ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE SISTEMAS

UNIDAD DE TITULACIÓN

Cryptocurrency upward trend direction prediction to optimize trading strategy

TRABAJO DE TITULACIÓN PREVIO A LA OBTENCIÓN DEL GRADO DE MAGÍSTER EN COMPUTACIÓN

Efrain Gonzalo Zaldumbide Cevallos

efrain.zaldumbide@epn.edu.ec

Director: MSc. Henry Patricio Paz Arias

henry.paz@epn.edu.ec

2020

DIRECTOR’S APPROVAL

As director of the CRYPTOCURRENCY UPWARD TREND DIRECTION PREDICTION TO OPTIMIZE TRADING STRATEGY degree work developed by Efraín G. Zaldumbide C., student of the Master’s degree in Computing, having supervised the completion of this work and made the corresponding corrections, I approve the final drafting of the written document to continue with the corresponding procedures to support the oral defense.

MSc. Henry Patricio Paz Arias
DIRECTOR

AUTHORSHIP DECLARATION

I, Efrain Gonzalo Zaldumbide Cevallos , declare under oath that the work described here is my responsibility; that has not been previously submitted for any degree or professional qualification; and, that I have consulted the bibliographic references included in this document.
Escuela Politécnica Nacional (EPN) and the participating universities in the project CRYPTOCURRENCY UPWARD TREND DIRECTION PREDICTION TO OPTIMIZE TRADING STRATEGY may use the rights corresponding to this work, as established by the Intellectual Property Law, by its Regulations, and by current institutional regulations.

Efrain Gonzalo Zaldumbide Cevallos

DEDICATED TO

My family EZ, MZ, AZ, XZ, CS, VS, KO, JZ.

ACKNOWLEDGMENT

The culmination of this work has been possible thanks to the guidance of the MSc. Henry Paz, to the knowledge imparted by the professors of the Master in Computer Science. I thank the authorities of the faculty and especially the coordinators of the master's degree who, at one point, have not hesitated to support us, MSc. Ivan Carrera, Dra Tania Calle, Dra. Diana Yacchirema, I thank my family and my girlfriend for giving me their support, patience, and love.

Contents

List of Figures

List of Tables

Abstract

The prediction of stock market behavior is of high interest in an extensive community of people. No matter how small an improvement is in this field, it can drive investors to increase their profit. In this study, we analyze one section inside the big problem of micro trading by using intraday short time intervals to reduce the social influence and market speculation. We attempt to predict when an altcoin price starts to rise and reach a predefined minimum profit percentage. We select the cryptocurrency Ethereum (ETH/BTC) as the test element from many existing altcoins and carry out experiments using a support vector machine algorithm. We explain why we chose SVM, our dataset construction, and the training and testing processes. We break down the results by comparing them with related works that share our interests.

Keywords— support vector machine, stock market, stock index, stock direction prediction, cryptocurrency, trading

Resumen

La predicción del comportamiento del mercado de valores es de gran interés en una amplia comunidad de personas. No importa cuán pequeña sea una mejora en este campo, puede impulsar a los inversores a aumentar sus ganancias. En este estudio, analizamos una sección dentro del gran problema del micro-trading utilizando intervalos de tiempo cortos con el objetivo de reducir la influencia social y la especulación del mercado. Intentamos predecir cuándo el precio de una altcoin comienza a subir y alcanza un porcentaje de ganancia mínima que predefinimos. Seleccionamos la criptomoneda Ethereum (ETH / BTC) como elemento de prueba de entre muchas altcoins existentes y llevamos a cabo experimentos utilizando support vector machine. Explicamos por qué elegimos SVM, la construcción de nuestro conjunto de datos y los procesos de entrenamiento y pruebas. Desglosamos los resultados comparándolos con trabajos relacionados con los objetivos de esta investigación que comparten nuestro interés.

Palabras clave— support vector machine, mercado de valores, stock index, predicción de índice de mercado, cryptomoneda, trading

Chapter 1

Introduction

1.1 Background

The prediction of stock price trend direction is an important goal in the financial market. Any small improvement can be very profitable (Ballings et al., 2015), but predicting stock price trend direction is difficult due to the volatility and uncertainties. Investors can perform two types of studies before investing; the ”Fundamental Analysis” and the ”Technical Analysis.”

Fundamental Analysis points to each industry business model, economy, politics, climate, season, etc. Technical Analysis uses the evaluation of stock markets based on the study of statistics generated by market activity, such as past prices, volumes, and predefined mathematical indicators (Patel et al., 2014).

When we carry out Technical Analysis, we usually use stock charts to identify patterns, definitions, and trends that may suggest how a stock price will behave in the future.

In this chapter, we explain the directives of this research, its focus, its hypothesis, and objectives.

1.2 Theoretical Framework

1.2.1 Research Focus and Objectives

This study directs its interest in micro trading strategies within the technical analysis, which can be defined as either: few investment amounts, short time intervals, or both. Our attention is fixed in short time intervals, and we will refer during this research to micro trading as the strategy that uses short time intervals in the order of minutes. This strategy commonly uses computer programs to automate the processes since the fundamental objective is to carry out continuous analyses to trigger many trading operations. Micro trading and other trading strategies' primary challenges are identifying "when to buy" and "when to sell."

Within the "when to buy," there are several subjects that are related to it. In particular, one of them is where this research focuses: trying to identify *when it is most likely that a price increase will occur in the immediate future*.

1.2.2 Research question and Hypothesis

Due to the intrinsic difficulty of making predictions in volatile environments such as the stock market, using machine learning techniques to on-board this problem becomes challenging. The variables, conditions, and behaviors obey very complex models touching the region of the unpredictable. With all of this in mind, we decide to select a small problem from the universe of problems to solve in this context, reaching our research question and hypothesis.

Research Question

It is possible that by using machine learning techniques, we can predict an upward trend direction in the stock market price to achieve profitable intraday trading operations?

Hypothesis

Historical market pricing data and financial indicators provide sufficient information to create machine learning models to predict the upward trend direction of stock prices inside intraday periods.

Research Goal

To improve micro trading strategies by predicting the upward trend direction of the stock price in short-term periods, using historical pricing data, financial indicators, and a machine learning technique.

1.2.3 Specific goals of the Research

We organize our work in separate blocks that will help us understand their purpose and how they integrate to achieve this research's main goal.

1. To complete a literature review to identify information gaps and adjust our primary ideas.
2. To select a machine learning technique that allows us to carry out the experimentation of this study.
3. To tag upward stock price trends from preprocessing candlesticks [1] data and choose the financial indicators that better describe this phenomenon.
4. To choose the trading strategy threshold that has the potential to enhance the total financial profit.
5. To properly analyze the performance of the selected machine learning technique in different short-term time intervals to predict the upward trend direction of stock prices.
6. To propose a method to use the obtained model in micro trading strategies.

[1]"A **candlestick** is a type of price chart used in technical analysis that displays the high, low, open, and closing prices of a security for a specific period." (Achelis, 2001)

1.2.4 Value of this Research

We focus on understanding how short-term periods analysis of stock market prices can improve micro trading strategies by using financial indicators with a detailed tagging strategy and a machine learning technique. The training, testing, and usage of the model should fit the micro trading data availability and its specific characteristics. Additionally, we carry out validation tests by defining a strategy that proposes a dynamic updating methodology of the trained model.

We divide this work into four chapters:

Chapter 1: Introduction
This chapter contains an overview of the problem that we are analyzing, here we explain the goals that we pursue and also contains the theoretical framework

Chapter 2: Research Methods
This chapter is about the research strategy, methodology, and procedures used in this research.

Chapter 3: Experimental Results
This chapter contains the analysis of the results collected by the utilization of our stated methodology.

Chapter 4: Conclusions
This chapter contains an analysis regarding the objectives of this research, presenting a review of our main conclusions and examining the results.

1.2.5 Machine learning in the prediction of stock price trend direction

In this section, we consider some works that utilize machine learning algorithms to predict stock market trend directions. We describe the search and selection processes and subsequently present a breakdown of the analysis of these studies.

1.2.5.1 Search Process

For the literature review, the scientific databases listed below are used:

- ACM digital library (https://dl.acm.org/)
- IEEE Xplore (http://www.ieeexplore.ieee.org)
- Science Direct (Elsevier)(http://www.sciencedirect.com)
- Scopus (https://www.scopus.com)
- Springer (http://www.springerlink.com)
- Web of Science (https://login.webofknowledge.com)

In this research, we seek to satisfy the literature review process by incorporating several parameters for the construction of the search chain:

1. The general research context:
   (a) stock market
   (b) cryptocurrencies
   (c) pricing
   (d) trading
2. Particular context:
   (a) trend
   (b) index
   (c) direction
3. Outcome:
   (a) classification
   (b) detection
   (c) prediction

Table 1.1: Search String

(”stock*” **OR** ”crypto*” **OR** ”price” **OR** ”trading”) **AND** (”classification” **OR** ”detection” **OR** ”prediction”) **AND** (”trend” **OR** ”index” **OR** ”direction”)

1.2.5.2 Study Selection

1.2.5.3 Inclusion and Exclusion criteria

To recognize the most appropriate documents in the search stage, we implemented the criteria bellow.

Exclusion:

- Any of the techniques not in the scope of Machine Learning, see Fig. 1.1.
- Studies without results.
- Studies prior 2010.
- Studies using unsupervised learning techniques.
- Studies using data in intervals greater than one month.

Inclusion:

- Studies using financial indicators.
- Studies using Neural Networks.
- Studies using Support Vector Machines.
- Studies that compare between techniques.
- Studies using in intervals shorter than one month.

1.2.5.4 Studies Evaluation

We organize the studies selected using the study selection criteria in a way that helps us identify:

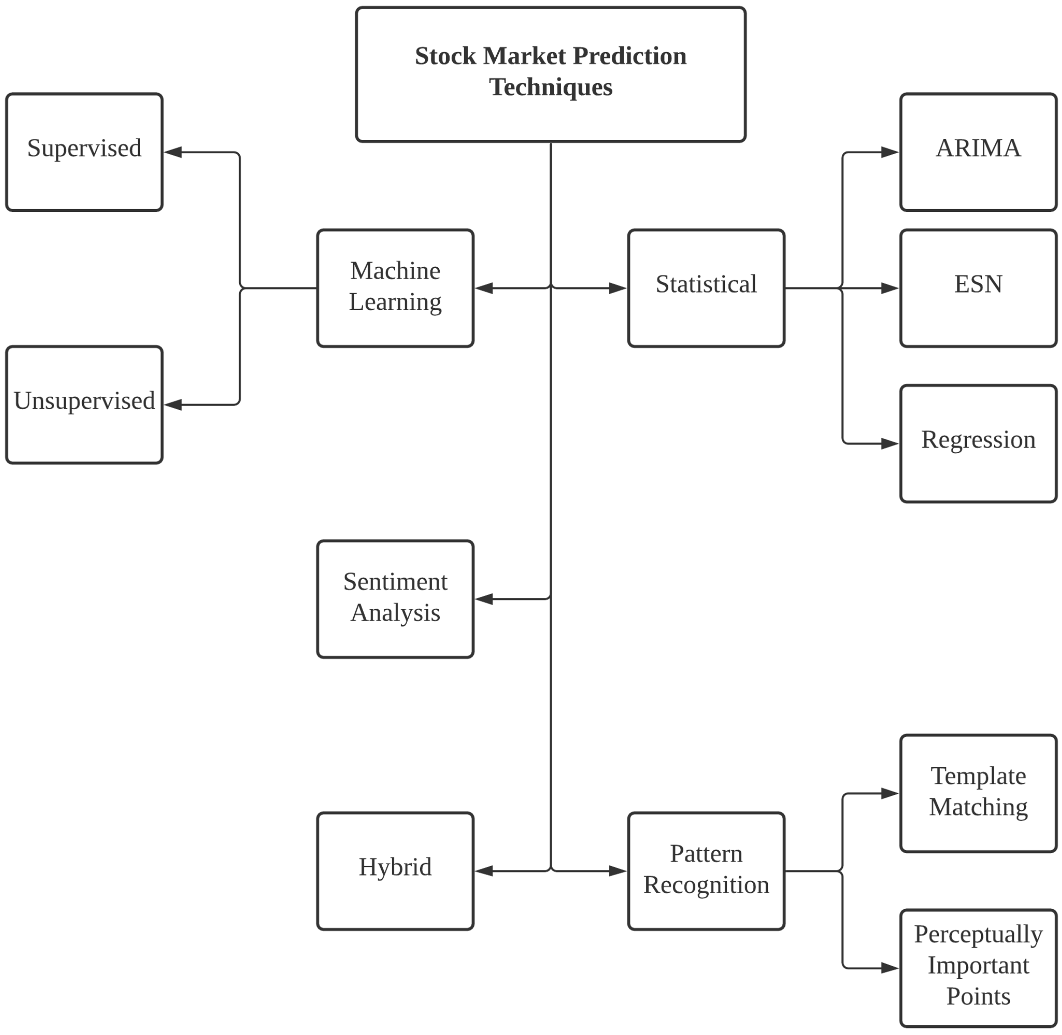

Figure 1.1: Categorization of stock market prediction techniques

- The time intervals used in the studies.
- The algorithms used in the studies.
- The studies results in a defined context.

1.2.5.5 Time intervals

We have selected 22 studies that we show in Table 1.2, 18 of them limit their research to the use of data in 1-day intervals, one of them in 5-day intervals, another of 1 week, and 2 in 1-month intervals. 1.2

One-day intervals have a significant presence in studies mainly because of the availability of datasets. We can appreciate the absence of studies using intraday data. This is probably because of the popularity of daily trading strategies. For approaches such as the one part of this study (micro trading), it is imperative to use intervals in the range of minutes.

The limited usage of intraday data in studies that employ machine learning techniques drives us to a gap in the information justifying new researches that uses intraday data.

1.2.5.6 Algorithms

The algorithms used in the studies are listed in Table 1.4, and a record of their usage in Table 1.3.

We can see that the most used machine learning algorithms are Neural Networks, Support Vector Machines, and Random Forest, but we are also interested in knowing the number of studies that compare these three algorithms as part of their researches, Table 1.5.

1.2.5.7 Studies Results

We focus this study on micro trading with a training methodology that aims to dynamically update the models after a defined elapsed time. We must also consider the continuous evaluation of the prices that reach many trading operations

Table 1.2: Selected studies time intervals

Ref	Year	Period
(Bernal, Fok, & Pidaparthi, 2012)	2012	1 day
(Ji, Che, & Zong, 2014)	2014	1 day
(Bisoi & Dash, 2014)	2014	1 day
(Shynkevich et al., 2015)	2015	1 day
(Żbikowski, 2015)	2015	1 day
(Hafezi, Shahrabi, & Hadavandi, 2015)	2015	1 day
(Patel et al., 2014)	2015	1 day
(Ballings et al., 2015)	2015	1 month
(Milosevic, 2016)	2016	1 month
(Dey et al., 2016)	2016	1 day
(Di Persio & Honchar, 2017)	2017	5 day
(Yang, Gong, & Yang, 2017)	2017	1 day
(Zhang et al., 2018)	2018	1 day
(Hossain et al., 2018)	2018	1 day
(Althelaya, El-Alfy, & Mohammed, 2018)	2018	1 day
(Zhou et al., 2019)	2019	1 day
(Mallqui & Fernandes, 2019)	2019	1 day
(Ghosh, Jana, & Sanyal, 2019)	2019	1 day
(Carmona, Climent, & Momparler, 2019)	2019	1 day
(Lee et al., 2019)	2019	1 week
(Ribeiro & dos Santos Coelho, 2020)	2020	1 day
(Ismail et al., 2020)	2020	1 day

in short-term intervals. There are many requests to an API, and there are restrictions not allowing to obtain an amount of significant data in a single query and limit the number of requests in a given period. Due to these limitations, the algorithm we select must not depend on an abundant amount of data for its correct operation.

Both SVM and Neural Networks can map the input data to a higher dimensional space to assign a decision boundary. Both algorithms have good customization parameters. In Neural Networks, we can commonly adjust the number of layers, learning rate, epochs, activation functions. In Support Vector Machines, we can adjust the gamma and the regularization parameter and also employ distinct kernels. In Random Forest, there are not many parameters to be controlled, but the forest's depth and the number of trees in each level.

Table 1.3: Number of studies using a particular algorithm

Algorithm	Presence in selected studies
Neural Networks: (NN)	14
Support Vector Machines (SVM)	8
Logistic Regression (LR)	4
K-Nearest neighbors (KNN)	2
Gradient boosting (XGBoost)	3
Adaptive Boosting (AdaBoost)	1
Multiple kernel learning (MKL)	2
Random Forest (RF)	8

A neural network requires a large number of input data compared to SVM and Random Forest since, in neural networks, the more data is available, the better.

Random Forest is intrinsically well suited for multi-class problems, while SVM is primarily for two-class.

Random Forest works well when we use numerical and categorical features while SVM maximizes the boundary, relying on the concept of distance between different points.

In Table 1.6, we summarize the results of the selected studies, limiting to only those that we have chosen as the candidates that have definitions close to the objectives, limitations, and definitions of this study.

We can observe in a general way that both: neural networks and support vector machines have very close results in the chosen studies. Support vector machines show slightly better results in 3 of the four studies in which comparisons are made within each study's context and their specific evaluation metrics. In the context of this study, we decide to use support vector machines. It is beneficial that SVMs are not limited to a large amount of data while maintaining good control of its parameters. We also consider that SVM conceptual definition based on decision margins fits with the volatile characteristics of the problem we want to solve.

Table 1.4: Selected studies algorithms

Ref	NN	SVM	LR	KNN	XGBoost	AdaBoost	MKL	RF
(Bernal, Fok, & Pidaparthi, 2012)	✓							
(Ji, Che, & Zong, 2014)	✓							
(Bisoi & Dash, 2014)	✓							
(Shynkevich et al., 2015)	✓	✓					✓	
(Żbikowski, 2015)		✓						
(Hafezi, Shahrabi, & Hadavandi, 2015)	✓							
(Patel et al., 2014)	✓	✓						✓
(Ballings et al., 2015)	✓	✓	✓	✓		✓	✓	✓
(Milosevic, 2016)		✓	✓					✓
(Dey et al., 2016)					✓			
(Di Persio & Honchar, 2017)	✓							
(Yang, Gong, & Yang, 2017)	✓							
(Zhang et al., 2018)								✓
(Hossain et al., 2018)	✓							
(Althelaya, El-Alfy, & Mohammed, 2018)	✓							
(Zhou et al., 2019)	✓							
(Mallqui & Fernandes, 2019)		✓						
(Ghosh, Jana, & Sanyal, 2019)	✓							✓
(Carmona, Climent, & Momparler, 2019)					✓			
(Lee et al., 2019)		✓	✓					✓
(Ribeiro & dos Santos Coelho, 2020)				✓	✓			✓
(Ismail et al., 2020)	✓	✓	✓					✓

1.2.6 Cryptocurrencies

Since Nakamoto's publication (Nakamoto, 2019) in 2008, and the first Bitcoin transaction in January 2009, several successive situations have driven to an increase in the importance of cryptocurrencies. Some big and small companies have included the acceptance of cryptocurrencies as part of their politics, and others are in the way of doing the same. The records in the usage of Bitcoin and other cryptocurrencies are achieving states that have drastically influenced cryptocurrencies in the market. Academic literature, formal reports, and worldwide statements are full of controversy concerning their value, dynamics, and volatility.

The holding time of Cryptocurrency is shorter than in any other stock market, critically influencing the exchanges over their operations due to polarised viewpoints about cryptocurrencies. Cryptocurrencies are also negatively affected by market speculation. News and social media strongly influences the market behavior (Mai

Table 1.5: Number of studies comparing most used algorithms

Algorithm	Comparison present in studies
(NN) vs (SVM)	4
(NN) vs (RF)	4
(RF) vs (SVM)	5
(NN) vs (RF) vs (SVM)	3

Table 1.6: Selected Studies Results

Ref	Metrics	Better Result	NN Result	SVM Result
(Bernal, Fok, & Pidaparthi, 2012)	Test error	0.0027	0.0027	
(Ji, Che, & Zong, 2014)	Error	0.045	0.045	
(Bisoi & Dash, 2014)	MAPE	1.9188	1.9188	
(Shynkevich et al., 2015)	Acc	81.63	**81.63**	**79.59**
(Żbikowski, 2015)	Return	228.21%		228.21%
(Hafezi, Shahrabi, & Hadavandi, 2015)	MAPE	2.84	2.84	
(Patel et al., 2014)	Acc/F-1	0.92/0.92	**0.88/0.88**	**0.92/0.92**
(Ballings et al., 2015)	AUC	0.9037	**0.7279**	**0.8395**
(Milosevic, 2016)	Acc/F-1	0.765/0.751	**0.765/0.751**	**0.636/0.624**
(Di Persio & Honchar, 2017)	Acc	72%	72%	
(Yang, Gong, & Yang, 2017)	Acc	81.47%	81.47%	
(Hossain et al., 2018)	MSE	0.00098	0.00098	
(Althelaya, El-Alfy, & Mohammed, 2018)	RMSE	0.00736	0.00736	
(Zhou et al., 2019)	RMSE	0.1435	0.1435	
(Mallqui & Fernandes, 2019)	Acc	59.45		59.45
(Ghosh, Jana, & Sanyal, 2019)	RMSE	0.027	0.027	
(Lee et al., 2019)	Acc	60.9%		59.3%
(Ismail et al., 2020)	Acc	77.55%	**73.47%**	**76.53%**

et al., 2018).

This behavior dramatically affects trading operations since it complicates the observations of a possible rise or fall in price. All of these complications are why it is necessary to move as far as possible from the social effect when using technical analysis.

1.2.7 Support Vector Machines (SVM)

A Support Vector Machine (SVM) is a very powerful and versatile Machine Learning model, capable of performing linear and nonlinear classification, regression, and even outlier detection. SVMs are particularly well suited for classification of complex but small or medium sized datasets. (Geron, 2017)

1.2.7.1 Large Margin Classification

The primary idea of SVMs is presented in Figure 1.2. This figure shows how two classes can be separated with a straight line (linearly separable). The $Plot\ a$ of this figure presents three potential linear classifiers. The model in which the dashed line represents the decision border is so bad that it does not even depart the classes accurately. The other two models work correctly, but their decision boundaries come so near to the examples that these models will not correctly operate on new data points. In distinction, the line in $Plot\ b$ also describes the decision boundaries of an SVM classifier. The continuous line not only divides the two classes but is also as far apart from the nearest training examples as it can be[2]".

Notice that adding more training instances "off the street" will not affect the decision boundary at all. It is fully determined or "supported" by the instances located on the edge of the street. These instances are called the support vectors, which are circled in *Plot b* of Figure 1.2.

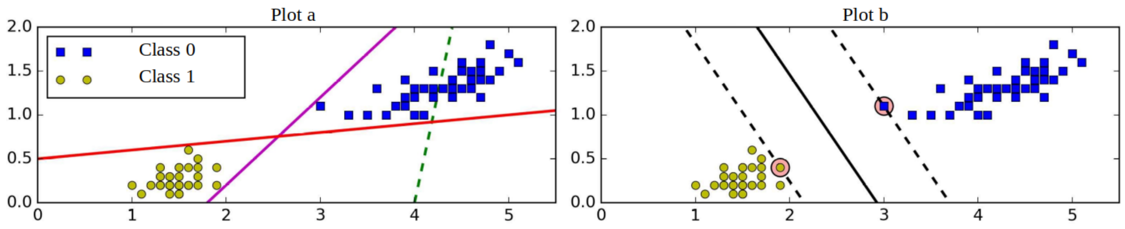

Figure 1.2: Large margin classification
Source: (Geron, 2017)

SVMs are sensitive to the feature scales, as we can see in Figure 1.3. On the left with unscaled data, the vertical scale is much larger than the horizontal scale, so

[2]"You can think of an SVM classifier as fitting the widest possible street (represented by the parallel dashed lines) between the classes. (Geron, 2017)

the widest possible street is close to horizontal. On the right, after feature scaling the decision boundary looks much better.

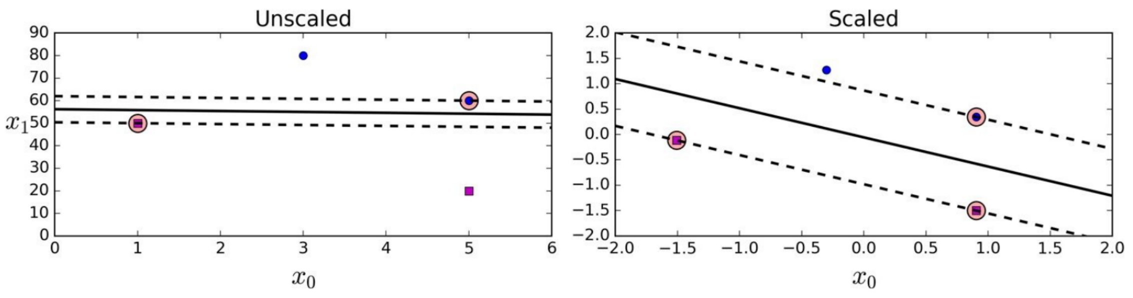

Figure 1.3: Sensitivity of feature scales
Source: (Geron, 2017)

1.2.7.2 Soft Margin Classification

If we strictly impose that all instances be off the street and on the right side, this is called hard margin classification. There are two main issues with hard margin classification:

1. It only works if the data is linearly separable.
2. It is considerably sensible to outliers.

Figure 1.4 shows with just one additional outlier. On $Plot\ a$, it is impossible to find a hard margin, and on $Plot\ b$ the decision boundary ends up very different from the one we saw in Figure 1.2 without the outlier, and it will probably not generalize as well.

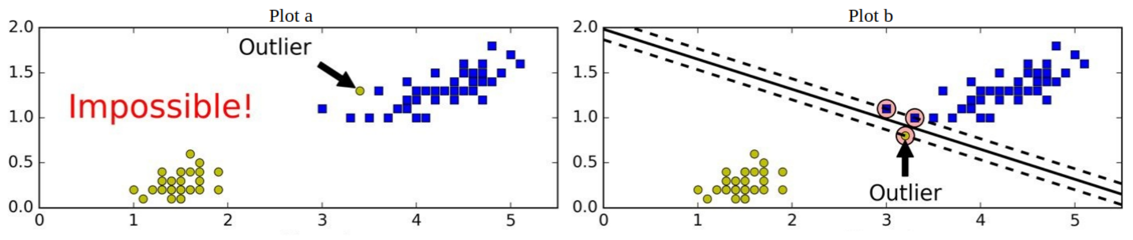

Figure 1.4: Hard margin sensitivity to outliers
Source: (Geron, 2017)

To avoid those problems, it is better to adopt an extra adjustable model. The purpose is to obtain a good balance between keeping the street as large as possible and restricting the boundary invasion. This is named soft margin classification.

We can control this discretion applying the C hyperparameter; a shorter value of C drives to a more extended street but more extended boundary invasions. Figure 1.5 reveals the decision limits and boundaries of two soft margin support vector machine classifiers on a non-linearly separable dataset. On the left, with a high C value of 100, the classifier produces fewer boundary invasions but finishes up with a shorter margin.

On the right, with a low C value of 1, the boundary is much higher, but many examples end up on the street. Nevertheless, it looks like that the second classifier generalizes better; even on this set, it gets fewer prediction errors because most of the boundary invasions are on the correct side of the decision boundary.

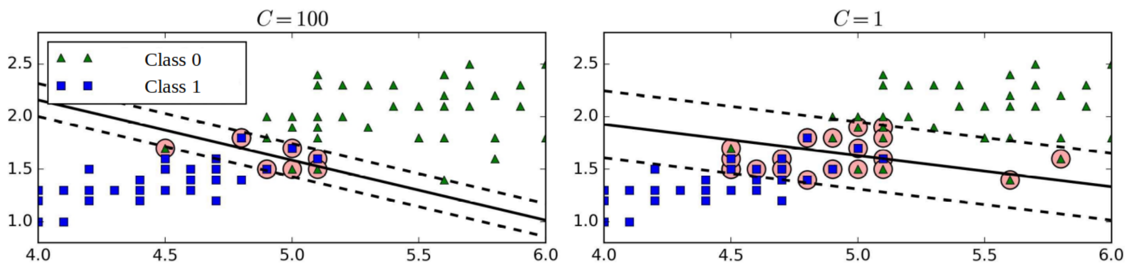

Figure 1.5: Boundary invasions vs margin size

Source: (Geron, 2017)

1.2.7.3 Nonlinear SVM Classification

Although linear SVM classifiers are efficient and work surprisingly well in many cases, many datasets are not even close to being linearly separable. One approach to handling nonlinear datasets is to add more features, such as polynomial features that, in some cases, can result in a linearly separable dataset.

Consider $Plot\ a$ of Figure 1.6, it represents a simple dataset with just one feature x_1. This dataset is not linearly separable, as you can see. But if we add a second feature $x_2 = (x_1)^2$, the resulting 2D dataset is perfectly linearly separable.

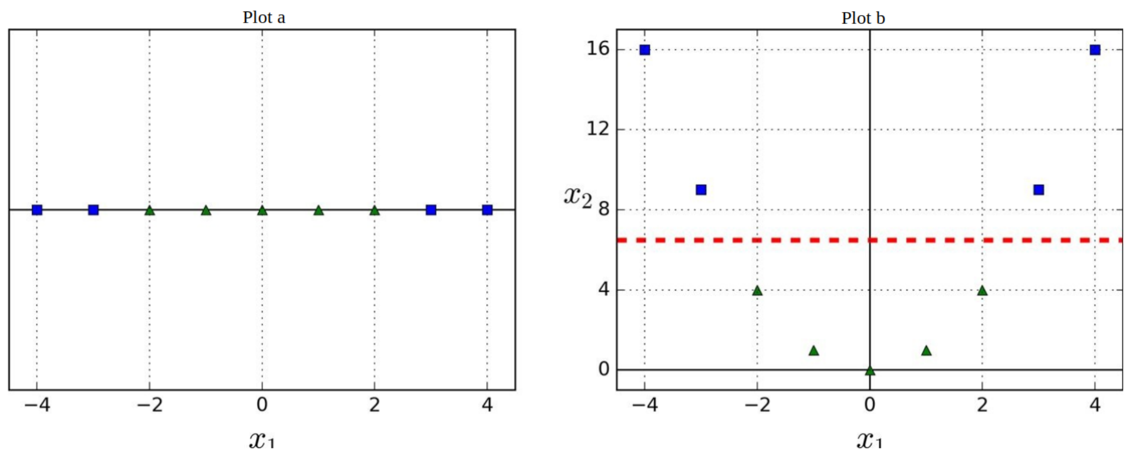

Figure 1.6: Adding features to make a dataset linearly separable
Source: (Geron, 2017)

1.2.7.4 Polynomial Kernel

Adding polynomial features is simple to implement and can work great with all sorts of Machine Learning algorithms (not just SVMs). Still, at a low polynomial degree, it cannot deal with very complex datasets. It creates a considerable number of features with a high polynomial degree, making the model too slow. Fortunately, when using SVMs, we can apply a kernel trick technique. It makes it possible to get the same result as if we added many polynomial features, even with very high-degree polynomials, without actually having to add them. So there is no combinatorial explosion of the number of features since we don't add any features.

In $Plot\ a$ of Figure 1.7, we have a 3rd-degree polynomial kernel SVM Classifier. In $Plot\ b$, we have another SVM classifier using a 10th -degree polynomial kernel. If our model is overfitting, we can reduce the polynomial degree. If it is underfitting, we can try increasing it. The hyperparameter C controls how much high-degree polynomials versus low-degree polynomials influence the model.

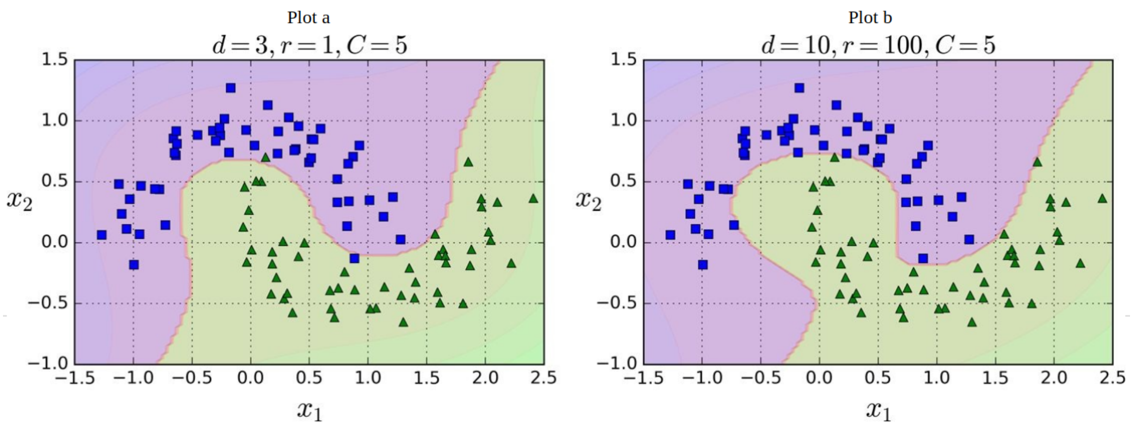

Figure 1.7: SVM classifiers with a polynomial kernel
Source: (Geron, 2017)

1.2.7.5 Adding Similarity Features

Another technique to tackle nonlinear problems is to add features computed using a similarity function that measures how much each instance resembles a particular landmark. We can add two landmarks at $x_1 = –2$ and $x_1 = 1$ (see *Plot a* in Figure 1.8).

The Gaussian Radial Basis Function (RBF) is a bell-shaped function varying from 0 (very far away from the landmark) to 1 (at the landmark). We can compute new features. For example, the instance $x_1 = –1$, is located at a distance of 1 from the first landmark, and 2 from the second landmark. Therefore its new features are $x_2 = exp(–0.3×1^2) ≈ 0.74$ and $x_3 = exp(–0.3×2^2) ≈ 0.30$. *Plot b* of Figure 1.8 shows the transformed dataset (dropping the original features). As we can see, it is now linearly separable.

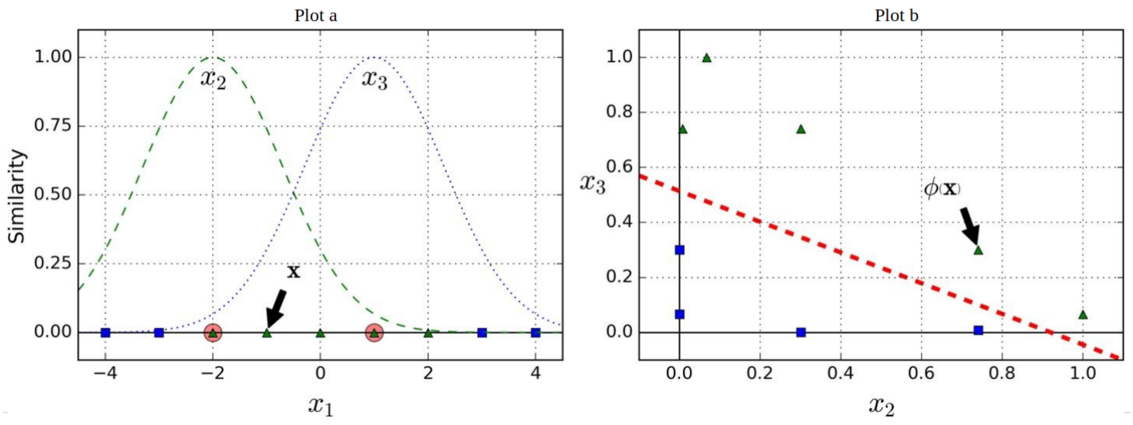

Figure 1.8: Similarity features using the Gaussian RBF
Source: (Geron, 2017)

The simplest approach for selecting the landmarks is to create a landmark at every instance in the dataset. This creates many dimensions and thus increases the chances that the transformed training set will be linearly separable. The downside is that a training set with m instances and n features gets transformed into a training set with m instances and m features. If our training set is very large, we end up with an equally large number of features.

1.2.7.6 Gaussian RBF Kernel

Like the polynomial features method, the similarity features method can be useful with any Machine Learning algorithm. Still, it may be computationally expensive to compute all the additional features, especially on large training sets. However, once again, the kernel trick makes it possible to obtain a similar result as if you had added many similarity features without actually having to add them.

This model is represented in $Plot\ c$ of Figure 1.9. The other plots show models trained with different values of hyperparameters gamma (γ) and C. Increasing gamma makes the bell-shape curve narrower. As a result each instance's range of influence is smaller: the decision boundary ends up being more irregular, wiggling around individual instances.

Conversely, a small gamma value makes the bell-shaped curve wider, so instances have a larger range of influence, and the decision boundary ends up smoother. So γ acts like a regularization hyperparameter. If the model is overfitting, we should reduce it, and if it is underfitting, we should increase it (similar to the C hyperparameter).

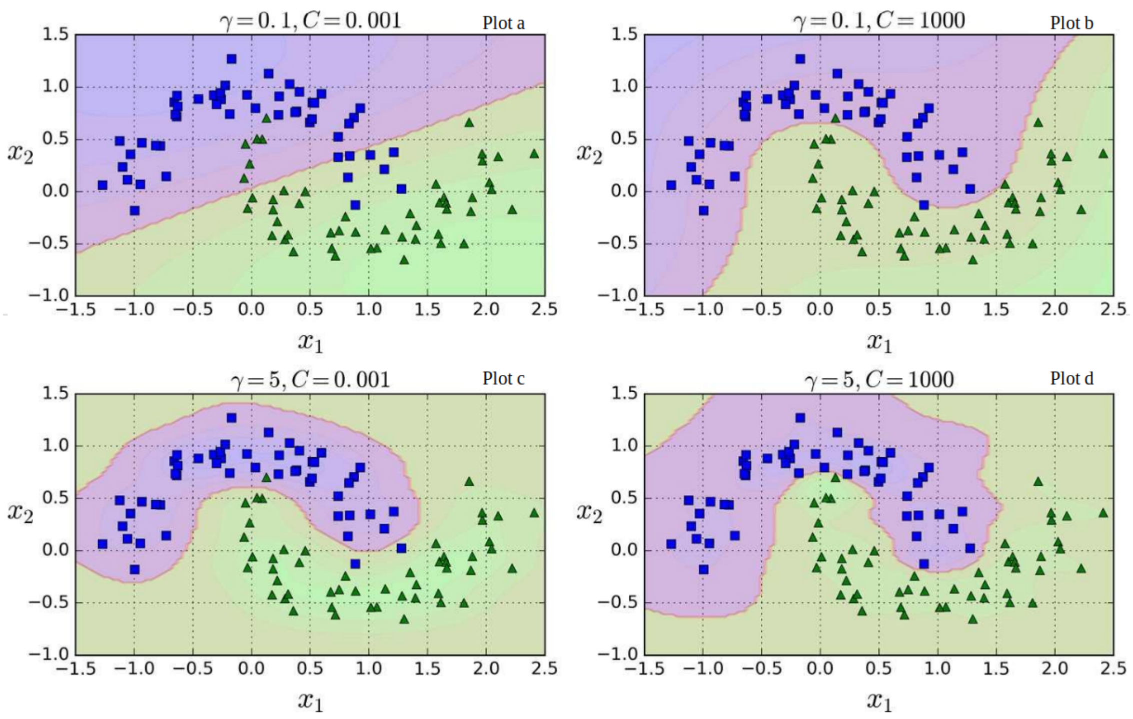

Figure 1.9: SVM classifiers using an RBF kernel
Source: (Geron, 2017)

1.2.8 Trading indicators

Trading indicators are mathematical calculations and part of trading technical analyses. They can help us to identify insights into price trends within the market.

1.2.8.1 Indicators

Considering that:

C_t is the closing price at time t, L_t the low price at time t, H_t the high price at time t and,

$$M_t = (H_t + L_t + C_t)/3, SM_t = \frac{\sum_{i=1}^{n} M_{t-i+1}}{n}, D_t = \frac{\sum_{i=1}^{n}|M_{t-i+1} - SM_t|}{n}, M_{\cup} = \frac{Smothing factor}{1+period}$$

$Commodity\ Channel\ Index$ $(\boldsymbol{CCI})$: Equation 1.1. It measures the variation of a security price from its statistical mean, used to help determine when an investment vehicle is reaching a condition of being overbought or oversold (Achelis, 2001; Chang et al., 1996).

When the *CCI* goes from positive or near-zero readings to below -100, then a downtrend may be starting.

$CCI\ Uptrend\ (\boldsymbol{CCI_{UP}})$: When CCI moves above +100, a security is considered to be entering into a strong uptrend and a buy signal is given.(Achelis, 2001; Chang et al., 1996).

$$CCI_t = \frac{(M_t - SM_t)}{0.0015D_t} \tag{1.1}$$

$Exponential\ Moving\ Average\ (\boldsymbol{EMA})$: Equation 1.2. Is a type of moving average that places a greater weight and significance on the most recent data points. (Sincere, 2010)

(EMAs) are often the most quoted and analyzed short-term averages, they are commonly used to create indicators like the moving average convergence divergence (MACD). It usually gives insights about a market move.

$$EMA_t = (C_t \times M_{\cup}) + EMA_{t-i}\,(1 + M_{\cup}) \tag{1.2}$$

$Stochastic$ %K $(\boldsymbol{K})$: Equation 1.3. Compares a particular closing price of a security to a range of its prices over a certain period of time. (Achelis, 2001).

$$\%K_t = \frac{C_t - LL_{t-n}}{HH_{t-n} - LL_{t-n}} \tag{1.3}$$

$Stochastic$ %D $(\boldsymbol{D})$: Equation 1.4. Moving average of %K. (Achelis, 2001)

$$\%D_t = \frac{\sum_{i=0}^{n-1} \%K_{t-i}}{n} \tag{1.4}$$

$\boldsymbol{K}$ and $\boldsymbol{D}$ are stochastic oscillators and are range-bound, meaning they are always between 0 and 100.

$\%J\ Line\ (\boldsymbol{J})$: Equation 1.5. Represents the divergence of the %D value from the %K. (Achelis, 2001).

$KDJ\ Uptrend\ (\boldsymbol{KDJ_{UP}})$: If we get a value greater than 50, that means the probability of rising is higher than falling. (Murphy, 1999).

$$\%J_t = 3 * \%K * -2\%D \tag{1.5}$$

$Moving\ Average\ Convergence\ Divergence\ (\boldsymbol{MACD})$: Equation 1.6. Is a trend-following momentum indicator that shows the relationship between two moving averages of a security's price. (Achelis, 2001)

$MACD\ Uptrend\ (\boldsymbol{MACD_{UP}})$: The MACD will remain positive as long as there is a sustained uptrend and will remain negative when there is a sustained downtrend. (Achelis, 2001).

$$MACD_t = \frac{C_t - LL_{t-n}}{HH_{t-n} - LL_{t-n}} \tag{1.6}$$

$Relative\ Strength\ Index\ (\boldsymbol{RSI})$: Equation 1.7. Measures the magnitude of recent price changes to evaluate overbought or oversold conditions in the price of a stock or other asset. (Achelis, 2001)

$RSI\ Oversold\ (\boldsymbol{RSI_{OV}})$: RSI is considered overbought when above 70 and oversold when below 30. (Achelis, 2001).

$RSI\ Swing\ Rejection\ (\boldsymbol{RSI_{SR}})$: In case of a bullish swing rejection (Achelis, 2001):

1. RSI oscillator drops under the mentioned 30% barrier.

2. RSI crosses back above 30%.
3. RSI dips again, this time without crossing back into oversold territory RSI then breaks towards the most recent high.

$$RSI_t = 100 - \frac{100}{1 + \left(\sum_{i=0}^{n-1} U\rho_{t-1}/n\right) / \left(\sum_{i=0}^{n-1} Dw_{t-i}/n\right)} \tag{1.7}$$

Chapter 2

Research Method

2.1 Introduction

We mentioned in previous chapters that this study focuses on the analysis of price variations in short time intervals in the order of minutes. We also mention the limitations of data availability in these intervals, and within the last chapter, we select support vector machines as our working algorithm.

Almost all of the studies related to our research, and in the case of the studies that we selected in the previous chapter, all of them focus their attention on predicting both an upward trend and a downward trend.

In the particular case of this study, we will focus on the identification of an upward trend. In the universe of events that can arise in price variations, there is great volatility between rises, falls, stability, and small variations. We define only one independent event as a *”notable long enough upward trend”*, that we will refer in the upcoming chapters as only *”upward trend”* limiting the spectrum of events to this occurrence and anything else that is not.

2.2 Problem Formulation

Let’s define $\boldsymbol{X} \in \mathbb{R}^{nxm}$ as a matrix where each of its vectors $\mathbf{x}_i$ contain the indicators at its specific time at index $\boldsymbol{i}$, and $\boldsymbol{Y}$ as the vector of the class labels $\mathbf{y}_i \in \{0, 1\}$ where $\boldsymbol{i} = 1, 2, ..., n$, a 1 label describes an upward trend direction and the label 0 describes any other ruled out case. Our first goal is to properly tag the data in a way that we can have well-describing labels. Our second goal is to use support vector machines to find a decision boundary with a kernel function $K(\mathbf{x}_i, \mathbf{x}_j)$ that performs the mapping of a feature space $\phi(\mathbf{x}_i)$.

2.3 Research Strategy

Our strategy consists of pre-processing, labeling, training, and testing. The data acquisition stage consists of defining the candlesticks' availability in the intervals that we will be evaluating. We further define our data labeling strategy related to the outputs we want to send to the training and testing stages. We additionally calculate the indicators, and define the features that will include our analysis. Once all of this gets ready, we have to create a training and testing environment to outline the experiments in stock market intervals and indicators within this research context.

We will describe some definitions and a more in-depth evaluation analysis.

The strategy is defined by the following purposes:

1. Tag upward trend labels in data.
2. Select the kernel to use.
3. Select the regularization parameter to apply.
4. Identify the features to use.
5. Select one short time interval.

2.3.1 Data

Data is one of the building blocks when trying to solve any problem using machine learning techniques. Most of the researchers in the context of this study use multiple sources containing public historical datasets, forcing them to adjust their solutions to that amount of data and a fixed data structure. Our approach is to acquire data directly from Binance [1] API. Receiving data from an API allows us

[1]**Binance** is the largest cryptocurrency trading platform nowadays and provides an open-source API to acquire data and trigger trading actions.

to have the opportunity of analyzing historical prices for several cryptocurrencies in different time intervals.

2.3.1.1 Data Tagging

Most of the related research does not explain how data is tagged and how different tagging techniques can impact this particular problem. We believe that this is a highly important process that deserves a well-explained scheme.

One of the indicators that describes a trend change is the exponential moving average (EMA). We show how it is calculated in Equation 1.2.

EMA is just a curve of consecutive points in the same domain of the price curve and can be calculated based on different time range lengths. The shorter this range, the more close is the EMA curve to the price curve. In Figure 2.1 we can see a candlesticks chart with two EMA curves calculated, each one using a different range.

In Figure 2.1 the crossovers are marked in cross symbols. It is important to appreciate that the crossovers are not on the actual cross point of these two EMA curves, this is because each EMA curve is calculated at different intervals, and causes that they do not share points. As a result of this time intervals offsets, we can only define the crossovers close to the cross-points, but they are not actually on them.

We define *Long EMA* $(\boldsymbol{E_L})$ as a vector containing the values of the EMA curve calculated in large-time ranges (red line in Figure 2.1). We also define *Short EMA* $(\boldsymbol{E_S})$ as a vector containing the values of the EMA curve calculated in short-time ranges (blue line in Figure 2.1).

Any dataset containing prices in the structure of candlesticks will start from index 1 to index n, EMA calculations need at least the number of values that belong to the first range, making it to start from index i and to end at index n, it does not have values from indexes 1 to i.

We describe a crossovers vector $\boldsymbol{\alpha}$ evaluated from i to n and containing the in-

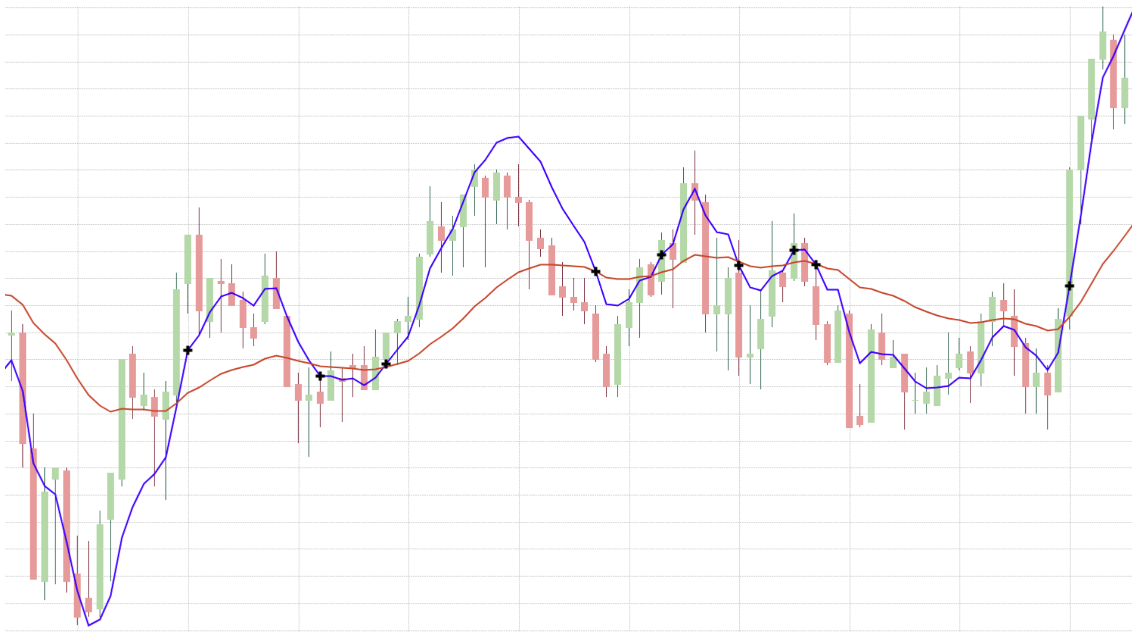

Figure 2.1: Candlesticks with long and short EMAs

dexes when any crossover is occurring, we define this in Equation (2.1).

$$\boldsymbol{\alpha} = \{\, \forall i \;\mid\; \left[(E_{S_i} < E_{L_i}) \land \left(E_{S_{i+1}} > E_{L_{i+1}}\right)\right] \lor \left[(E_{S_i} > E_{L_i}) \land \left(E_{S_{i+1}} < E_{L_{i+1}}\right)\right] \,\} \tag{2.1}$$

2.3.1.2 Exponential moving average in upward and downward trends

When comparing the *Long EMA* against the *Short EMA*, we find that each time they crossover, they are inside an specific trend direction. By following this behavior across all EMAs curve, we have some crossovers in upward trends and the others in downward trends.

2.3.1.3 Crossovers in upward trends

Every time the *Short EMA* changes from being below *Long EMA* to be above it, we are in the situation where the crossover satisfies $[(E_{S_i} < E_{L_i}) \land (E_{S_{i+1}} > E_{L_{i+1}})]$ at index i, $\boldsymbol{\alpha_i}$, so it is inside an upward trend.

2.3.1.4 Crossovers in downward trends

When the *Short EMA* changes from being above *Long EMA* to be bellow it, the opposite happens, the crossover satisfies $[(E_{S_i} > E_{L_i}) \wedge (E_{S_{i+1}} < E_{L_{i+1}})]$, so it is inside a downward trend.

2.3.1.5 Target upward trends

In Figure 2.1 we can appreciate how some trends last less than others till the trend change its direction again. We have a particular interest in long upward trends. A long upward trend should last enough time to let the price enter in a considerable increment till it reaches the desired value of gain percentage.

To identify the upward trends that are part of our interest we have to first extract all of the regions that contains the indexes from price curve where $E_{S_i} > E_{L_i}$, each of these regions has as boundary $[\boldsymbol{\alpha_i}, \boldsymbol{\alpha_{i+1}}]$ where $[(E_{S_i} < E_{L_i}) \wedge (E_{S_{i+1}} > E_{L_{i+1}})]$. We call this region as *Beta Region*.

We also have to extract all of the regions that contains the indexes from price curve where $E_{S_i} < E_{L_i}$, each of these regions has as boundary $]\boldsymbol{\alpha_i}, \boldsymbol{\alpha_{i+1}}[$ where $[(E_{S_i} > E_{L_i}) \wedge (E_{S_{i+1}} < E_{L_{i+1}})]$. We call this region as *Gamma Region*.

We can see a representation of one of these regions (*Beta Region*) in Figure 2.2. To avoid confusions and distinguish between regions we have assigned j to each element of a given region. We define a *Beta Region* as $\boldsymbol{\beta_r}$ and a *Gamma Region* as $\boldsymbol{\gamma_r}$.

Within a Beta or a Gamma region, we define ΔI_i in Equation (2.2) as the number of samples inside any of them, and ΔI_m in Equation (2.3) as the mean ceil length of a group of regions.

$$\Delta I_i = \sum_{j=\alpha_i}^{\alpha_i+1} j \tag{2.2}$$

$$\Delta I_m = \left\lceil \frac{1}{n} \sum_{i=1}^{n} \Delta I_i \right\rceil \tag{2.3}$$

2.3.1.6 Regions of interest

The purpose of $\boldsymbol{\Delta I_i}$ and $\boldsymbol{\Delta I_m}$ calculations is to distinguish the long upward trends we defined above from any others. Knowing the mean length of all *Beta Regions* allows us to establish boundaries and select only those over that boundaries. For this study, we select only *Beta Regions* with a length above the mean length. For each of these regions, we select an upward trend as we show in Figure 2.3, where we can see a highlighted region over the curve of Short Range EMA representing an upward trend.

The complete region of the upward trend is located between both peaks of Figure 2.3, but we take only the highlighted region. We denominate this region a *Long Upward Trend*, and it is located from the peak of the Gamma Region $(Peak_{T_\gamma})$ till the middle point of the crossover point and the peak of the consecutive Beta Region $(Peak_{T_\beta})$.

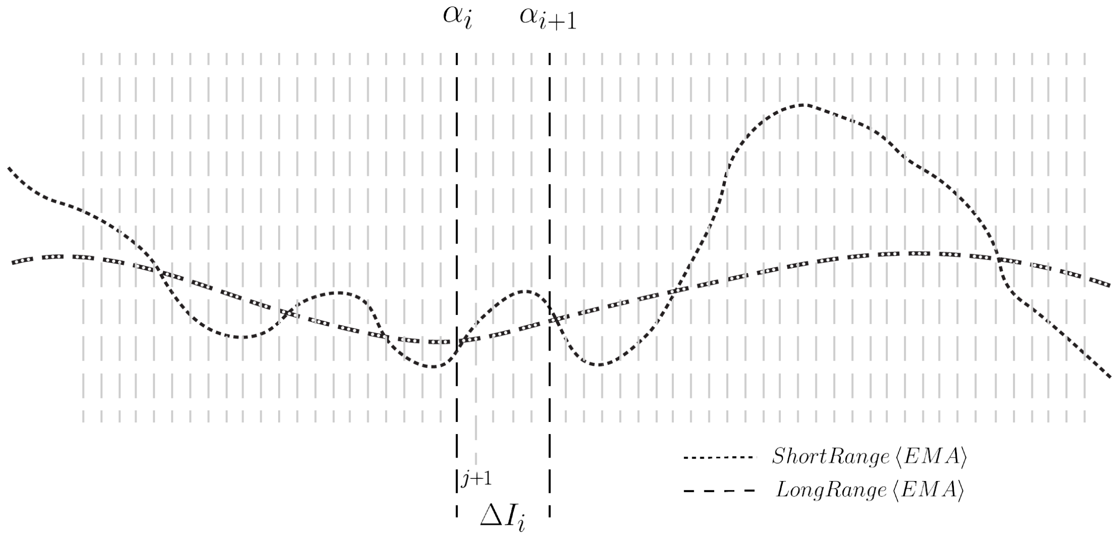

Figure 2.2: Regions of interest

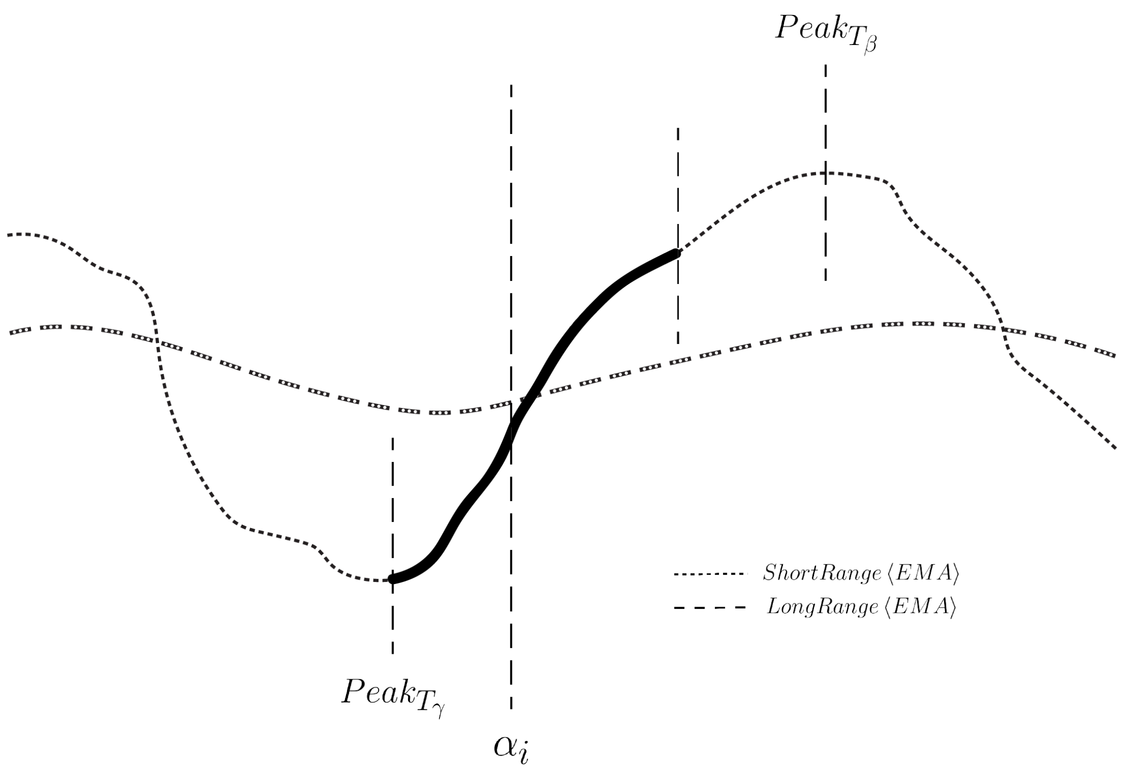

Figure 2.3: Long upward trend

Basically, we have defined class labels that tags with **1** the marked area in Figure 2.3, and all of the rest with a **0** class label.

2.3.2 Features

In the last section, we described how we define data labels for upward trend directions. In the following section, we will cite the features that will be included as part of each vector $\mathbf{x_i}$.

These features have been chosen because they are part of the most common used indicators and help to describe market behavior (Sincere, 2010). We show how they get calculated and their meaning in 1.2.8.

Table 2.1: Feature Vector Attributes

#	Features
1	$ClosePrice$
2	$OpenPrice$
3	$HighPrice$
4	$LowPrice$
5	CCI
6	EMA_7
7	EMA_{40}
8	K
9	D
10	J
11	$MACD$
12	RSI_{14}
13	RSI_{SR}
14	RSI_{OV}
15	CCI_{UP}
16	KDJ_{UP}
17	$MACD_{UP}$

2.3.3 Evaluation metrics

Let's consider that we want to predict the patients with cancer, and we define y equals 1 if a patient has cancer and y equals 0 otherwise. Let's also say we've trained a classifier that outputs probabilities between 0 and 1. As usual, we are going to predict y equals 1 if the probability is greater than equal to 0.5 and predict 0 if the probability is less than 0.5. This classifier may give us some value for precision and some value for recall. We want to predict that the patient has cancer only if we are very confident that they do because if we tell a patient that he has cancer, this will give them a huge shock. Maybe we want to tell someone that we think they have cancer only if we are very confident. One way to do this would be to modify the algorithm so that instead of setting this threshold at 0.5, we might instead say they will predict that y is equal to 1 only if the probability is greater than or equal to 0.7. We are now more confident and end up with a classifier with higher precision because all of the patients we are going to and say we think they have cancer, they will.

In contrast, this classifier will have lower recall because now we are going to make predictions on a smaller number of patients. Now consider a different example and suppose we want to avoid missing too many actual cancer cases, so we want to avoid false negatives. In particular, if a patient has cancer, but we fail to tell them that they have cancer, that could be bad because they will not go for treatment. Let's suppose that when in doubt, we want to predict that y equals 1, so we predict that they have cancer. At least they look further into it. In this case, rather than setting a higher probability threshold, we might instead take a lower value, so maybe 0.3, and be more conservative. In this scenario, we would have a higher recall classifier because we're going to be correctly flagging a higher fraction of all patients to have cancer. Still, we will end up with lower precision because a higher fraction of the patients we said have cancer will not have cancer after all.

Now, let's go into the context of our problem and say that if we identify an upward trend that is actually a downward trend, we will get into a big mistake in a trading circumstance, so, even if we miss some upward trends, we might want to be very confident that the upward trends we have identified are actually upward trends even if we identify a less amount of them. Comparing this with our cancer problem analysis, we will want to set as a target a ***high precision - low recall*** model.

2.3.3.1 Precision/Recall Trade-off

If we consider that for each instance we compute a score based on a decision function, and if that score is greater than a threshold, it assigns the instance to the positive class, or else it assigns it to the negative class. Figure 2.4 shows a few digits positioned from the lowest score on the left to the highest score on the right. Suppose the decision threshold is positioned at the central arrow (between the two 5s): we will find 4 true positives (actual 5s) on the right of that threshold, and one false positive (actually a 6). Therefore, with that threshold, the precision is 80% (4 out of 5). But out of 6 actual 5s, the classifier only detects 4, so the recall is 67% (4 out of 6). Now if we raise the threshold (move it to the arrow on the right), the false positive (the 6) becomes a true negative, thereby increasing precision (up to 100% in this case), but one true positive becomes a false negative, decreasing recall down to 50%. Conversely, lowering the threshold increases recall and reduces precision.

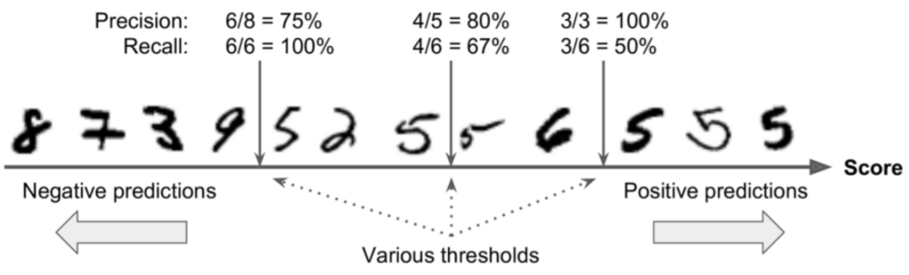

Figure 2.4: Decision threshold and precision/recall trade-off
Source: (Geron, 2017)

2.3.3.2 Decision Thresholds

An evaluation that considers the precision and recall as metrics must also take into account the decision limits for which a model can be set. To cover this, we will carry out a performance evaluation taking into account several thresholds that have been manually predefined in order to determine which is the necessary threshold to obtain a certain precision value.

2.4 Experimental Research Design

2.4.1 Entries

Binance API provides the options to acquire different candlesticks intervals: $1m$, $3m$, $5m$, $15m$, $30m$, $1h$, $2h$, $4h$, $6h$, $8h$, $12h$, $1d$, $3d$, $1w$, $1M$, from any specific timestamp, the API sets the limit to a max of 1000 candlesticks. We have selected three evaluation intervals: $3m$, $5m$, $15m$.

2.4.2 Making of Dataset

To get into the dataset, we have to calculate all features we defined in Table 2.1 for each candlesticks vector. With this, we will have an $\boldsymbol{X}$ dataset space coming from 1000 candlesticks elements. The length of each $\mathbf{x}_i$ vector is limited to the amount of past data needed to calculate the longer interval from all of the moving averages. In our case, the longer moving average is EMA_{40}. With this, we have a dataset of $length = 960$ that we will split 80% for training and 20% for testing. We also add extra data in which its first element is consecutive to the last $\mathbf{x}_i$ vector of the $\boldsymbol{X}$ dataset. This extra data is for validation purposes, and it will be the same length as the testing one Figure 2.5.

Table 2.2: Binance data origin

Interval (m)	**Initial Timestamp** $(epoch/Unix)$	**Cryptocurrency**
3	1588169700000	Ethereum
5	1588050000000	Ethereum
15	1587442500000	Ethereum

Table 2.2 shows the reference timestamps and the chosen cryptocurrency for each interval; this is particularly important due to the data that is used in this research is available through Binance API.

2.4.3 Features combinations

We use a total of 17 features 2.1. However, it is part of our interest to discover what are the minimum combinations of these features that allow us to obtain favorable results to the solution of the problem. This is important since our research focuses on micro trading with a dynamic update of models, which is why optimization of computational resources is necessary.

Figure 2.5 shows how we distribute the data. We will use tree intervals for our experiments: 3m, 5m, 15m. For all of the intervals we use share the same reference point representing the ”*present*”. However, for the shorter intervals, the covered time is also short, but the amount of values remains. We also use this same point of reference to define the starting point of the validation data.

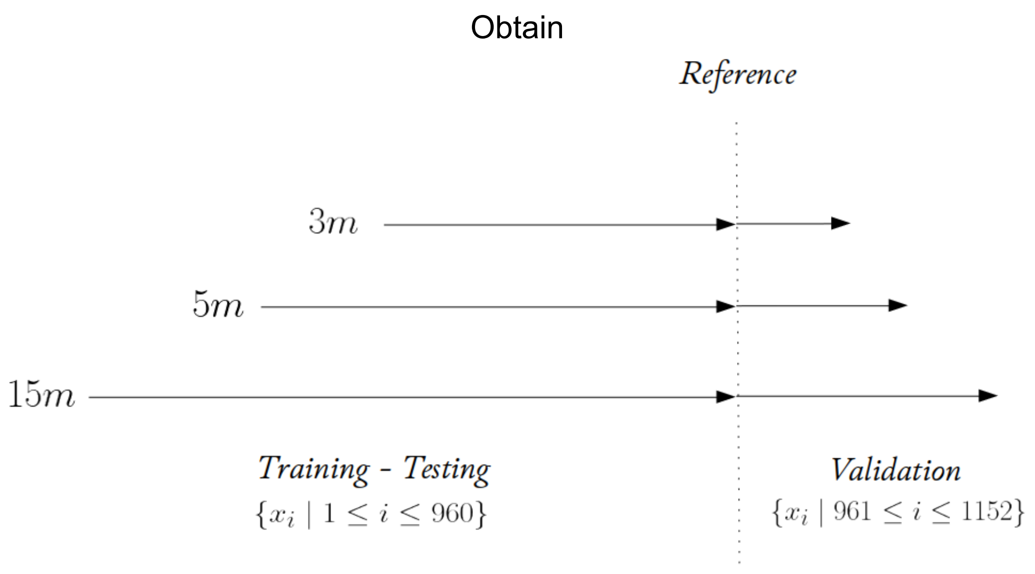
Obtain
Reference
$3m$
$5m$
$15m$
Training - Testing
$\{x_i \mid 1 \leq i \leq 960\}$
Validation
$\{x_i \mid 961 \leq i \leq 1152\}$

Figure 2.5: Dataset distribution

2.4.4 Materials and Instrumentation

We use Python Scikit-Learn v0.23 to scale feature vectors using (StandardScaler), train each model using (SVC), save models and scalers with (Pickle), this for being able to run new classifications using the same models and environments.

2.4.5 Experimental Procedure

The procedure that is defined in Figure 2.6, contemplates all the flow that we have designed to adequately complete all the necessary steps in the construction of the system that will allow us to carry out the experiments.

The strategy is defined by the following purposes:

1. Tag upward trend labels in data.
2. Select the kernel to use.
3. Select the regularization parameter to apply.
4. Identify the features to use.
5. Select one short time interval.

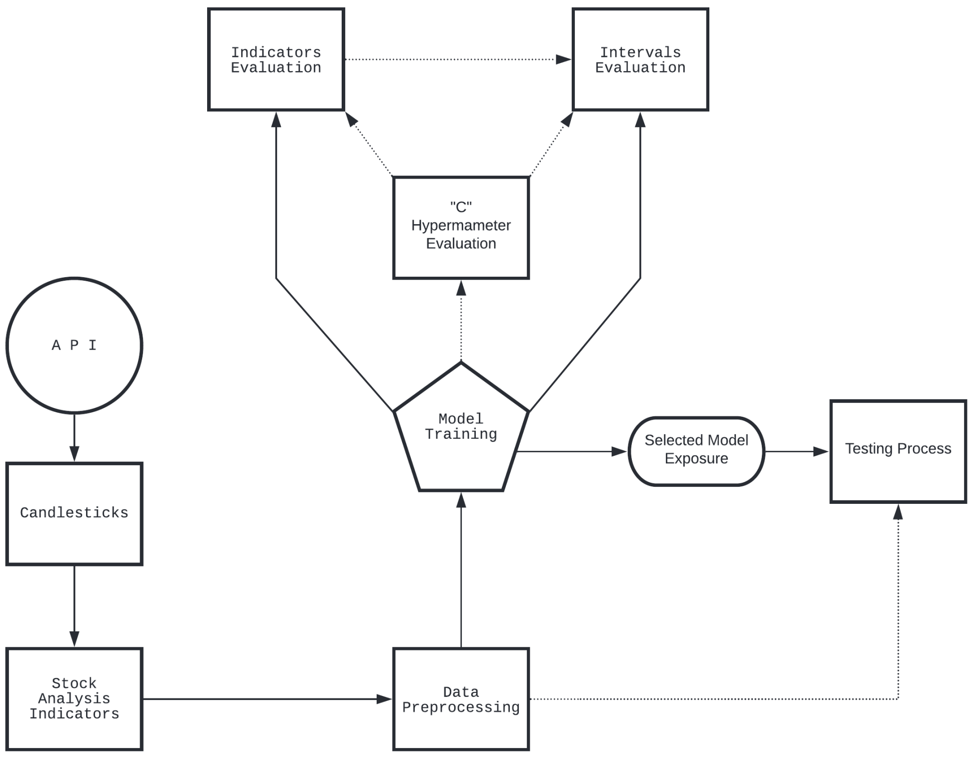

Figure 2.6: General experimental flow

2.4.5.1 Kernels evaluation

(Patel et al., 2014) perform a comparison between Polynomial and Radial Basis Kernel (RBF) in which the kernel that performs best is the **Polynomial Kernel**; in other researches, like in (Ballings et al., 2015) there is a comparison between Linear, Polynomial, Radial Basis and Sigmoid kernel and stands out that the **Radial Basis Kernel** performs better than Polynomial, followed by Sigmoid and Linear. (Ismail et al., 2020) focuses only in Radial Basis Kernel, (Shynkevich et al., 2015) performs a deeply kernel performance analysis using Polynomial, Radial Basis, and Linear kernels, obtaining better results with RBF and Polynomial kernels, this is why we choose Radial Basis and Polynomial kernels for our analysis.

Since we are going to work with the intervals of 3, 5, and 15 minutes, we choose 5 minutes to help us determine the most suitable kernel. The procedure begins by establishing all the unique combinations between the features vector and obtaining the confusion matrix of each of these evaluations using Support Vector Machine.

2.4.5.2 Regularization parameter evaluation

In conjunction with the kernel evaluation, we will use regularization parameters of $[0.1, 1, 10, 100, 1000]$ with each kernel. For all these experiments, we will use a Gamma Hyper-parameter ($Kernel\ Coefficient\ \gamma$) of $\gamma = 1/(\#features)$.

2.4.5.3 Intervals evaluation

Once the most suitable kernel and regularization parameters that best fit the problem are defined, we train models for 3 and 15 minutes with this selected kernel and parameters in such a way that we can establish a comparison.

2.4.5.4 Trading strategy definition

The solution to a problem using machine learning techniques is only validated by the adequate implementation in the field, which is why we propose to define a trading strategy that fits the modeling of this research.

Chapter 3

Experimental Results

3.1 Introduction

In this section, we show results in such a way that they are easy to assimilate for readers. We present a compilation of graphs, tables, and an explanation of the most relevant experimental results. All of the results follow the sequence that we defined in the methodology and in the research design. We show partial and total results, as well as outcomes with a comparison with similar works.

3.2 Results

In Table 3.1, we summarize the results of running different experiments by training SVM models with regularization parameter values of $[0.1, 1, 10, 100, 1000]$. These experiments use Polynomial and Radial Basis Kernels for 5 minutes interval for each unique features combinations from the feature vector attributes.

We can see in 3.1 that Radial Basis Kernel has the highest values of precision and recall with C Hyperparameter of 100 and 1000. This is why we proceed to train the remaining two intervals (3 minutes and 15 minutes) with same procedure but only using Radial Basis Kernel and C Hyperparameter of 100 and 1000.

Table 3.1: Kernels and C Hyper-parameter evaluation summary

Kernel	c	Precision Range	Recall Range	Figure
Polynomial	0.1	0.92 - 0.96	0.32 - 0.38	3.1
Radial Basis	0.1	0.89 - 0.93	0.32 - 0.38	3.2
Polynomial	1	0.86 - 0.89	0.35 - 0.46	3.3
Radial Basis	1	0.85 - 0.97	0.64 - 0.7	3.4
Polynomial	10	0.81 - 0.83	0.45 - 0.48	3.5
Radial Basis	10	0.86 - 0.92	0.67 - 0.72	3.6
Polynomial	100	0.76 - 0.88	0.47 - 0.59	3.7
Radial Basis	100	0.92 - 0.96	0.70 - 0.72	3.8
Radial Basis	1000	0.90 - 0.93	0.68 - 0.75	3.10

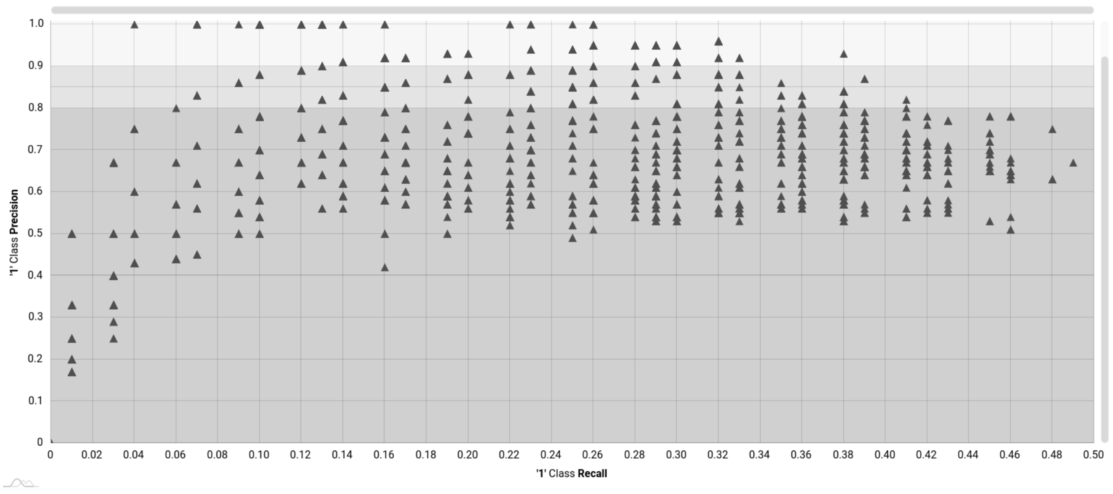

Figure 3.1: Polynomial Kernel, $c = 0.1$

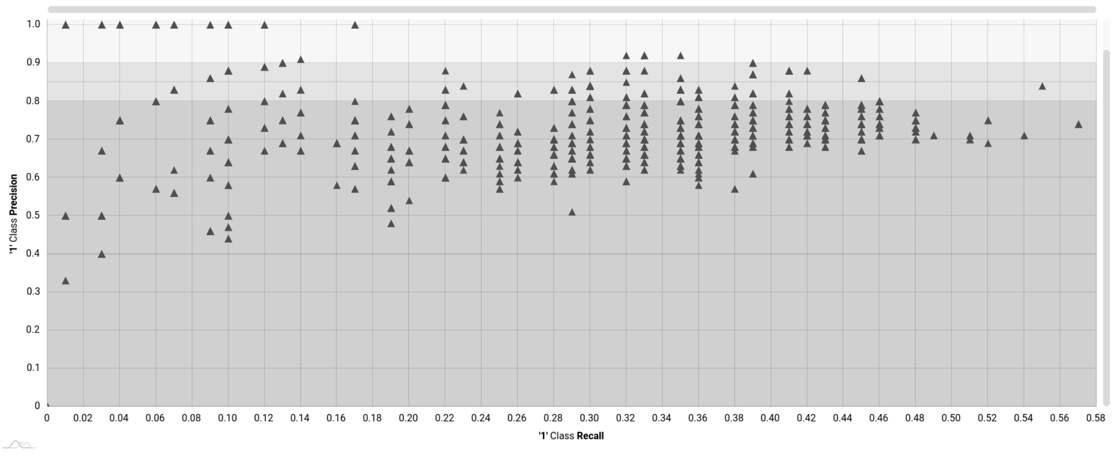

Figure 3.2: Radial Basis Function Kernel, $c = 0.1$

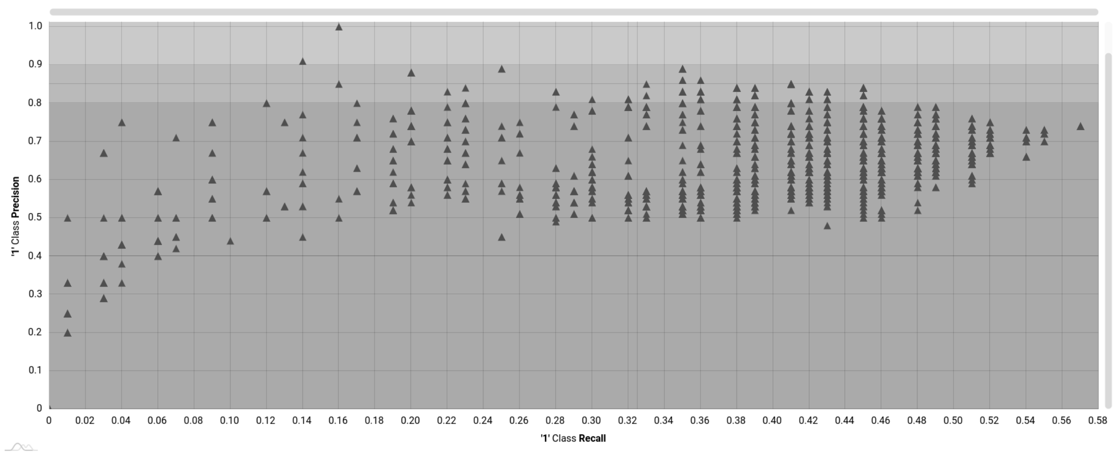

Figure 3.3: Polynomial Kernel, $c = 1$

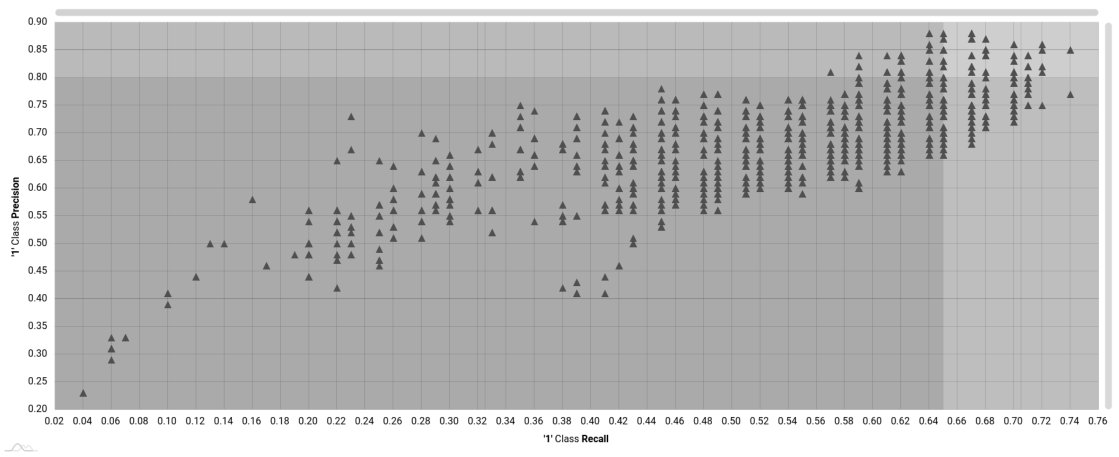

Figure 3.4: Radial Basis Function Kernel, $c = 1$

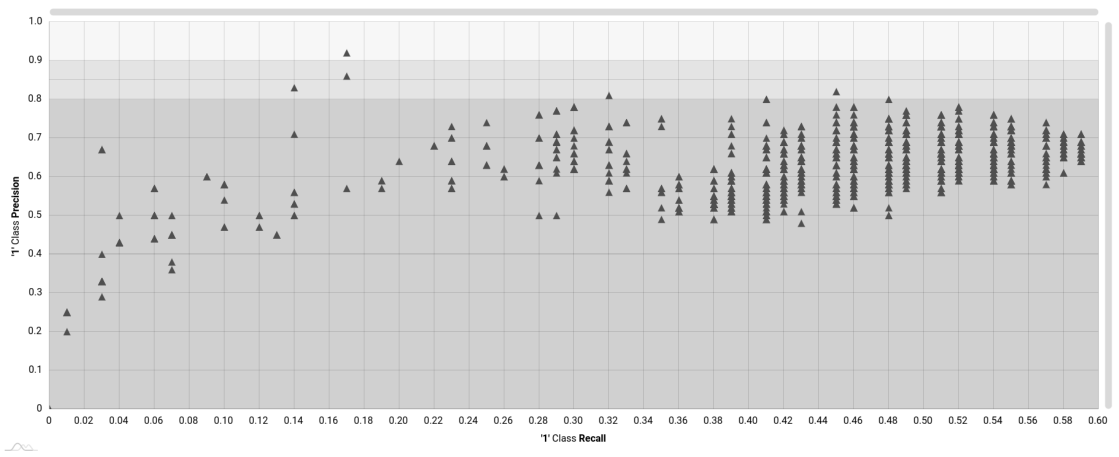

Figure 3.5: Polynomial Kernel, $c = 10$

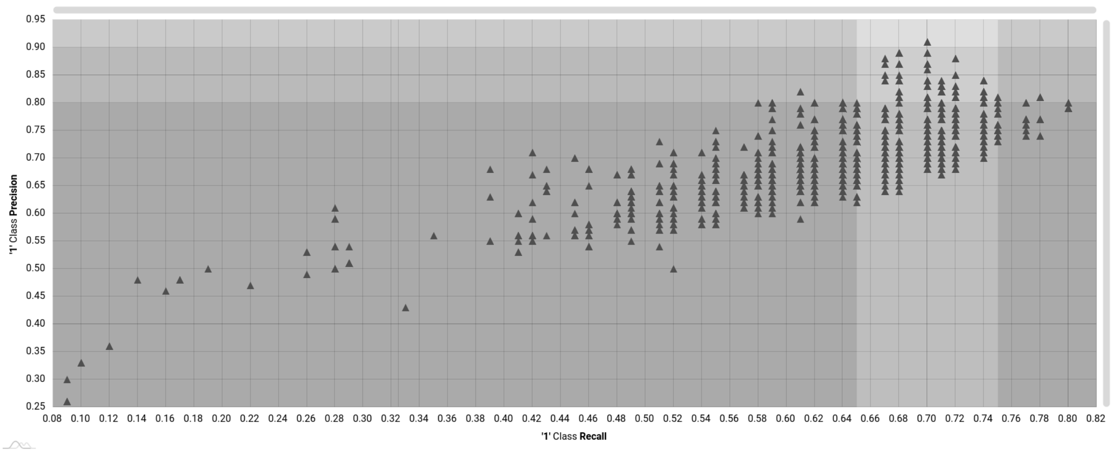

Figure 3.6: Radial Basis Function Kernel, $c = 10$

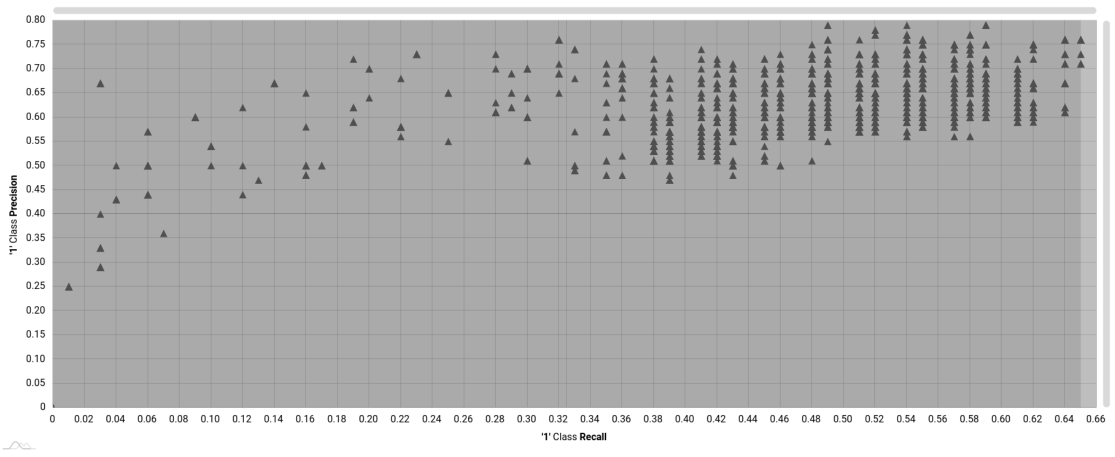

Figure 3.7: Polynomial Kernel, $c = 100$

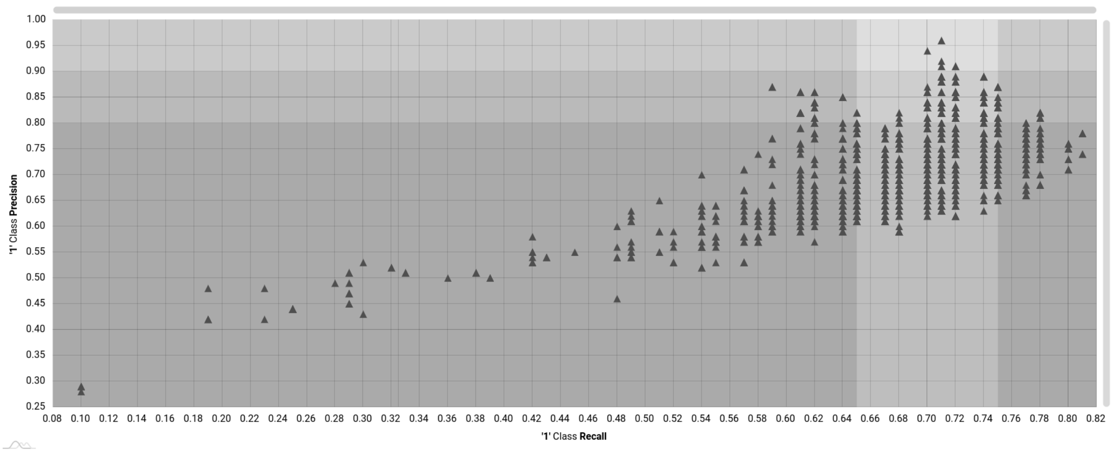

Figure 3.8: Radial Basis Function Kernel, $c = 100$

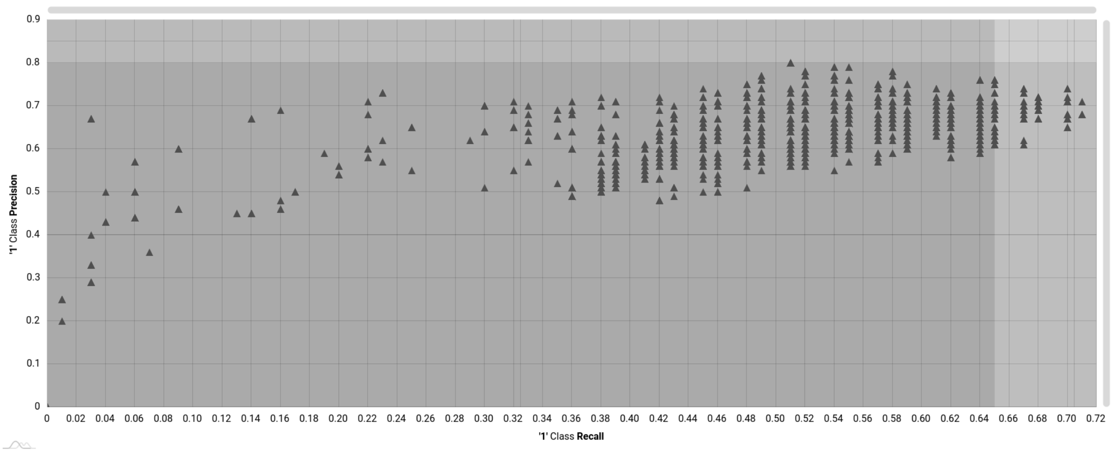

Figure 3.9: Polynomial Kernel, $c = 1000$

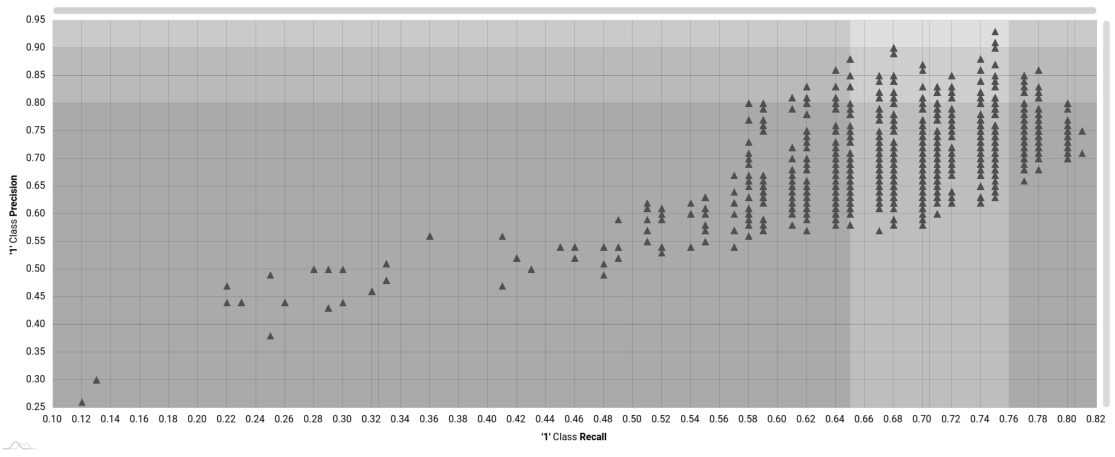

Figure 3.10: Radial Basis Function Kernel, $c = 1000$

Table 3.2: Most suitable indicators with $c = 100$, and $c = 1000$

Interval	c	**Precision 0**	**Recall 0**	**Precision 1**	**Recall 1**	EMA_7	EMA_{40}	RSI	$HighPrice$	$MACD$
3m	100	0.87	0.91	0.83	0.77	✓		✓	✓	✓
3m	100	0.83	0.98	0.96	0.66	✓	✓	✓	✓	
3m	100	0.87	0.92	0.84	0.76	✓	✓	✓	✓	✓
3m	1000	0.87	0.92	0.85	0.77	✓	✓	✓	✓	
5m	100	0.86	0.96	0.91	0.71	✓		✓	✓	✓
5m	100	0.86	0.98	0.96	0.71	✓	✓	✓	✓	
5m	100	0.86	0.97	0.92	0.71	✓	✓	✓	✓	✓
5m	1000	0.88	0.97	0.93	0.75	✓	✓	✓	✓	
15m	100	0.82	0.92	0.82	0.64	✓		✓	✓	✓
15m	100	0.78	0.86	0.71	0.59	✓	✓	✓	✓	
15m	100	0.82	0.92	0.82	0.66	✓	✓	✓	✓	✓
15m	1000	0.79	0.85	0.7	0.61	✓	✓	✓	✓	

We present in Table 3.2 the combination of technical indicators with the highest values of precision and recall using both C Hyperparameters candidates (100 and 1000) for each of the labeled classes. We can see that for all of the tree intervals the most suitable features result to be EMA_7, EMA_{40}, RSI, $HighPrice$ and $MACD$. Those outcomes that have the highest values for precision and recall for each interval are highlighted.

3.2.1 Threshold selection

After identifying the most suitable feature indicators and C Hyperparameter for each interval in Table 3.2, we have to go through the definition of the most appropriate threshold. We have to handle the probability inside the decision boundary and establish different evaluations for different thresholds, the following figures show $(Precision - Recall)$, and $PrecisionThresholds$ plots for each of the highlighted outcomes of Table 3.2

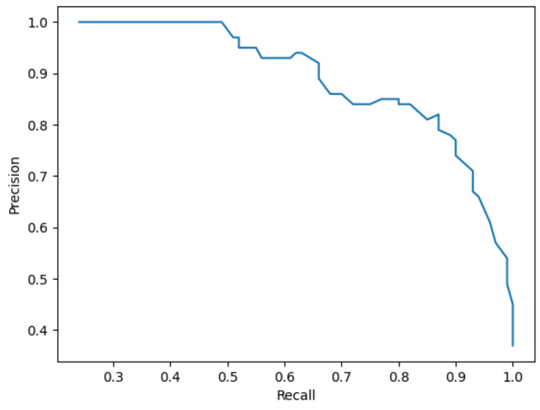

Figure 3.11: Precision Recall Curve $3m, c = 1000$

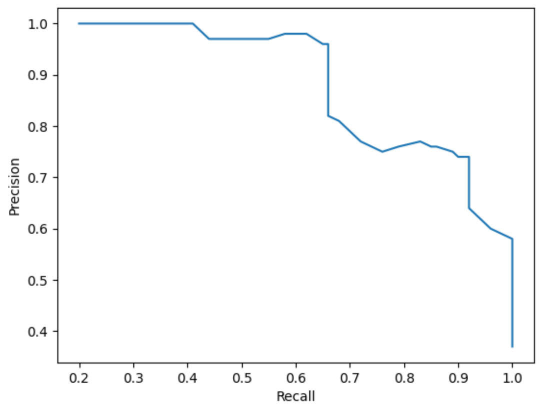

Figure 3.12: Precision Recall Curve $3m, c = 100$

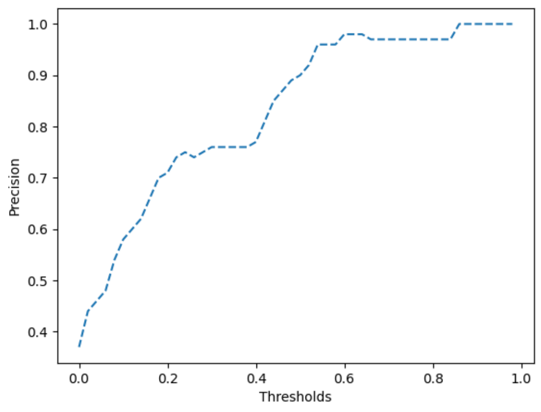

Figure 3.13: Precision Thresholds Curve $3m, c = 100$

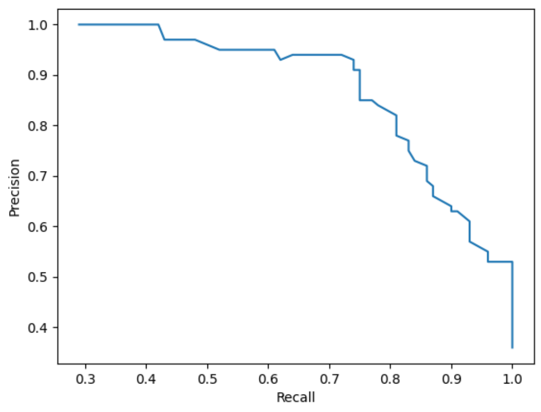

Figure 3.14: Precision Recall Curve $5m, c = 1000$

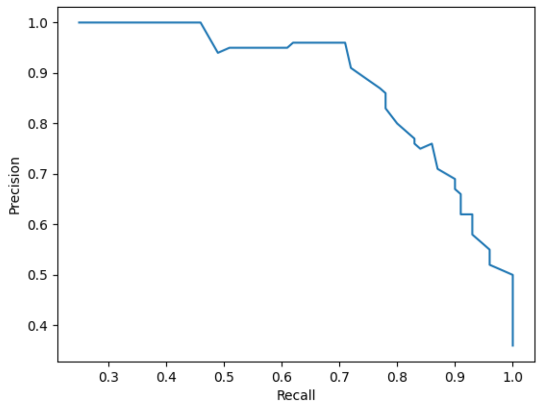

Figure 3.15: Precision Recall Curve $5m, c = 100$

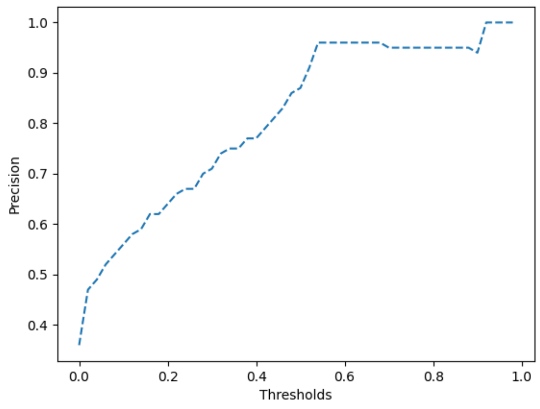

Figure 3.16: Precision Thresholds curve $5m, c = 100$

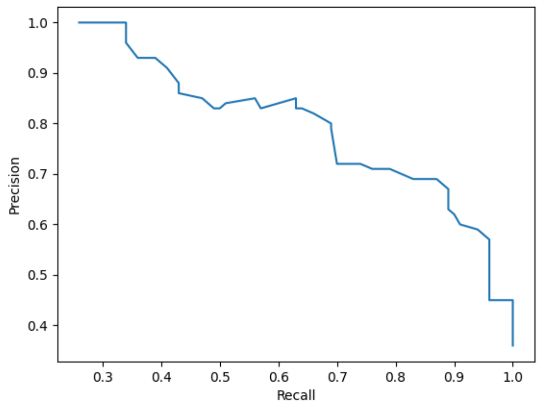

Figure 3.17: Precision Recall Curve $15m, c = 100$

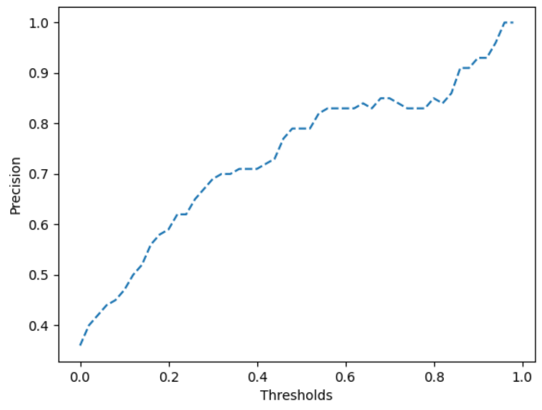

Figure 3.18: Precision Thresholds curve $15m, c = 100$

We can see in Figure 3.17 how the 15 minutes interval has the lower performance comparing with 3 minutes in Figure 3.11 and 5 minutes in Figure 3.15. This two figures stand out maintaining a comparable high precision with acceptable values of recall. The outcome is that the best performing regularization parameter In terms of the relation between precision and recall is $c = 100$ with a Radial Basis Kernel.

Table 3.3: Overview with similar studies

Ref	**Period**	**Metrics**	**Output**	**Technique**	**SVM Result**	**Kernel**	c
(Shynkevich et al., 2015)	1 day	Acc	U/D	KL	79.59	RBF	100
(Patel et al., 2014)	1 day	Acc/F-1	U/D	SVM	0.92/0.92	Poly	1
(Ballings et al., 2015)	1 month	AUC	U/D	KL	0.8395	-	-
(Lee et al., 2019)	1 week	Acc	U/D	LR	81.6%	RBF	10
(Ismail et al., 2020)	1 day	Acc	U/D	RF	76.53%	RBF	-
This research	5m	AP	U/O	SVM	0.91/0.845	RBF	100

-**U/D**- stands for UP/DOWN predictions, -**U/O**-: stands for UP/(AND OTHERS OUTPUTS)

Indeed, a comparison with other studies that use different time intervals, diverse data sets, and algorithms can lead to confusion. However, it is important to note that the problem to be solved is predicting the trend direction. (Patel et al., 2014) is the work that uses SVM with very encouraging results. This work uses ten indicators and points out that a polynomial kernel with a regularization parameter of 1 is the one that best fits its study, which differs from what we have obtained. We use RBF Kernel with a regularization parameter of 100. It should be noted that the present study we use four indicators as a result of the analysis within the proposed methodology.

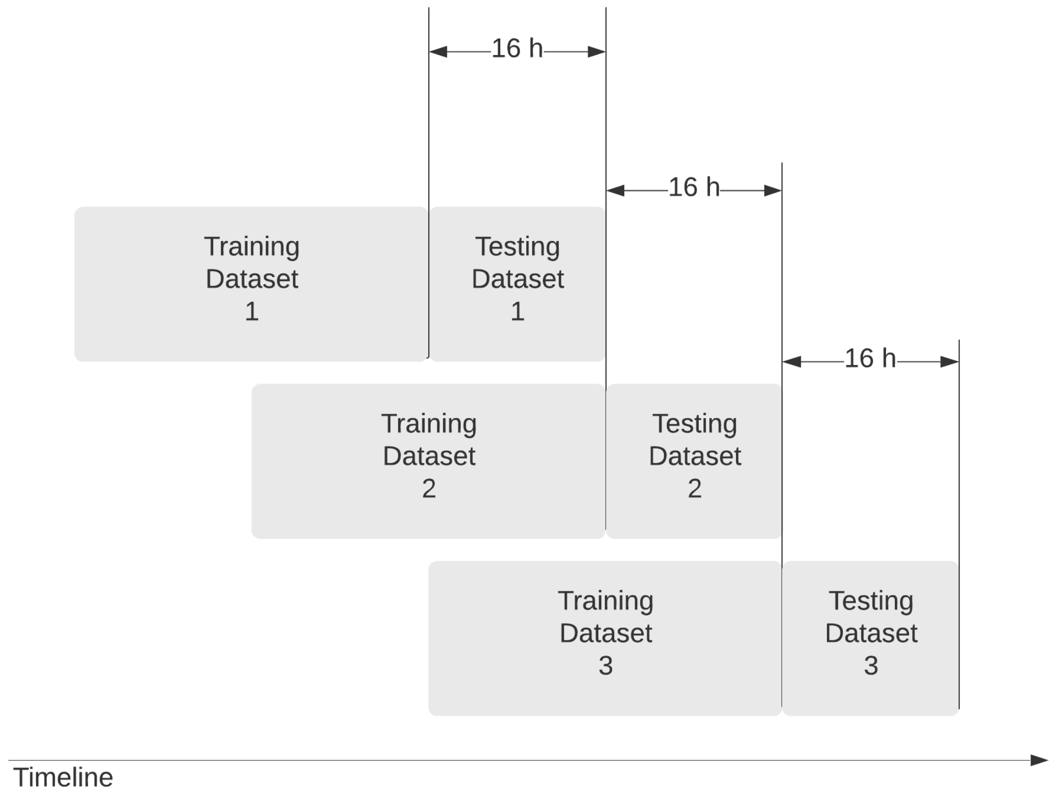

Figure 3.19: Progressive training and testing definition

3.2.2 Testing

In Figure 3.19 we outline a progressive training and testing definition to update the model every 16 hours, which corresponds to the size description of our test data set in Figure 2.5. With this schema, we can obtain a simulation of the progressive operation of 48 hours.

In Figure 3.20, we can additionally see an example of one of the testing runs. The classification seems to work better for the samples that are closest to the training dataset time-frame, and performance decreases gradually the further it moves away from the model training data. This behavior is the main reason why this research focuses on short-term intraday intervals and why a proper trading strategy could only work if we continuously update the model.

In Figure, the lighter tonality area defines a time period when the model works reasonably well (predicting most of the upward trends). However, this behavior changes in the second region, where the prediction is targeting a short upward trend.

We have added some additional data to show what happens over time while using the same trained model. As time progresses inside the darker area things are not encouraging for the trained model. We can see how the model has lost the context and has predictions very far from what we would expect, appearing in nonsensical predictions.
This is one reason why a model trained in a volatile environment cannot be subject to generalization in extended periods since it does not have long-term validity. It is only adequate for running in close to the training period.

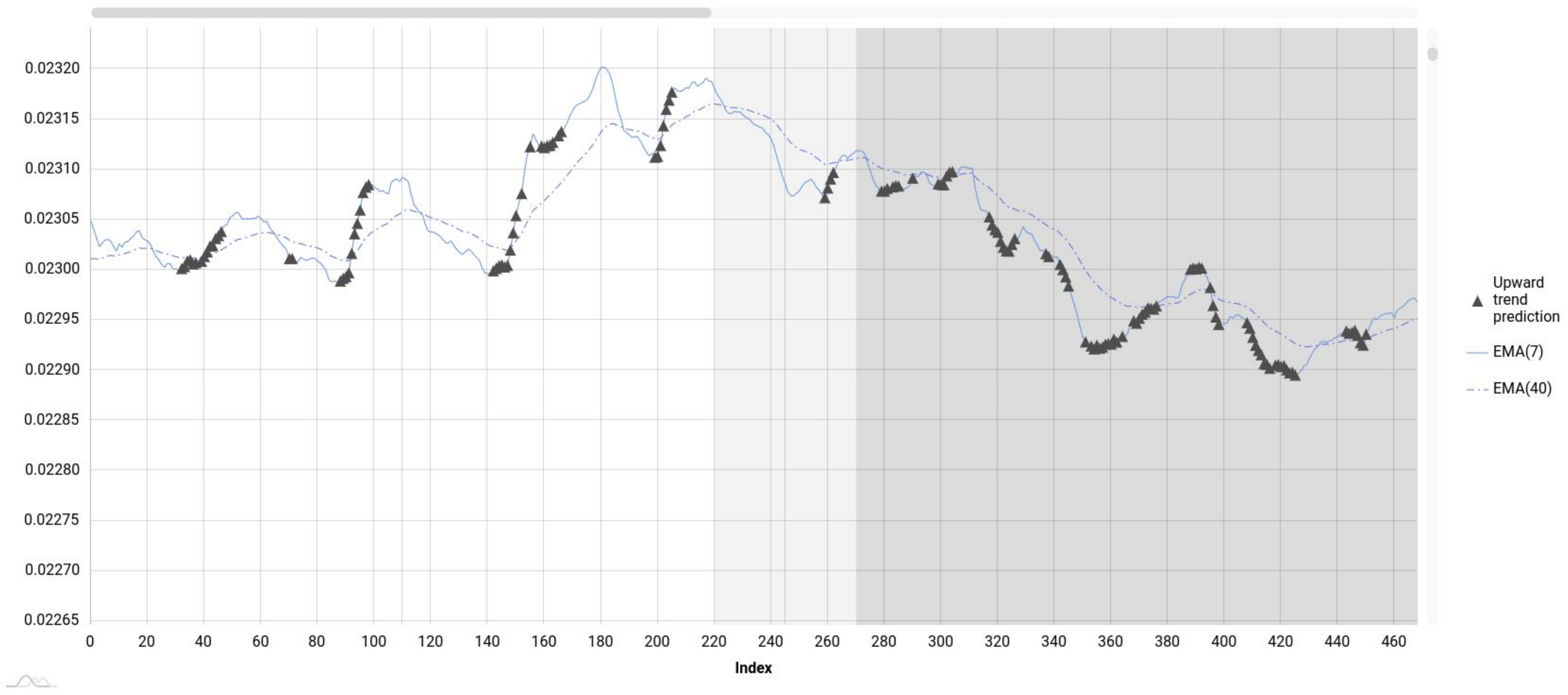

Figure 3.20: Single testing run example

In Table 3.4 we group the results obtained in the testing stage for each of the datasets defined in Figure 3.19. For this stage we use $5m$ interval, RBF Kernel with C Hyperparameter of 100, and a threshold of 0.8.

Table 3.4: Testing results

Dataset	**Interval**	c	**Kernel**	**Precision 0**	**Recall 0**	**Precision 1**	**Recall 1**
1	5m	100	RBF	0.83	0.98	0.92	0.26
2	5m	100	RBF	0.81	0.98	0.91	0.37
3	5m	100	RBF	0.84	0.98	0.94	0.27

3.2.3 Trading strategy

The advantage of carrying out a continuous evaluation in time stands out when we have a purchase decision that allows us to work on an *”active buy and hold”* strategy defined as the setting of a profit limit to which we promise to sell, regardless of what happens in the future. With this strategy, we can take advantage of a considerably high number of trading operations with sustained profit.

It is also necessary to define a sale decision logic when the desired profit ceiling is not obtained, and it could be defined with something as simple as a ”stop-loss”.

Figure 3.20 shows how we can define a trading strategy that consists of identifying a fixed number of consecutive or ”close-enough” positive predictions and trigger a buy action to hold till achieving a ceil percentage profit.

If we use the predicted upward trend directions in figure 3.20, (in the lighter shaded region between indexes 0 to 221), we can define the strategy: buy after having three consecutive or semi consecutive upward predictions and sell when a 0.3% of profit has been reached. If we represent each operation in such a way that describes a buying action with buying price at i: we will get an accumulative profit calculated on the relation between each buying and selling action, each one with 0.3% of profit. In our testing data the buying actions would be triggered at indexes $[34, 92, 148, 201]$, giving us a cumulative profit of 1.13% within 16,66 hours and is important to highlight that this strategy can be performed in the context of a global reduction of stock pricing, but taking advantage of smaller changes.

Chapter 4

Conclusions

4.1 Introduction

Stock market evaluation is a high-interest task developed over time, and due to this, over the years, a considerable number of methodologies, strategies, and evaluation mechanisms have been proposed trying to get the most out of it. However, this high interest and profit aspirations, have caused many of the theories or strategies to be challenging to verify and many of them must meet particular requirements, this is why this research bases its study on trying to establish a relative small problem, by defining a strategy presenting a clear solution and an implementation path so that it can be: evaluated, replicated, accepted or rejected by future works that try to onboard similar problems.

4.2 Research Goals Analysis: Summary of Findings and Conclusions

It is well known by people who dedicate their efforts to work in a stock market analysis that machine learning algorithms are powerful tools that give them an additional instrument that helps in getting the most out of the stock market. However, the methodologies, algorithms, implementations, and evaluation techniques play essential roles in obtaining the expected outcomes or at least approaching them. All of this is why our clear intention is to use one machine learning algorithm with a clear tagging strategy. We focus in a tight scope inside the universe of problems that any stock market analysis has. We intend to solve a small problem as part of a more robust complementary strategy that we can define in future analyses.

Goal: To properly tag upward stock price trends from preprocessing candlesticks data and choose the financial indicators that better describe this phenomenon.

A good part of this research effort has been to establish a detailed methodology of data tagging. A particular interest directs us to place with an appropriate precision what we are willing to predict (profitable upward trend directions), as shown in Figure 2.3. Regardless of the algorithms to be used, this is an essential part without which we could not define the phenomenon adequately having a high impact on the results and the evaluation metrics.

Goal: To choose the trading strategy threshold that has the potential to enhance the total financial profit.

Since the stock market has an intrinsic characteristic of volatility, we have identified in Figure 3.20 that the usage of short evaluation intervals has the advantage of minimizing risks on two main fronts. The first one is the reduction of the risk of high losses since the buy, hold, or sale operations are in the context of short time intervals, the second is that the changes that are influenced by volatility factors have a low impact in the domain of minutes or hours, thus allowing us to reduce the degree of social influences or speculation.

Goal: To properly analyze the performance of the selected machine learning technique in different short-term time intervals to predict the upward trend direction of stock prices.

We summarize in Table 3.2 the results of the training process for the three evaluation intervals, Figure 3.15 and Figure 3.16 help us to confirm that the best performing interval is 5 minutes, followed by 3 minutes. Within this study context we have found that a 15 minutes interval is not a good fit for this analyses, and we infer that larger intervals than 15 minutes, would not also be a good fit.

Goal: To evaluate the achieved model in terms of training process and trading performance.

In the strict context of this research, one of the most outstanding discoveries is the considerably low number of financial indicators necessary to characterize the behavior of a trend direction, since many of the related works group a large number of them that turns out not to be necessary. (Patel et al., 2014), for example, uses 10 indicators; however, this research has found that four are enough.

The present work outcomes have allowed us to determine a trading strategy, which allows us to obtain sustained profits with a lower risk of loss, even when the market is slightly falling.

We propose a methodology that targets continuous model updates, and we have validated this idea in the testing process of this research, where we verify that this strategy is a good alternative when we are in a domain in which the conditions are constantly changing.

We obtained an average precision of 0.91 and an average recall of 0.845 during the training stage, which stands good for a general perspective but is not enough to make conclusions. This is why we performed a testing process, showing that the class of our interest can reach an average precision of 0.92 and an average recall of 0.3 with a decision threshold of 0.8: this particular result of high precision and low recall let us be more confident that the majority of ”buy actions” will drive us to a profitable upward trend.

4.3 Recommendations and Future Works

One important limitation of this strategy is that it gets affected by big variations in the price of Bitcoin against the Dollar. One approach could it be to explore options like in (Segovia, Fernández-Martínez, & Sánchez-Granero, 2019) to identify high volatility and a better understanding of the high-level perspective in each market context.

We also have to explore options that complement this research with a selling strategy that improves gains.

It is important not to consider this strategy as sufficient. There are parallel analyses that must be carried out. This technique can be implemented in reality when it has complementary tactics. We have more challenges since identifying a stock market upward trend direction is only a part of a bigger problem.

Appendix A

Appendix

A.1 Data plots

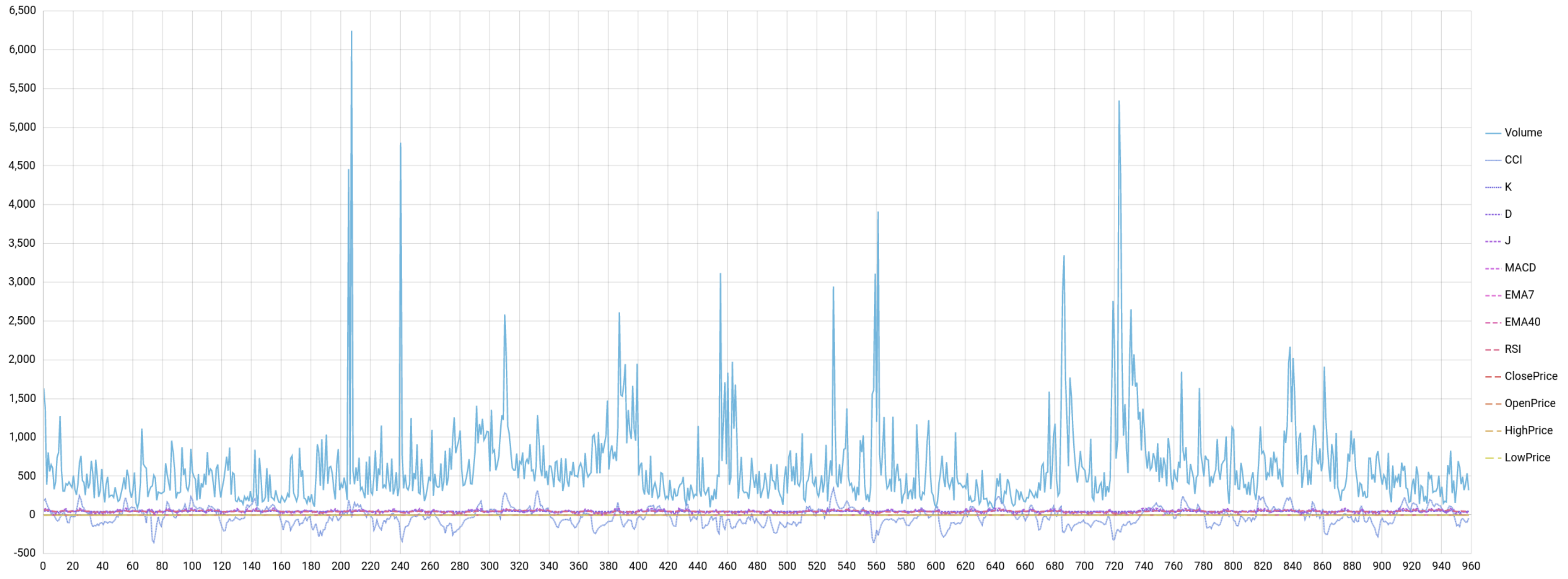

Figure A.1: Sample of non-scaled grouped testing data financial and indicators

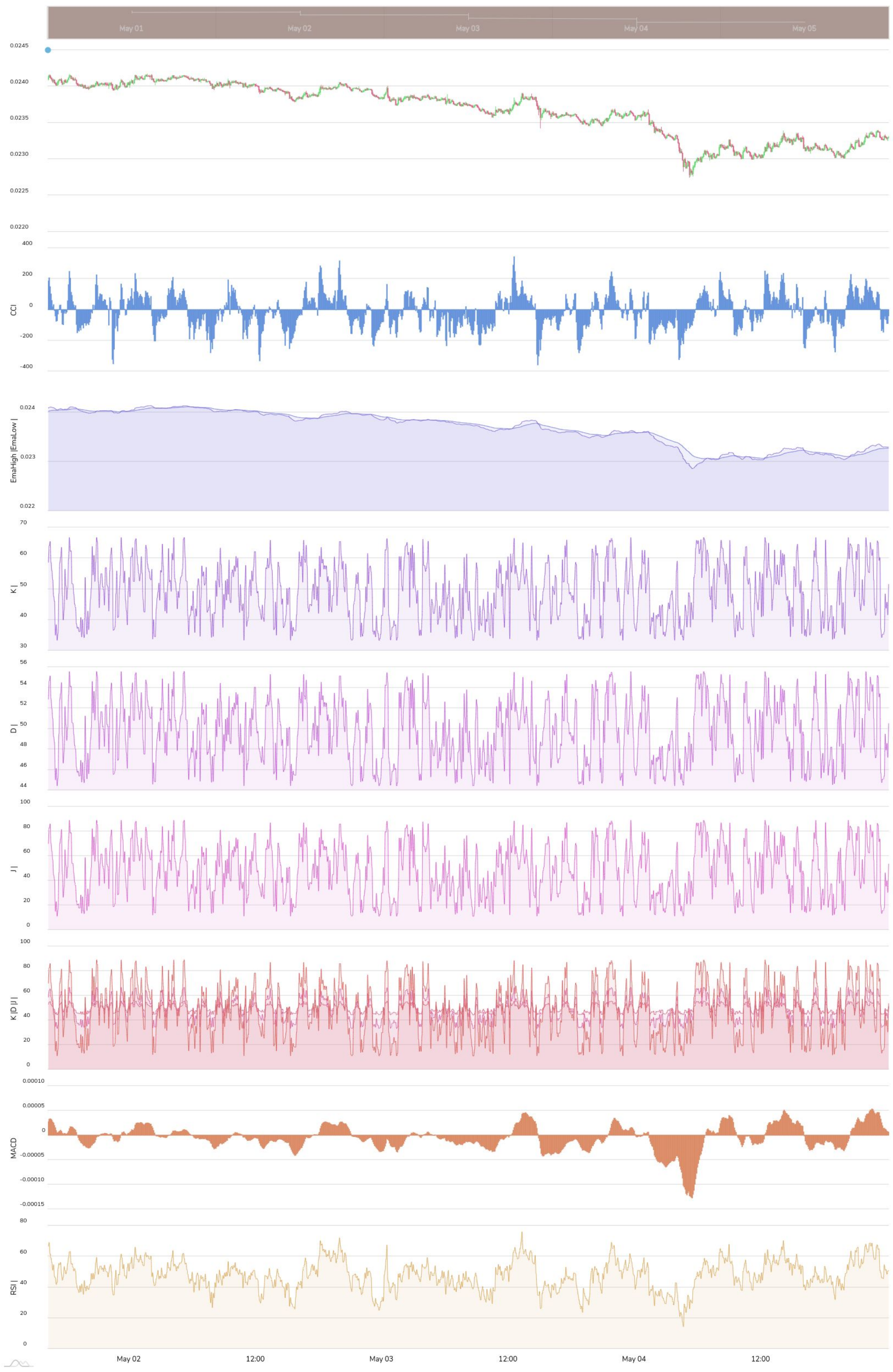

Figure A.2: Sample of non-scaled single testing data indicators

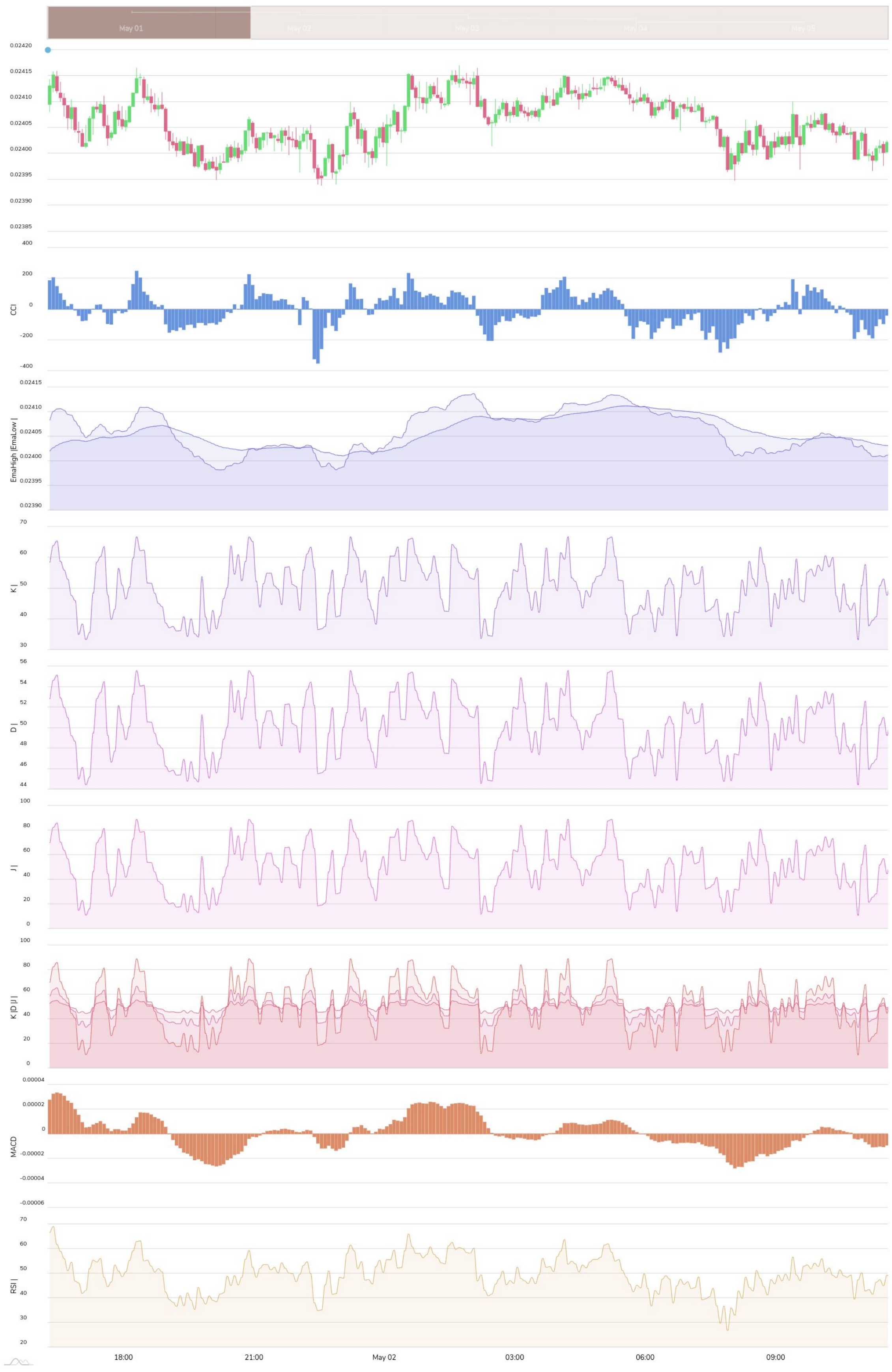

Figure A.3: Sample of non-scaled single closeup testing data indicators

Bibliography

Achelis, S. B. (2001). Technical analysis from a to z. McGraw Hill New York.

Althelaya, K. A., El-Alfy, E.-S. M., & Mohammed, S. (2018). Evaluation of bidirectional lstm for short-and long-term stock market prediction, In *2018 9th international conference on information and communication systems (icics)*. IEEE.

Ballings, M., Van Den Poel, D., Hespeels, N., & Gryp, R. (2015). Evaluating multiple classifiers for stock price direction prediction. *Expert Systems with Applications*, *42*(20), 7046–7056. https://doi.org/10.1016/j.eswa.2015.05.013

Bernal, A., Fok, S., & Pidaparthi, R. (2012). Financial market time series prediction with recurrent neural networks. *State College: Citeseer.[Google Scholar]*.

Bisoi, R., & Dash, P. K. (2014). A hybrid evolutionary dynamic neural network for stock market trend analysis and prediction using unscented kalman filter. *Applied Soft Computing*, *19*, 41–56.

Carmona, P., Climent, F., & Momparler, A. (2019). Predicting failure in the us banking sector: An extreme gradient boosting approach. *International Review of Economics & Finance*, *61*, 304–323.

Chang, J., Jung, Y., Yeon, K., Jun, J., Shin, D., & Kim, H. (1996). Technical indicators and analysis methods. *Seoul: Jinritamgu Publishing*.

Dey, S., Kumar, Y., Saha, S., & Basak, S. (2016). Forecasting to classification: Predicting the direction of stock market price using xtreme gradient boosting. *PESIT, Bengaluru, India, Working Paper*.

Di Persio, L., & Honchar, O. (2017). Recurrent neural networks approach to the financial forecast of google assets. *International journal of Mathematics and Computers in simulation*, *11*, 7–13.

Geron, A. (2017). *Hands-on machine learning with scikit-learn and tensorflow: Concepts, tools, and techniques to build intelligent systems* (1st). O'Reilly Media, Inc.

Ghosh, I., Jana, R. K., & Sanyal, M. K. (2019). Analysis of temporal pattern, causal interaction and predictive modeling of financial markets using nonlinear dynamics, econometric models and machine learning algorithms. *Applied Soft Computing*, *82*, 105553.

Hafezi, R., Shahrabi, J., & Hadavandi, E. (2015). A bat-neural network multi-agent system (bnnmas) for stock price prediction: Case study of dax stock price. *Applied Soft Computing*, *29*, 196–210.

Hossain, M. A., Karim, R., THulasiram, R., Bruce, N. D., & Wang, Y. (2018). Hybrid deep learning model for stock price prediction, In *2018 ieee symposium series on computational intelligence (ssci)*. IEEE.

Ismail, M. S., Md Noorani, M. S., Ismail, M., Abdul Razak, F., & Alias, M. A. (2020). Predicting next day direction of stock price movement using machine learning methods with persistent homology: Evidence from Kuala Lumpur Stock Exchange. *Applied Soft Computing Journal*, *93*, 106422. https://doi.org/10.1016/j.asoc.2020.106422

Ji, T., Che, W., & Zong, N. (2014). Stock market forecast based on rbf neural network, In *Practical applications of intelligent systems*. Springer.

Lee, T. K., Cho, J. H., Kwon, D. S., & Sohn, S. Y. (2019). Global stock market investment strategies based on financial network indicators using machine learning techniques. *Expert Systems with Applications*, *117*, 228–242.

Mai, F., Shan, Z., Bai, Q., Wang, X., & Chiang, R. H. (2018). How does social media impact bitcoin value? a test of the silent majority hypothesis. *Journal of Management Information Systems*, *35*(1), 19–52.

Mallqui, D. C., & Fernandes, R. A. (2019). Predicting the direction, maximum, minimum and closing prices of daily bitcoin exchange rate using machine learning techniques. *Applied Soft Computing*, *75*, 596–606.

Milosevic, N. (2016). Equity forecast: Predicting long term stock price movement using machine learning. *arXiv preprint arXiv:1603.00751*.

Murphy, J. (1999). *Technical analysis of the financial markets: A comprehensive guide to trading methods and applications*. Prentice Hall PTR. https://books.google.com.ec/books?id=lV-ZAAAACAAJ

Nakamoto, S. (2019). *Bitcoin: A peer-to-peer electronic cash system* (tech. rep.). Manubot.

Patel, J., Shah, S., Thakkar, P., & Kotecha, K. (2014). Expert Systems with Applications Predicting stock and stock price index movement using Trend Deterministic Data Preparation and machine learning techniques. *EXPERT*

SYSTEMS WITH APPLICATIONS, (August). https://doi.org/10.1016/j.eswa.2014.07.040

Ribeiro, M. H. D. M., & dos Santos Coelho, L. (2020). Ensemble approach based on bagging, boosting and stacking for short-term prediction in agribusiness time series. *Applied Soft Computing*, *86*, 105837.

Segovia, J. T., Fernández-Martínez, M., & Sánchez-Granero, M. (2019). A novel approach to detect volatility clusters in financial time series. *Physica A: Statistical Mechanics and its Applications*, *535*, 122452.

Shynkevich, Y., McGinnity, T. M., Coleman, S., & Belatreche, A. (2015). Stock price prediction based on stock-specific and sub-industry-specific news articles, In *2015 international joint conference on neural networks (ijcnn)*. IEEE.

Sincere, M. (2010). *All about market indicators*. McGraw-Hill Education. https://books.google.com.ec/books?id=xUPwxB-1LvsC

Yang, B., Gong, Z.-J., & Yang, W. (2017). Stock market index prediction using deep neural network ensemble, In *2017 36th chinese control conference (ccc)*. IEEE.

Żbikowski, K. (2015). Using volume weighted support vector machines with walk forward testing and feature selection for the purpose of creating stock trading strategy. *Expert Systems with Applications*, *42*(4), 1797–1805.

Zhang, J., Cui, S., Xu, Y., Li, Q., & Li, T. (2018). A novel data-driven stock price trend prediction system R. *Expert Systems With Applications*, *97*, 60–69. https://doi.org/10.1016/j.eswa.2017.12.026

Zhou, F., Zhou, H.-m., Yang, Z., & Yang, L. (2019). Emd2fnn: A strategy combining empirical mode decomposition and factorization machine based neural network for stock market trend prediction. *Expert Systems with Applications*, *115*, 136–151.