptocurrency market".

SVM was implemented to build the trading strategy in the following way. Training set is a tail set which is defined to be a group of coins whose volatility-adjusted price change is in the highest and lowest quantile. Each asset is presented by a set of six technical features. SVM is trained on historical tail sets and tested on current data. The classifier is chosen to be a nonlinear support vector machine. The SVM is trained and tested once per reallocation period. Portfolio is formed by ranking coins using the SVM output. The highest ranked coins are used for long positions.

Our results show that EqW portfolio outperforms all the benchmark strategies and also SVM strategy. It gives the highest values of IR1 and IR2 demonstrating abnormal returns. The performance of the SVM strategy was ranked the fourth being better only from S&P B&H strategy. Therefore, the main hypothesis stated at the beginning of this paper was rejected based on IR1 values.

The SVM strategy has not demonstrated abnormal returns. Moreover, the results are not stable and the algorithm itself does not provide robust outcomes. The performance of the portfolio is extremely sensitive to the parameters. In this study only the influence of the four parameters has been checked. The performance is highly sensitive to the number of assets kept in the portfolio. If we include only these assets recommended by SVM function output in the portfolio (N = VAR), the results get closer to the best EqW strategy. The magnitude of transaction costs and the length of reallocation period heavily impact the performance statistics as well. Only the size of training set does not have any significant impact on the outcome.

It is important that quite a large number of parameters which are deemed to be fixed in our analysis, might influence the final results of the portfolio performance. Especially, the choice of the meta parameters C and γ play a very important role. Actually, the strategy produces notably different figures due to the fact that the method of choice of meta parameters is greed search with fixed sampling. The computer power does not allow to estimate the optimal parameters for the whole training set and it is the reason why the cost and gamma can be different if we run the analysis for broader set of possible values. As application of SVM implies setting of quite a large number of parameters, this makes the model very prone to the problem of overfitting. Therefore, length of historical data taken to calculate technical features, lambda λ used to calculate exponential moving average for returns, length of training data which were fixed parameters in the model can influence the final results of our analysis.

"Buy" candidates for the portfolio are defined by SVM output basing on the rule that assets are included in the portfolio if their returns are predicted to grow. It implies that decision is guided mainly by momentum rule. We invest in these assets whose returns are predicted to grow. In the paper of Kość et. al. (2018) was shown that there is a lack of the momentum effect on the cryptocurrency market. In the opposite, the results proved the existence of strong contrarian effects. Therefore, it is worth checking the performance of the contrarian portfolio, i.e. if we select as long positions the assets whose returns were assumed to decrease in the past and therefore were predicted by SVM to fall in the future. As one more potential continuation of this study, there can be carried out the sensitivity analysis for these parameters which were deemed fixed in the model. Moreover, it will be interesting to run the market neutral strategy including both long and short positions, so that there will be hedging for long positions.

BIBLIOGRAPHY

Boyd, S., Vandenberghe, L., 2004. Convex Optimization. Cambridge University Press, New York, NY, USA.

Cawley, G., Talbot, N., 2010. On over-fitting in model selection and subsequent selection bias in performance evaluation. J. Mach. Learn. Res. 11, 2079–2107.

Chen M., Narwal N., Schultz. M., 2018, Stanford University. Predicting Price Changes in Ethereum. URL: https://pdfs.semanticscholar.org/ceff/65e02b2b9b6b181cfc956350351b8e284a01.pdf?_ga=2.139748214.472574922.1533418619-1566045322.1533418619

Chordia, T., Swaminathan, B., 2002. Trading volume and cross-autocorrelations in stock returns. J. Financ. 55 (2), 913–935.

Cristianini, N. and Shawe-Taylor, J. (2000), An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods, Cambridge University Press, Cambridge

David Meyer. Support Vector Machines. The Interface to libsvm in package e1071. URL: https://cran.r-project.org/web/packages/e1071/vignettes/svmdoc.pdf

Hastie, T., Tibshirani, R., Friedman, J., The Elements of Statistical Learning. Data Mining, Inference, and Prediction

Huang, W., Nakamori, Y., Wang, S.-Y., 2005. Forecasting stock market movement direction with support vector machine. Comput. Oper. Res. 32 (10), 2513–2522.

Huang, Z., Chen, H., Hsu, C.-J., Chen, W.-H., Wu, S., 2004. Credit rating analysis with support vector machines and neural networks: A market comparative study. Decis. Support Syst. 37 (4), 543–558.

Huerta, R., Corbacho, F., Elkan, C., 2013. Nonlinear support vector machines can systematically identify stocks with high and low future returns. IOS Press. Algorithmic Finance 2, (45–58).

Huffman, S., Moll, C., 2011. The impact of asymmetry on expected stock returns: An investigation of alternative risk measures. Algorithmic Financ. 1 (2), 79–93.

James, G., Witten, D., Hastie, T., Tibshirani, R., An Introduction to Statistical Learning with Applications in R

Jegadeesh, N., Titman, S., 2012. Returns to buying winners and selling losers: Implications for stock market efficiency. J Financ. 48 (1), 65–91.

Joachims, T. 1998. Text categorization with Support Vector Machines: Learning with many relevant features. European Conference on Machine Learning ECML 1998: Machine Learning: ECML-98 pp 137-142.

Kim, K., 2003. Financial time series forecasting using support vector machines. Neurocomputing 55 (1), 307–319.

Kość K., Sakowski P., Ślepaczuk R., 2018, *Momentum and Contrarian Effects on the Cryptocurrency Market,* Working Papers of Faculty of Economic Sciences, University of Warsaw, WP 9/2018 (268), https://www.wne.uw.edu.pl/files/2615/2405/6484/WNE_WP268.pdf

Meyer D., Misc Functions of the Department of Statistics, Probability Theory Group (Formerly: E1071), TU Wien. Package "e1071". URL: https://cran.r-project.org/web/packages/e1071/e1071.pdf

Muller, K., Mika, S., Ratsch, G., Tsuda, K., Scholkopf, B., 2001. An introduction to kernel-based learning algorithms. IEEE Neural Network. 12 (2), 181–201.

Package "TTR". URL: https://cran.r-project.org/web/packages/TTR/TTR.pdf

Performance Analytics in R. URL: https://cran.r-project.org/web/packages/PerformanceAnalytics/vignettes/portfolio_returns.pdf

Pistole, T. C. Comparison of three technical trading methods vs. buy-and-hold for the S&P 500 market. Graduate Student of Finance, University of Houston – Victoria. URL: http://swdsi.org/swdsi2010/SW2010_Preceedings/papers/PA153.pdf

Rouwenhorst, K., 2002. International momentum strategies. J. Financ. 53 (1), 267–284.

Sewell, M., 2010. The Application of Intelligent Systems to Financial Time Series Analysis, PhD thesis, PhD dissertation, Department of Computer Science, University College London, University of London

Ślepaczuk R., Sakowski P., Zakrzewski G., 2018, *Investment strategies beating the market. What can we squeeze from the market?*, eFinanse, forthcoming.

Tay, F., Cao, L., 2001. Application of support vector machines in financial time series forecasting. Omega 29 (4), 309–317.

Tay, F., Cao, L., 2002. Modified support vector machines in financial time series forecasting. Neurocomputing 48 (1), 847–861.

Van Gestel, T., Suykens, J., Baestaens, D., Lambrechts, A., Lanckriet, G., Vandaele, B., et al., 2001. Financial time series prediction using least squares support vector machines within the evidence framework. Neural Netw., IEEE Trans. 12 (4), 809–821.

Vapnik, V., 1999. The Nature of Statistical Learning Theory. springer, Heidelberg, Germany

University of Warsaw
Faculty of Economic Sciences
44/50 Długa St.
00-241 Warsaw
www.wne.uw.edu.pl