EXTRACTING CRYPTOCURRENCY TRADING SIGNALS FROM RAW FINANCIAL DATA USING TECHNICAL INDICATORS AND DEEP LEARNING

BAŞAK KALFA

BOĞAZİÇİ UNIVERSITY

2020

EXTRACTING CRYPTOCURRENCY TRADING SIGNALS FROM RAW FINANCIAL DATA USING TECHNICAL INDICATORS AND DEEP LEARNING

Thesis submitted to the

Institute for Graduate Studies in Social Sciences

in partial fulfillment of the requirements for the degree of

Master of Arts

in

Management Information System

by

Başak Kalfa

Boğaziçi University

2020

Extracting Cryptocurrency Trading Signals From Raw Financial Data

Using Technical Indicators and Deep Learning

The thesis of Başak Kalfa

has been approved by:

Assist. Prof. Ahmet Onur Durahim _____
(Thesis Advisor)

Prof. Ceylan Onay Şahin _____

Prof. Ali Rana Atılgan _____
(External Member)

August 2020

DECLARATION OF ORIGINALITY

I, Başak Kalfa, certify that

- I am the sole author of this thesis and that I have fully acknowledged and documented in my thesis all sources of ideas and words, including digital resources, which have been produced or published by another person or institution;
- this thesis contains no material that has been submitted or accepted for a degree or diploma in any other educational institution;
- this is a true copy of the thesis approved by my advisor and thesis committee at Boğaziçi University, including final revisions required by them.

Signature.....

Date .....22.09.2020.....

ABSTRACT

Extracting Cryptocurrency Trading Signals from Raw Financial Data Using Technical Indicators and Deep Learning

The purpose of this study is to investigate deep learning methods for predicting cryptocurrency, specifically Bitcoin, trading decisions considering different oversampling and augmentation methods, and using different feature groups as input, extracted using technical indicators. For this purpose, financial time series data is labelled as three classes namely, buy, sell, hold, by a window sliding approach. However, this labeling approach gives rise to imbalanced data set which is problematic in deep learning setting. In order to address imbalanced data set problem, three oversampling methods which are Direct Copying, SMOTE, ADASYN, and two augmentation methods, jittering and time warping, are used for increasing the size of data sets that belong to minority classes. Moreover, technical indicators are included in this study to extract financial data features to be fed into deep learning models. Feature selection and ablation study are performed on the extracted features. Six different feature groups are created which can be listed as; all indicators, top 15 indicators selected using mutual information metric, and momentum, volume, volatility and trend indicators. The proposed deep learning architectures, convolutional neural network and attention mechanisms with two different encoder architectures, are trained with different combinations of oversampling and augmentation methods, and feature groups. Results are promising where the best macro average F1 score of 56.48 is achieved by the model generated using attention deep learning architecture which is fed with top 15 indicators where SMOTE oversampling technique is applied.

ÖZET

Teknik Göstergeler ve Derin Öğrenme Kullanarak İşlenmemiş Finansal Verilerden Kripto Para Alım ve Satım Sinyalleri Elde Etmek

Bu çalışmanın amacı, teknik göstergeler ile elde edilen farklı özellik grupları ile farklı örnekleme ve çoğaltma yöntemlerini kullanarak kripto para alım-satım hareketlerinin, özellikle Bitcoin alım-satım hareketlerinin tahmini için derin öğrenme yöntemlerini araştırmaktır. Bu amaçla, finansal zaman serisi verileri pencere kaydırma algoritması ile üç sınıfa ayrılmıştır: satın al, sat, beklet. Bu sınıflandırma algoritması, derin öğrenme ortamında bir problem olan dengesiz veri seti ile sonuçlanmıştır. Dengesiz veri seti problemini çözebilmek ve azınlık sınıflarına ait veri sayılarını arttırabilmek için doğrudan kopyalama, SMOTE, ADASYN adı verilen üç örnekleme yöntemi ve titreşim ve zaman çarpıtma adı verilen iki çoğaltma yöntemi kullanılmıştır. Ayrıca, derin öğrenme modellerine girdi olacak finansal veri özelliklerini elde etmek için bu çalışmaya teknik göstergeler dahil edilmiştir. Elde edilen finansal veri özellikleri üzerinde özellik seçme ve özellik çıkarma işlemleri gerçekleştirilmiştir. Bütün göstergeler, karşılıklı bilgi metriği kullanılarak seçilen en iyi 15 gösterge, momentum, hacim, oynaklık ve trend göstergeleri olmak üzere altı farklı özellik grubu oluşturulmuştur. Önerilen derin öğrenme mimarilerimiz, evrişimli sinir ağı ve iki farklı kodlayıcı mimarisine sahip ilgi mekanizmaları, özellik gruplarının ve örnekleme, çoğaltma yöntemlerinin farklı kombinasyonları ile eğitilmiştir. Sonuçlar umut vericidir ve SMOTE örnekleme tekniğinin uygulandığı, ilk 15 göstergeyi girdi olarak alan ilgi derin öğrenme mimarisi kullanılarak oluşturulan model ile 56.48 olan en iyi makro ortalama F1 skoru elde edilmiştir.

ACKNOWLEDGEMENTS

First of all, I'd like to thank my thesis advisor, Assist. Prof. Ahmet Onur Durahim, for allocating his time for this research and sharing his knowledge in order to guide me through this research. Then, I'd like to thank my jury members, Prof. Ceylan Onay and Prof. Ali Atılgan since they have accepted to attend my thesis jury and they have given the most valuable feedback on my thesis.

I'd like to thank my colleagues at work since they have shown understanding to me at the times I had to study. Moreover, I'd like to thank TUBITAK 2210 Domestic Graduate Scholarship Program for providing me financial support.

I am also grateful to my friends, Seden Tezel and Özcan Gündeş since their supports through both my Bachelor's and Master's degree. Özcan was the one always listens to me, shares his knowledge and feedback with me. My dearest friend Seden and I were together not only in this, but at all the milestones in our lives. I'm so glad we are graduating together as we were starting this journey together.

My deepest appreciation goes to my family; my mother, my father and my sister. I'd like to thank them for the way they have raised me. They always show me how to be stronger and more successful, but they are the ones who support me the most even when I am not able to succeed.

Lastly but most importantly, I'd like to share my gratitude to my number one supporter, Orkun Kocatürk. He has effects at every stage of my life and my success. He makes everything more beautiful and easier that I couldn't imagine my life and this achievement without him.

TABLE OF CONTENTS

CHAPTER 1: INTRODUCTION 1

CHAPTER 2: LITERATURE REVIEW 6

2.1 Deep learning models for financial trading 6

2.2 Technical indicators as feature extractors 10

2.3 Time series data augmentation 13

2.4 Imbalanced learning and oversampling methods 15

CHAPTER 3: TECHNICAL INDICATORS 17

3.1 Momentum indicators 20

3.2 Volume indicators 25

3.3 Volatility indicators 28

3.4 Trend indicators 33

3.5 Versatile indicator: Ichimoku Cloud 36

CHAPTER 4: RESEARCH METHODOLOGY 39

4.1 Collection of data 41

4.2 Labelling process 43

4.3 Calculating technical indicators as new features 46

4.4 Data preprocessing and preparation 47

4.5 Building deep learning models 54

CHAPTER 5: EXPERIMENTS AND RESULTS 63

5.1 Convolutional neural network 64

5.2 Attention mechanism 67

5.3 Feature selection 70

5.4 Feature ablation study 71

CHAPTER 6: CONCLUSIONS AND MANAGERIAL IMPLICATIONS 75

CHAPTER 7: FURTHER RESEARCH 80

APPENDIX A: FORMULAS OF MOMENTUM INDICATORS 82

APPENDIX B: FORMULAS OF TREND INDICATORS 85

APPENDIX C: DATA SAMPLE 87

REFERENCES 88

LIST OF TABLES

Table 1. Categorization of Momentum and Trend Technical Indicators 19

Table 2. Categorization of Volume and Volatility Technical Indicators 20

Table 3. Output Classes and Number of Samples per Class 45

Table 4. Technical Indicators Chosen by SelectKBest Function 50

Table 5. Confusion Matrix for the Hold Class 61

Table 6. Formulas of Performance Metrics.....61

Table 7. Sample Confusion Matrix of CNN Model Trained with Imbalanced Data 64

Table 8. The Results of CNN Models Trained with Oversampled Data 65

Table 9. The Results of CNN Models Trained with Augmented Data 66

Table 10. The Results of Attention 1 Models Trained with Oversampled Data.....68

Table 11. The Results of Attention 1 Models Trained with Augmented Data 68

Table 12. The Results of Attention 2 Models Trained with Oversampled Data.....69

Table 13. The Results of Attention 2 Models Trained with Augmented Data 70

Table 14. The Results of the Models Trained with Top 15 Features.....71

Table 15. The Results of the Models Trained with Momentum Indicators 72

Table 16. The Results of the Models Trained with Volume Indicators 73

Table 17. The Results of the Models Trained with Volatility Indicators.....73

Table 18. The Results of the Models Trained with Trend Indicators 74

LIST OF FIGURES

Figure 1. Steps of research methodology 40

Figure 2. Candlestick chart of Bitcoin data 42

Figure 3. Random 24-hour slice of Bitcoin data 43

Figure 4. Pseudocode of window sliding algorithm 45

Figure 5. 80-hour section of augmented close prices generated by time warping 52

Figure 6. CNN as base model 56

Figure 7. Attention mechanism with CNN encoder and LSTM decoder 58

Figure 8. Soft attention mechanism 58

Figure 9. Second CNN as an encoder of attention 59

Figure 10. Time series cross validation model with train and test 60

CHAPTER 1

INTRODUCTION

Since the stock market entered our lives, forecasting trading signals of any financial instrument has always been one of the most desired problem to be solved. Many approaches have been developed to address this problem. Fundamental analysts and technical analysts have been working on this problem for many years. As machine learning field was starting to become successful in all types of prediction tasks, people working in this field have also developed solutions to the forecasting problem.

Fundamental analysts work to figure out financial position and performance of the company in the stock market. The studies of fundamental analysts rely on financial ratios of the company. The ratios are used for comparing financial situation and performance of the company to its competitors and industry benchmarks. Financial ratios include the ratios like working capital ratio, return on equity and earnings per share. Moreover, there are different factors which should be taken into account by fundamental analysis in order to determine worth of the company. These factors are exemplified as policies of government and news releases. To conclude, all the factors investigated by fundamental analysis help investors to determine whether stock price of the company is overvalued or undervalued. On the other hand, technical analysis methods are based on price and volume data of the stocks. Technical analysts extract patterns from historical data by using technical indicators. Technical indicators include volume, momentum, volatility, trend based calculations and candlestick charts. These calculations and charts make investors to recognize trading signals.

Oberlechner (2001) shared the findings of a survey related to importance of technical and fundamental analysis for traders and financial journalists from Europe. The results show that financial journalists consider fundamental analysis more than traders, whereas technical analysis become more popular among traders in recent years. Moreover, Lui and Mole (1998) conducted a similar survey for foreign exchange dealers in Hong Kong. The findings show that 85% of participants rely on both technical and fundamental analysis, however they find technical analysis more useful than fundamental analysis.

Artificial intelligence is the way of simulating human intelligence by computer based systems. Machine learning is a subset of artificial intelligence and it allows the system to learn and solve the problems according to past experiences. Machine learning includes three broad types of algorithms which are supervised, unsupervised and reinforcement learning. Deep learning is a subset of machine learning which includes deep artificial neural networks. Artificial neural networks (ANN) are inspired by architecture of human brain and they intent to solve the problems the way human brain does. ANNs are based on units and nodes which are called neurons. Output of each neuron is computed by some function which uses input data, weights and biases. The main goal of the network is to minimize a given loss function and therefore maximize the accuracy of its predictions by adjusting weights and biases. If relevant and sufficient data is given as input, the percentage of correctly predicting or classifying the output by ANN increases. This means an increase in accuracy of the trained ANN model and a decrease in loss. There are various types of ANN architectures such as convolutional neural networks since different architectures are needed to solve specific kinds of problems successfully.

As ANNs are becoming state-of-art technology in predictive tasks, data and computer scientists start utilizing them for forecasting trading signals together with the tools of technical analysis and fundamental analysis. Vargas, Dos Anjos, Bichara, and Evsukoff (2018) used deep learning models for predicting daily directional movements of a stock price. They employed both tools of fundamental and technical analysis in this study by using financial news and technical indicators. The results of this study and many other studies state that the most important problem of ANNs is preparing the relevant and sufficient input data. Huge number of data is needed for improving the performance of ANNs. However, quality of data is also important since irrelevant data might deteriorate ANNs' performance. As a result, it is important to select the proper big data and its features. Naik and Mohan (2019) analyzed the effects of different technical indicators on predicting prices of stocks. They created combinations of technical indicators by feature selection method and used them as input for different machine learning models. While the study shows the importance of feature selection, different models of neural networks were also investigated by this study. So, both the architecture of neural network and the quality of data need to be considered in order to increase the performance of predictions made by ANNs.

The study conducted in this thesis aims to predict future movements of stock prices. The output to be predicted is the buy-sell-hold action for traders. The approach uses different deep learning models with 71 technical indicators which are generated as new features of the data. This study classifies time series financial data according to past behaviors of stock prices. The data used in this study is Bitcoin data and is labelled with sliding window algorithm. However, imbalanced output class distribution is obtained by running the labeling algorithm. To overcome imbalance class problem and to increase

the number of examples in minority classes, different oversampling and augmentation methods are analyzed and implemented. Furthermore, similar to the novel image conversion approach proposed by Sezer and Özbayoglu (2018), the features generated using technical indicators are converted to images. Since convolutional neural networks (CNN) have obtained great results in image classification problems, they are also used in this study. For improving the success obtained by CNN, visual attention mechanism is also added to scope of this study. Visual attention mechanism is previously used for image captioning (Xu et al., 2015), however, this study proposes a similar approach to use it for classifying time series financial data. Moreover, six feature groups which include different sets of technical indicators are generated. Four of these groups are created according to types of technical indicators and they are named as trend, momentum, volume and volatility indicators. In order to see the effects of different types of technical indicators on the performance of deep learning models, an ablation study is conducted on the generated features. In addition, 15 features which have strong relationship with target are selected as fifth feature group and the effect of feature selection is further investigated. The sixth and last feature group is a group which uses all technical indicators.

In this chapter, the goal of this research has been introduced. Problem definition, possible solutions of the problem and contributions made by this research have also been mentioned. Chapter 2 mentions the previous researches related to deep learning models, oversampling methods, augmentation methods and technical indicators which are subjects of this study. Chapter 3 explains the definitions and calculations of technical indicators. It also groups the technical indicators used in this research according to their characteristics. Chapter 4 explains the methodological steps followed in this research

and Chapter 5 details the experiments conducted by this research and gives the results obtained in these experiments. Finally, Chapter 6 gives the details of conclusion and possible implications of this study while Chapter 7 proposes further researches in order to improve the findings of this study.

CHAPTER 2

LITERATURE REVIEW

The purpose of this study is to investigate deep learning methods for financial trading by using technical indicators as input features. For this purpose, first the historical open, high, low, close prices and volume (OHCLV) of Bitcoin is collected and labelled. However, the labelling algorithm used in this research gives rises to imbalanced data set problem. In order to address this problem, different oversampling and augmentation methods are investigated. This chapter mentions previous works related to deep learning methods, technical indicators, and augmentation and oversampling methods.

2.1 Deep learning models for financial trading

Forecasting future changes in prices of financial securities has always been a popular topic of interest for many researchers. Machine learning algorithms try to learn mathematical models in order to solve prediction problems. Machine learning models are created using training data. Deep learning is a subset of machine learning algorithms and deep learning architectures are created based on ANNs. Deep ANNs extract high level features from raw data progressively through its multiple layers. This section explains deep learning models that are used for predicting financial decisions to be made at each time step. In Section 2.2, technical indicators are discussed as feature extractors.

Fischer and Krauss (2018) deploy long short-term memory (LSTM) networks for forecasting directional movements of S&P 500 constituent stocks. LSTM networks are a special type of recurrent neural networks (RNN). They are designed to learn long-term dependencies and they overcome the problems that RNNs previously encounter with

such as vanishing gradient problem. Fischer and Krauss (2018) compare the performances of LSTM and memory-free classification methods. The benchmarking models used in the study are a deep neural network (DNN), a random forest (RF) classifier and a logistic regression (LOG) classifier. The stocks used in the study are selected according to some characteristics which include high volatility and strong short-term reversal. The daily data of stocks are collected from 1992 to 2015. Performance of LSTM network is resulted obviously higher than DNN and LOG. However, RF is not outperformed during the global financial crisis. To conclude, LSTM is the method with the highest performance compared to others if prediction accuracy and daily returns after transaction costs are analyzed. To conclude, their study shows that LSTM is a successful method which can be used even with noisy financial time series data.

The study conducted by A. Chen, Leung, and Daouk (2003) also aims to predict the direction of price movement. However, this study uses Taiwan Stock Exchange Index as its data and probabilistic neural networks (PNN) as its model. The data set includes the daily data from January 1982 to August 1992. The data is split into two periods which are from January 1982 to August 1987 and September 1987 to August 1992, respectively. The first period is used for training and validation, the second one is used for testing. Simulation of trading is also performed on second period data. PNN is selected because of its three important characteristics. First of all, it has high speed for training. Secondly, it can identify outliers and erroneous data so that it reduces the need for data preprocessing. Finally, it provides Bayesian probability of the class affiliation and this information is used for determining dynamic asset allocation. PNN model is compared to buy-and-hold strategy, random walk model and generalized methods of moments (GMM) with Kalman filter and PNN obtained the highest returns.

Sezer and Özbayoglu (2020) conducted another study that uses deep learning models for financial trading. They came up with a new approach and instead of using time series financial data, they decided to use 2-D stock bar chart images. They obtained 2-D images by converting time series financial data to images which includes bar charts. They developed an algorithm to label these images by calculating the slopes of images. They named this approach as CNN-BI (Convolutional Neural Network with Bar Images) since they used deep CNN for training. The results of this approach were compared to results of buy-and-hold strategy. CNN was selected for its success on image classification. Hyperparameter optimization of the model parameters, i.e. dropout rate and kernel sizes, were also performed. The daily financial data of Dow 30 stocks were collected as the data set for the study. Moreover, the study used two different time periods for training and testing the model. The first time period was in between 1997-2007, and second period was taken to be in between 1997-2012. For test data, the first period was from 2007 to 2012 and second one is from 2012 to 2017. The first test data includes highly volatile 2008 financial crisis data. However, second test data is steady bull market and it makes this data less volatile. This observation might show the importance of cross validation in order to evaluate the models. The training and test results of these models were compared to buy and hold strategy. The CNN model outperformed the buy and hold strategy and the results were even better in trendless or bear markets. The overall accuracy of the model is between 44% to 52% and number of negative returns with CNN model is lower than number of buy and hold strategy.

Qin et al. (2017) proposed a novel dual-stage attention-based recurrent neural network (DA-RNN) for time series prediction. Encoder of DA-RNN was an input attention mechanism while decoder of it was a temporal attention mechanism. These two

attention models were well integrated with an LSTM based RNN. Long-term temporal dependencies of time series data were captured by temporal attention mechanism. Furthermore, relevant data was chosen by input attention mechanism. DA-RNN performed well even on noisy inputs. As a result, DA-RNN outperformed other RNN-based approaches and ARIMA which were used as benchmarking models. Other RNN-based approaches used in the study were NARX RNN, Encoder-Decoder, Attention RNN, Input-Attn-RNN. Qin et al. (2017) conducted experiments on stock prices of 81 major corporations traded under NASDAQ 100 and SML 2010 data set. SML 2010 is a public indoor temperature forecasting data set. NASDAQ 100 data is also publicly available and the stock prices were collected in minutes between 26-06-2016 and 22-12-2016. Each model was trained 10 times and their performance metrics were averaged. To sum up, three metrics were used for comparison which were root mean squared error (RMSE), mean absolute error (MAE) and mean absolute percentage error (MAPE). DA-RNN obtained the lowest error values among other models.

The study conducted by Güreşen, Kayakutlu, and Daim (2011) used four different models for forecasting future movements of NASDAQ Stock Exchange Index. The models were trained using a multi-layer perceptron (MLP), a dynamic artificial neural network (DAN2) and two hybrid neural networks. The hybrid neural network uses generalized autoregressive conditional heteroscedasticity (GARCH). Mean Square Error (MSE) and Mean Absolute Deviate (MAD) were used to compare the performances of these models. The daily exchange rates of NASDAQ index were collected between 07-10-2008 and 26-06-2009. The first 146 days of data set were used for training and cross validation. Besides, the last 36 days were allocated for testing. The results showed that GARCH-DAN2 had the worst performance. On the other hand, MLP

outperformed GARCH-MLP and DAN2 with a minor difference. MLP is a widely used feed-forward neural network whereas DAN2 is a relatively new architecture. According to this study, it behaved like a statistical method rather than an ANN. Improving DAN2 architecture was suggested by Güreşen et al. (2011) as future work.

2.2 Technical indicators as feature extractors

In this section, previous studies which extract new features from raw financial data by using technical indicators are discussed. Details about technical indicators are not given in this section, but Chapter 3 provides further details about definitions and implementations of technical indicators.

A novel approach that converts financial time series data into 15x15 images was proposed by Sezer and Özbayoglu (2018). The financial time series data was obtained by calculating some technical indicators. In order to calculate the technical indicators, daily OHCLV data of Exchange-Traded Fund (ETFs) and stocks belong to Dow 30 were used. The collected daily OHCLV data was between 01-01-2002 and 01-01-2017. The number of calculated technical indicators used in the study was 15. These indicators are Relative Strength Index, Williams %R, Weighted Moving Average, Exponential Moving Average, Simple Moving Average, Hull Moving Average, Triple Exponential Moving Average, Commodity Channel Index, Chande Momentum Oscillator, Moving Average Convergence Divergence, Percentage Price Oscillator, Rate of Change, Chaikin Money Flow, Directional Movement Index, and Parabolic Stop and Reverse. These 15 indicators were calculated for 15 different time periods which ranges between 6-day and 20-day. As a result, 15x15 images were used to train and test the underlying CNN model. Training data always consisted of 5-years while test data always included 1-year

data. There were five different train and test pairs of data sets that were used to train and evaluate the CNN models. These data sets were obtained by sliding the starting date one year forward from the beginning of data. The experiments have shown that results of ETFs were better than Dow 30. The overall accuracy obtained was 62% on the test data of ETFs while it was 58% on Dow 30 test data.

J. Chen and Tsai (2020) conducted a study by using candlestick charts. Candlesticks charts are patterns for showing open, high, low and close prices in a specific time period. There are approximately 103 candlestick patterns and traders use these patterns to determine future price movements. The study proposed two-step GAF-CNN method which used Gramian Angular Field as the first step and CNN as the second step. The research data included data from Geometric Brownian Motion (GBM) model. Moreover, they used EUR/USD price data in minutes from 01-01-2010 to 01-01-2018. Eight patterns out of 103 candlestick patterns were selected. These are Morning Star, Bullish Engulfing, Hammer, Shooting Star, Evening Star, Bearish Engulfing, Hanging Man, and Inverted Hammer. Since CNNs has shown superior performances working with images, time series data was converted into images by GAF. GAF is a time series encoding method which takes one-dimensional time series data as an input and outputs two-dimensional convolutional time-series matrix. It first represents one-dimensional time series data in a polar coordinate system. Then, it transforms the angles represented in polar coordinates into symmetry matrix. The CNN architecture used in the research is similar to leNet architecture, J. Chen and Tsai (2020). The experiments were conducted with max pooling and without max pooling layer of CNN. CNN without max pooling layer outperformed the one with max pooling layer. CNN architectures were also compared to LSTM. Moreover, feature selection was performed in order to increase

accuracy of the models. Close price, upper shadow, lower shadow and real-body (CULR) were used instead of OHCL prices. As a result, CNN model without max pooling layer which was trained with CULR data has shown the best performance.

Alonso Monsalve, Suárez-Cetrulo, Cervantes, and Quintana (2020) explored the success of neural networks on different kinds of cryptocurrencies by using various types of technical indicators. They compared four different neural network architectures namely, CNN, hybrid CNN-LSTM network, MLP and radial basis function neural network. The data of the most popular six cryptocurrencies (Bitcoin, Dash, Ether, Litecoin, Monero and Ripple) were collected in minute time frames. Crptocompare application programming interface was used for acquiring the cryptocurrency data between the dates 01-07-2018 and 30-06-2019. For generating new features, 18 trend-following technical indicators were used. The first eight technical indicators were Accumulation/Distribution Oscillator, Commodity Channel Index, Larry William's R, Momentum, Moving Average Convergence Divergence, Relative Strength Index, Stochastic D% and Stochastic K%. Remaining 10 technical indicators were calculated as Simple Moving Average and Weighted Moving Average with different time periods. They used python library ta-lib for calculating these indicators. The data was split into train, validation and test data in order to select the best configuration of each model. During the cross validation, parameter optimization was also done for each model separately. Results over 20 experiments shown that convolutional LSTM is the best performed model with all cryptocurrencies. Furthermore, convolutional LSTM model has shown its best performance with Monera data which resulted in approximately 80% accuracy. Results of CNN was similar to convolutional LSTM with cryptocurrencies Bitcoin, Ether and Litecoin.

2.3 Time series data augmentation

Data augmentation methods generate synthetic data, and then the synthetic data is converted into appropriate format for supervised learning. Guennec, Malinowski, and Tavenard (2016) used two time series data augmentation methods, window warping and window slicing. Data augmentation was used to increase the size of the data set since their proposed deep learning model, CNN, needs huge amount of data to be more efficient. Window slicing method extracts slices from time series data and then the classification was performed at the slice level. Window warping method is used for warping the slices by speeding up or down. Their research used a version of CNN model named as t-leNet which is time series version of leNet architecture. Win/Tie/Lose score was used as a performance metric and t-leNet with window slicing obtained better results compared to window warping. t-leNet with window slicing had lowest error rates among other classification models. The window warping method was also used by Rashid and Louis (2019). They used this method together with two different machine learning algorithms, ANNs and K-nearest neighbor (kNN). Training with augmented data increased the performance. Accuracy, precision, recall and F1 scores of 97.9%, 96.4%, 96.8% and 96.6% were obtained respectively by using kNN. When ANN was used for training, the results were lower, 76.1%, 66.1%, 70.5% and 67.5%, but it was still higher than results obtained by the models trained with raw data. The experiments conducted by Rashid and Louis (2019) include different number of folds and different volumes of augmented data. In the end, their study suggested using CNN and RNN with time series data in order to achieve even higher results. Moreover, these two studies used different warping ratios in order to see effects of warping ratio on the performance.

Um et al. (2017) used CNN architecture in their study with various time series data augmentation methods in order to classify their data. Wearable sensor data of Parkinson's Disease patients were used in the study. The augmentation methods were jittering, scaling, cropping, time and magnitude warping, rotating and permutation. These methods were applied on time series data one by one, and their effects on performance were investigated separately. The effects of using a combination of the methods were also experimented. Combination of rotation and permutation methods achieved the best results. Rotation and permutation are both label preserving methods. Their combination increased accuracy to 92%, which was 76.7% when their proposed CNN model was trained with non-augmented data. Their proposed CNN model had an input layer, seven sequential convolutional layers which were followed by one global averaging pooling layer and an output layer. However, the first convolutional layer was used as an input layer for cropping and warping methods. Cropping reduces length of time series data and variability. On the other hand, warping methods are named as sampling methods and they perturb data with local translations. Cropping and warping methods did not have positive effects on model training since these methods might drop informative samples and change labels. Jittering adds element-wise noise while scaling multiplies with window-wise noise. Similar to previous two works, this study performed scaling and jittering methods by using different standard deviation values and aimed to see their effects on augmentation methods. The study did not achieve better results with jittering method irrespective of the value of the standard deviation. The reason behind was the fact that scaling is a label preserving method, and jittering might have caused the changes in labels of data. Their study has shown the importance of label preserving augmentation methods on wearable sensor data.

2.4 Imbalanced learning and oversampling methods

Imbalanced data set problem emerges when output classes in a supervised learning data set has imbalanced distribution. There are two types of methods, undersampling and oversampling, used with the aim of balancing the data. Undersampling methods reduces the number of samples in majority class, whereas oversampling methods increase the number of samples in the minority class in order to balance the class distribution. The main problem with undersampling methods is the fact that they might remove an important member of the majority class so that valuable information is lost. So, in this study, oversampling methods are investigated and used rather than undersampling methods. While augmentation methods generate synthetic data as raw input, oversampling methods used to increase the number of samples in the already labelled data set.

Random oversampling (named as direct copying in this study), Synthetic Minority Over-sampling Technique (SMOTE), Borderline-SMOTE, Borderline-SVM-SMOTE, Adaptive Synthetic Sampling (ADASYN) and Majority Weighted Minority Oversampling (MWMOTE) are examples of the successful oversampling methods used by many researchers for increasing the samples of minority class. Durahim (2016) compared all of these techniques in his study. Random oversampling, or direct copying, simply duplicates the data of minority class until balance between the classes is established. SMOTE is proposed to solve the imbalanced data set problem and also addresses the overfitting problem that occurs more likely with random oversampling. It generates synthetic data from minority class by interpolating the data point which are close to each other by using kNN method. While it mitigates the overfitting problem, it might cause new problems like over generalization. Borderline-SMOTE, Borderline-

SVM-SMOTE, ADASYN and MWMOTE are proposed for solving these new problems. ADASYN generates synthetic data from minority class by considering data distribution. Finally, WMOTE is a similar approach to SMOTE but it uses cluster based approach instead of kNN method. In Durahim (2016), the oversampling methods were applied on two imbalanced data sets taken from UCI repository and these methods were evaluated by using three binary classification algorithms, Support Vector Machine (SVM) with linear kernel, RF and kNN with k = 5. In conclusion, using three classification algorithms together with ADASYN method outperformed other oversampling methods in terms of both time and accuracy.

In this thesis, Direct Copying, SMOTE and ADASYN oversampling methods are investigated and their effects on the performances of deep learning methods are analyzed.

CHAPTER 3

TECHNICAL INDICATORS

A technical indicator is a mathematical calculation applied to price (opening, low, high and closing), volume or even another technical indicator of any tradable financial instrument. Technical indicators consist series of data points which is acquired by applying the mathematical calculation on historical data of a given asset. Historical data of a financial asset can be any composition of OHCLV at any given time. While some indicator calculations need only the close price with corresponding time period, other indicator calculations might need volume and other price related data such as open, high, low price values.

Fundamental analysis and technical analysis are two types of methods used for predicting the upcoming behavior of the security prices. Technical analysis exclusively takes OHCLV as input whereas anything that affects the value of securities is a subject of fundamental analysis. Technical indicators are commonly used by technical analysts in order to make more accurate predictions about future price changes. With the help of technical indicators, a technical analyst can determine exit and entry point of trades. Each indicator is a powerful tool to analyze situation of a stock in the market. However, many technical analysts and developers of technical indicators strongly recommend to use more than one technical indicators together in order to analyze market situation more precisely.

According to Achelis (2001), indicators can be grouped into two categories, lagging and leading indicators. Most commonly used lagging indicators are moving averages and Moving Average Convergence Divergence. On the other hand, Relative

Strength Index and Williams %R are examples of leading indicators. Leading indicators show upcoming state changes of the prices and can tell you how overbought or oversold any tradable financial instrument is. Lagging indicators show state changes of prices up to now. In other words, leading indicators look ahead while lagging indicators look back. Leading indicators can be used by technical analysts to forecast upcoming market direction, however, lagging indicators are used to recognize market direction up to now. Lagging indicators can be named as trend-following indicators. Furthermore, Yazdi and Lashkari (2013) argued that indicators can be separated into four types. These four types are momentum indicators, volatility indicators, volume indicators and trend indicators. The momentum indicators compare two different close prices at different times in order to determine speed and strength of price changes. The volatility indicators determine volatility of price. If the price of a security show higher fluctuations, then this indicates that it has higher volatility and vice versa. Higher fluctuations can be considered as dramatic changes over a short time period. Volume indicators use volume of traded shares to identify strength and direction of trend (Colby, 2003). The trend indicators are useful when price shows a trend. Trends can be identified when price changes are always in a certain direction. Possible trend directions are up, down or sideways. Furthermore, trends can be identified in various time periods.

This study categorizes technical indicators in four types as momentum, volatility, trend and volume indicators. In addition, each indicator that is a member of one of these four types also belongs to either leading or lagging categories except one. Ichimoku Kinko Hyo that is an indicator that belongs to more than one type and category. Ichimoku Kinko Hyo is all-in-one indicator. According to Patel (2010), the Ichimoku Kinko Hyo system consists of five components. The components are five lines which are

called Tenkan Sen, Kijun Sen, Senkou Span A, Senkou Span B and Chikou Span, respectively. Each component of Ichimoku Kinko Hyo indicator can belong to different types and categories. Moreover, this research uses many statistical methods as indicators and some of them do not belong to neither lagging nor leading categories. Table 1 and Table 2 show the types and categories of all technical indicators which are used in this study. Following sections explain the characteristics of four technical indicator types and technical indicators that belong to each type.

Table 1. Categorization of Momentum and Trend Technical Indicators

	Momentum	Trend
Lagging	- Percentage Price Oscillator - Aroon Up and Down Indicators - Aroon Oscillator - Senkou Span A - Senkou Span B - Directional Movement Index - Minus Directional Indicator - Minus Directional Movement - Plus Directional Indicator - Plus Directional Movement - Average Directional Movement Index - Average Directional Movement Index Rating - Moving Average Convergence Divergence, Signal Line and Histogram	- Parabolic Stop and Reverse - Linear Regression Elements - Eight Types of Moving Average - Time Series Forecast - Typical Price - Average Price - Median Price - Weighted Close Price - Midpoint Price over Period - Midpoint over Period
Leading	- Balance of Power - Commodity Channel Index - Chande Momentum Oscillator - Momentum - Rate of Change - Rate of Change Percentage - Rate of Change Ratios - Relative Strength Index - Fast and Slow Stochastic Oscillators - Stochastic Relative Strength Index - Triple Exponential Average - Ultimate Oscillator - Williams %R	

Table 2. Categorization of Volume and Volatility Technical Indicators

	Volume	Volatility
Lagging	- Chaikin Accumulation Distribution Oscillator - Chaikin Money Flow	- Kaufman's Adaptive Moving Average - Normalized Average True Range - Average True Range - True Range - Tenkan Sen - Kijun Sen
Leading	- Chaikin Accumulation Distribution Line - On Balance Volume - Money Flow Index	- Upper and Lower Bollinger Bands
Statistical		- Beta - Standard Deviation - Variance - Pearson's Correlation Coefficient

3.1 Momentum indicators

Momentum indicators calculate change of price and the calculation results in speed or velocity of price changes. J. J. Murphy (1999) defines market momentum as a practice of oscillator analysis and it should be used in collaboration with the market trend. Oscillator is a line which ranges between two extreme points. Oscillators warn the technical analysts about short-term overbought and oversold conditions. If an oscillator reaches the upper and lower ends of its boundaries, it is said to be respectively overbought and oversold.

This section gives further details and formulas of five momentum indicators. Moreover, formulas of other momentum indicators used in this research can be found in Appendix A.

3.1.1 Relative Strength Index

Relative Strength Index (RSI) was developed and introduced by Wilder (1978). RSI is an oscillator which takes values in between 0 and 100. It calculates velocity of price movement. Direction of price movement determines whether security is overbought or oversold at some point. The determination boundary points are 30 and 70. If RSI value is over 70, the security is considered to be overbought. In the meanwhile, the security is considered to be oversold if RSI value is under 30. The upcoming reversal of market can be identified by RSI when it gives a signal to technical analysts. It is the common characteristics of leading indicators.

According to Wilder (1978), RSI is calculated with Equation 3.1. RS in the formula stands for Relative Strength. Equation 3.2 and 3.3 are two ways to calculate the relative strength. Equation 3.3 is smoothing technique used by Wilder in order to make continuous RSI calculations simpler (Etzkorn, 1997).

It is remarkable that Wilder (1978) introduces the RSI formula by using 14-day time period. However, different time period can also be used while calculating the indicator. According to Achelis (2001), 9 and 25-day time periods also gain importance. On the other hand, Colby (2003) states that 5-day time period produces better results.

$$RSI = 100 - \frac{100}{1 + RS} \tag{3.1}$$

$$RS = \frac{Average\ of\ 14\ periods'\ up\ closes}{Average\ of\ 14\ periods'\ down\ closes} \tag{3.2}$$

$$RS = \frac{Previous\ average\ up\ closes\ *\ 13\ +\ Current\ up\ close}{Previous\ average\ down\ closes\ *\ 13\ +\ Current\ down\ close} \tag{3.3}$$

3.1.2 Fast and Slow Stochastic Oscillators

Stochastic Oscillator is a trading signal which was identified and popularized by George C. Lane (Colby, 2003). It is a momentum indicator which ranges between 0 and 100 values. There are various ways to interpret Stochastic Oscillator. Fundamentally, if its value rises above 80, the security is thought as overbought. On the other hand, the security is thought as oversold if its value falls below 20.

Stochastic Oscillator compares security's close price to trading range over predefined time period. Time period can be any number, however Lane suggests that it is better to use 5 to 21 days (Colby, 2003). The calculation of Stochastic Oscillator can be done with Equation 3.4.

Two signal lines are extracted from Stochastic Oscillator. The first line is main line which is known as %K. The second line is 3-period moving average of %K and named as %D. When these two lines intersects, the oscillator gives a signal which points out the upcoming change in market direction. When considered from this point of view, Stochastic Oscillator is a leading indicator.

$$K = 100 * \frac{(C - L)}{(H - L)} \tag{3.4}$$

$$C = the\ latest\ close\ price\ of\ security$$

$$L = the\ lowest\ traded\ price\ of\ security\ over\ n - period$$

$$H = the\ highest\ traded\ price\ of\ security\ over\ n - period$$

The result of Equation 3.4 is considered as Fast Stochastic Oscillator's %K. The difference between Fast and Slow Stochastic Oscillator is sensitivity. The Fast Stochastic Oscillator is too responsive to changes in prices and it causes false crossovers

(C. Murphy, 2019). In order to solve the problem, 3-period moving average is applied on Fast Stochastic Oscillator's %K and it is named as Slow Stochastic Oscillator's %K.

$$Fast\ \%D\ =\ 3 - period\ moving\ average\ of\ Fast\ \%K$$

$$Slow\ \%K\ =\ 3 - period\ moving\ average\ of\ Fast\ \%K$$

$$Slow\ \%D\ =\ 3 - period\ moving\ average\ of\ Slow\ \%K$$

At this point, it is important to declare that Fast Stochastic Oscillator's %D and Slow Stochastic Oscillator's %K are the same. As a result, this research does not use Slow Stochastic Oscillator's %K, it only uses Fast Stochastic Oscillator's %K and %D. Moreover, applying an additional 3-period moving average on Slow Stochastic Oscillator's %K results in Slow Stochastic Oscillator's %D. Slow Stochastic Oscillator's %D is also used in this research.

3.1.3 Williams %R

According to Lane (1984), a research group, which he was also a part of it, has developed some oscillators named as %A, %K, %D and %R. Larry Williams improved %R and it has become one of the most popular indicators as well. When viewed from this aspect, Williams %R indicator is very similar to Stochastic Oscillator and it is also a leading indicator. It is an oscillator which ranges between -100 and 0 values. It also measures overbought and oversold conditions of securities. If readings are between -20 and 0, a security is overbought and a security is oversold, if readings are between -80 and -100. Williams %R is calculated with Equation 3.5 and typically with 14-day period. However, any number of days can be used as a period.

$$K = -100 * \frac{(Highest\ high\ in\ n\ period - Latest\ close\ price\ of\ security)}{(Highest\ high\ in\ n\ period - Lowest\ low\ in\ n\ period)} \quad (3.5)$$

3.1.4 Ultimate Oscillator

Ultimate Oscillator is a leading momentum indicator which includes weighted average of three oscillator with different time periods. Most used time periods by technical analysts are 7, 14 and 28-day and they are, likewise, used by Ultimate Oscillator (Williams, 1985). The indicator generates buy and sell signals when there are divergences. The calculation of ultimate oscillator can be done with Equation 3.6, 3.7 and 3.8. TR in Equation 3.7 stands for True Range. It is a volatility indicator and it can be calculated by using Equation 3.28.

$$UO = 100 * [\frac{(A_7 * 4) + (A_{14} * 2) + (A_{28})}{4 + 2 + 1}] \quad (3.6)$$

$$A_n = \frac{\Sigma_{p=1}^{n} BP}{\Sigma_{p=1}^{n} TR} \quad (3.7)$$

$$Buying\ pressure(BP) = Close - MIN(Low, PriorClose) \quad (3.8)$$

3.1.5 Moving Average Convergence Divergence

Moving Average Convergence Divergence (MACD) is a momentum indicator. It has become one of the most popular indicators after it was developed in 1970’s (Appel & Dobson, 2007). The MACD indicator includes two lines and a histogram. The first line is main line which has same name as the indicator itself, MACD line. The second line is signal line. Buy and sell signals are triggered when MACD line intersects with its signal line. The speed of crossovers signals the overbought or oversold conditions of securities. Since the signal comes after market direction changes, MACD is a lagging indicator.

As stated by Appel and Dobson (2007), the MACD line is the difference between two exponential moving averages with 12 and 26-day time periods. The signal line is acquired by applying 9-day exponential moving average to MACD line. In other respects, any time period can be used in order to calculate MACD indicator. To illustrate, Falk and Moberg (2014) use 35-day as slow (long) period and 5-day as fast (short) period. Moreover, they calculate signal line by using 5-day time period.

The calculation of MACD line, signal line and histogram can be done with Equations 3.9, 3.10 and 3.11 respectively. EMA in Equation 3.9 and 3.10 refers to Exponential Moving Average which is a trend indicator. EMA can be calculated with Equation 3.35.

$$MACD\ Line\ =\ 12 - period\ EMA\ -\ 26 - period\ EMA \tag{3.9}$$

$$Signal\ line\ =\ 9 - period\ EMA\ of\ MACD\ line \tag{3.10}$$

$$Histogram\ =\ MACD\ line\ -\ Signal\ line \tag{3.11}$$

3.2 Volume indicators

Definition of market volume may change depending on marketplace of the financial instrument. However, it is generally the amount of traded security over specific time period. The study of Tsang and Chong (2009) claims that volume contains information which can not be exposed by price information. In the light of this information, it is inevitable to use volume for this research. Volume is participated in this study by using volume-based technical indicators. Exactly five volume indicators are used in this study and this section gives the details of these volume indicators.

3.2.1 On Balance Volume

On Balance Volume (OBV) was first introduced by Granville (1963). In his book, he claimed that price will go downward or upward if there is a sudden increase in volume but there isn't any considerable change in the price. The changes of OBV exists before the changes in prices. As a result, it can be said that it is a leading indicator. Equation 3.12 shows the calculation of OBV. As it can be seen in its formula, it is a cumulative indicator and it measures both negative and positive flows of the volume.

$$OBV = Previous\ OBV + \begin{cases} volume, & if\ close \geq previous\ close \\ 0, & if\ close = previous\ close \\ -volume, & if\ close \geq previous\ close \end{cases} \tag{3.12}$$

3.2.2 Chaikin Accumulation Distribution Line

Marc Chaikin was an inventor of Chaikin Accumulation Distribution (A/D) Line who began trading in 1966 and developing indicators in 1980 (Thomsett, 2010). Purpose of this indicator is identifying divergence between price and volume. If the divergence is detected, it can be the opportunity to identify bearish and bullish signals before giving a trading decision. Bullish signal occurs when A/D Line is upward and it means that price might follow the uptrend. On the other hand, bearish signal can be detected if A/D Line is downward. It might be a signal of downtrend in price. Similar to OBV, the signals occur before price changes are detected. This means A/D Line is also a leading indicator.

Equation 3.13 and 3.14 shows the calculation of A/D Line. This indicator is similar to OBV since they are both cumulative volume indicators. The difference between them is that A/D Line multiplies volume with current money flow.

$$A/D = Previous\ A/D\ +\ Close\ Location\ Volume\ *Volume \tag{3.13}$$

$$Close\ Location\ Volume\ =\ \frac{(Close\ -\ Low)\ -(High\ -\ Close)}{(High\ -\ Low)} \tag{3.14}$$

3.2.3 Chaikin Money Flow

Chaikin Money Flow (CMF) indicator was also found by Marc Chaikin. According to Thomsett (2010), CMF is an indicator based on analysis and results of A/D Line. If it is correctly investigated, the indicator can show traders whether trends in the market will continue or not. In this regard, this indicator is not a leading indicator like OBV and A/D Line. Since CMF follows the trends, it is a lagging indicator. Equation 3.15 and 3.16 shows the calculation of this indicator. CMF typically uses 21-day as time period, however any time period can also be used in calculating its value.

$$CMF\ =\ \frac{21\ period\ sum\ of\ money\ flow\ volume}{21\ period\ sum\ of\ volume} \tag{3.15}$$

$$Money\ Flow\ Volume\ =\ Close\ Location\ Volume\ *\ Volume \tag{3.16}$$

3.2.4 Chaikin Accumulation Distribution Oscillator

Chaikin Accumulation Distribution (A/D) Oscillator is a moving average oscillator that uses A/D Line, where both were developed by Marc Chaikin (Achelis, 2001). A/D Oscillator calculates MACD line of A/D Line with 3-day short period and 10-day long period. Equation 3.17 shows the formula of A/D Oscillator. The interpretation of oscillator depends on new high and new low values of stock price. Further, it is important to investigate situation of stock price compared to its high and midpoint values in a time period, more specifically in a day. When the similarity between MACD

and A/D Oscillator is considered, it can be said that A/D Oscillator is also a trend-following indicator.

$$A/D\ Oscillator = 3\ period\ EMA\ of\ A/D\ Line - 10\ period\ EMA\ of\ A/D\ Line \quad (3.17)$$

3.2.5 Money Flow Index

Money Flow Index (MFI) is an oscillator which takes values in between 0 and 100. It identifies overbought and oversold areas. Generally, 20 and 80 values are used as thresholds. If MFI value is above 80, the market can be considered as overbought and it is oversold when MFI reading is below 20.

Equation 3.18, 3.19 and 3.20 are used together to calculate MFI. TP in Equation 3.20 stands for Typical Price and it is a trend indicator. Its calculation can be found in Appendix B.

$$MFI = \frac{100}{1 + Money\ Flow\ Ratio} \quad (3.18)$$

$$Money\ Flow\ Ratio = \frac{14\ period\ positive\ money\ flow}{14\ period\ negative\ money\ flow} \quad (3.19)$$

$$Money\ Flow = TP * Volume \quad (3.20)$$

3.3 Volatility indicators

Volatility measures fluctuation of stock price in any time period. Tracking volatility of the financial market is important for almost all traders since volatility reflects the risk of the market. Various technical indicators and statistical functions are used to measure volatility of stock prices. The study conducted by Ebens (1999) shows that volatility is assessed by calculating either standard deviation or variance. Moreover, correlation and

beta are also used by technical analysts for assessing the market risk. While beta compares the volatility of stock prices to overall market, standard deviation measures the amount of dispersion (J. Chen, 2020). On the other hand, variance measures variability by taking average or mean of spread.

This chapter gives details about volatility indicators which are used in this research including variance, standard deviation, correlation and beta.

3.3.1 Variance

Variance measures the spread of data points from their mean. It compares data points to themselves and their mean. The data points in financial market are generally prices of stocks. It is calculated by using Equation 3.21. Its formula shows that distances of data points are squared before their averages are taken. By squaring, the distances are weighted. However, the result becomes a measure which has a different unit than mean when the distance is squared. This problem yields to further investigation which is done with using standard deviation.

$$Variance = \frac{\sum_{p=1}^{n}(x_i - mean)^2}{n} \tag{3.21}$$

3.3.2 Standard Deviation

Standard deviation is calculated by taking square root of variance. Similar to variance, it gauges historical volatility of stock by calculating its dispersion to the mean. As it is mentioned in Section 3.3.1, standard deviation is the square root form of variance. Taking square root of variance makes the result having same unit with mean as distances.

$$Standard\ deviation = \sqrt{Variance} \tag{3.22}$$

3.3.3 Beta

Beta is used for comparing volatility and market risk of stock to other stocks of the market. According to Equation 3.23, Beta is calculated by dividing covariance by variance. Covariance measures return of stock relative to stock market and variance measures return of stock relative to its mean.

$$Beta = \frac{Covariance}{Variance} \tag{3.23}$$

3.3.4 Pearson's Correlation Coefficient

Pearson's Correlation Coefficient measures the strength between two variables. The correlation coefficient takes values in between -1 and 1. Negative correlation is indicated with negative values and positive values show positive correlation. Pearson's Correlation Coefficient is used for technical analysis, but it does not have predictive power. Instead, it helps traders to understand correlation between financial instruments. Correlation is associated with market volatility and measures the risk of securities. The Pearson's Correlation Coefficient used in this study calculates correlation between high and low values of stock prices, respectively. Equation 3.24 shows the formula of Pearson's Correlation Coefficient where H and L in the formula stand for high and low values of stock prices.

$$Correlation\ Coefficient = \frac{Covariance(H, L)}{Standard\ Deviation(H)\ *\ Standard\ Deviation(L)} \tag{3.24}$$

3.3.5 Bollinger Bands

Bollinger (1992) developed Bollinger Bands which has volatility as key variable. The indicator plots three bands. The first band, named middle band, is same as Simple Moving Average indicator. Since SMA is a trend indicator, middle band is not considered as a volatility indicator. Other bands, upper and lower bands, are plotted above and below the middle band. The upper and lower bands are created by adding positive and negative standard deviations to middle band, respectively. Standard deviation is selected by Bollinger (1992) in order to gauge volatility because of its sensitivity. The bands are quick enough for showing reactions to the changes in the stock market. If price moves closer to the upper band, the market can be considered as overbought market, and it is oversold market on the other hand if the price moves closer to the lower band.

$$Middle\ Bollinger\ Band = n\ period\ SMA \tag{3.25}$$

$$Lower\ Bollinger\ Band = n\ period\ SMA\ -\ m * Standard\ Deviation \tag{3.26}$$

$$Upper\ Bollinger\ Band = n\ period\ SMA\ +\ m * Standard\ Deviation \tag{3.27}$$

Equations 3.25, 3.26 and 3.27 show the formulas of each band. n and m in the formulas are typically set to 20 and 2, respectively. However, the formulas can use another time period value as a parameter since there is no strict rule for calculating indicators with different time periods. Moreover, SMA in each formula refers to Simple Moving Average and Equation 3.33 shows its formula.

3.3.6 True Range

True Range (TR) is a volatility indicator which is used by many technical indicators. To illustrate, Average True Range and Ultimate Oscillator use this indicator in their formulas. Equation 3.28 shows formula of TR indicator.

$$True\ Range = MAX(High, Previous\ Close) - MIN(Low, Previous\ Close) \qquad (3.28)$$

3.3.7 Average True Range

Wilder (1978) introduced Average True Range indicator, and according Achelis (2001), interpretation of this indicator can be made by using same techniques that are applied to other volatility indicators. The indicator uses another indicator in its formula. It is calculated by taking simple moving average of TR indicator. Equation 3.29 shows formula of the indicator.

$$Average\ True\ Range = \frac{Previous\ TR\ *\ (n\ period\ -\ 1) + TR}{n\ period} \qquad (3.29)$$

3.3.8 Normalized Average True Range

Normalized Average True Range shows ATR as a percentage. It can be said that ATR and NATR are used for different purposes. According to Forman (2004), ATR is normalized in order to simplify indicator for traders who want to compare different securities or different periods of time. Equation 3.30 shows formula of the indicator.

$$Normalized\ Average\ True\ Range\ =\ 100\ *\ \frac{n\ period\ ATR}{Close} \qquad (3.30)$$

3.3.9 Kaufman's Adaptive Moving Average

Kaufman's Adaptive Moving Average (KAMA) is a moving average indicator which was designed by Kaufman (1995). The indicator is responsive to trend volatility. It is expected to stop moving if the market situation is uncertain. KAMA is lagging indicator. It follows trends since it is a moving average. Its formula is similar to EMA which is given in Equation 3.35. However, weights are given as efficiency ratio (ER) which is adjusted for volatility. Equation 3.31 and 3.32 give formulas of the KAMA indicator and the ER.

$$KAMA = Current\ Price * ER + Previous\ EMA * (1 - ER) \tag{3.31}$$

$$ER = \frac{n\ period\ price\ change}{sum\ of\ n\ period\ price\ difference} \tag{3.32}$$

3.4 Trend indicators

Trend indicators identify strength and direction of a trend in the market. The difference between trend indicators and other types of indicators is the fact that the main and only goal of trend indicators is measuring the characteristics of trend. On the other hand, other types of indicators also measure momentum, volatility and volume of securities. Trend indicators only include trend-following indicators. There is no leading indicator that belongs to this type of indicators. Peachavanish (2016) conducted a study related to trend indicators and in their study, trend-following strategy is defined as computing the direction and strength of trend from historical data and making profits by trading in the same direction of trend. Moreover, the strategy was named as "buy high and sell higher" since it is assumed that the trend continues on same direction.

This section explains five of the trend indicators used in this study and the rest can be found in Appendix B.

3.4.1 Moving averages

Moving averages are simple technical analysis tool. Nevertheless, they are the most commonly used indicators among other trend indicators. Moving averages basically averages the stock prices. However, there are many forms and different applications of them. There are eight different types of moving averages used in this study and four of them are explained in this section. Formulas of the remaining moving average indicators can be found in Appendix B.

3.4.1.1 Simple Moving Average

Simple Moving Average simply computes the average of the stock prices over a specific time period. It usually uses close prices of stock. It identifies whether the price of the security is uptrend or downtrend in the market. Equation 3.33 gives its formula.

$$SMA = \frac{\sum_{p=1}^{n} Price}{n} \tag{3.33}$$

3.4.1.2 Weighted Moving Average

Weighted moving average calculates average of prices over specific time period by weighting the prices. Most traders give higher weights to the most recent prices. The reason for giving weights is making the indicator more responsive while keeping other characteristics of moving averages same. Traders must be aware of the sensitivity of this indicator to the weights while giving trading decisions.

$$WMA = \frac{Price_1 * n + Price_2 * (n-1) + \cdots + Price_n}{\big(n * (n+1)\big)/2} \tag{3.34}$$

3.4.1.3 Exponential Moving Average

Exponential Moving Average (EMA) is similar to Weighted Moving Average. However, EMA gives higher weights to most recent prices and it makes recent prices to contribute more to the indicator. Equation 3.35 shows formula of the indicator. Smoothing in the formula is generally set to 2.

$$EMA = Current\ Price * \frac{Smoothing}{1+n\ period} + Previous\ EMA * \left(1 - \frac{Smoothing}{1+n\ period}\right) \tag{3.35}$$

3.4.1.4 Mesa Adaptive Moving Average

Mesa Adaptive Moving Average (MAMA) is an adaptive way of using EMA. It is similar to KAMA since they are both adaptive moving averages. The weights given to EMA was adapted by varying them. The MAMA has two lines, first MAMA line itself and Following Adaptive Moving Average (FAMA). FAMA is acquired by applying MAMA to first MAMA line (Ehlers, 2001). Even the indicator is an adaptive version of moving averages, similar approach used by traditional moving averages can be followed for trading with MAMA and FAMA.

3.4.2 Parabolic Stop and Reverse

Parabolic Stop and Reverse indicator was also introduced by Wilder (1978), like RSI and ATR indicators. It identifies trend direction and possible reversals of security price. Possible reversals give information to traders related to entry and exit points. The indicator is used to set trailing points by using stop and reverse method. Equation 3.36

shows formula of the indicator. Extreme point in the formula is the lowest low in downtrend or the highest high in uptrend. Moreover, acceleration factor (AF) is usually given as 0.02 and increases by adding 0.02 each time extreme point makes new high or new low. The maximum value of AF is usually 0.2.

$$SAR = XP - Previous\ SAR * AF + Previous\ SAR \tag{3.36}$$

$$XP = Extreme\ Point$$

$$AF = Acceleration\ Factor$$

3.4.3 Time Series Forecast and Linear Regression

Time Series Forecast (TSF) shows the statistical trend of security over a specific time period and it is based on Linear Regression analysis. TSF indicator contains same information as Linear Regression Trendline (LRT). Any point along the TSF is equal to last point of LRT. In this respect, TSF indicator is also called as regression oscillator or moving linear regression indicator similar to moving averages. However, TSF has advantages over moving averages (Achelis, 2001).

Linear regression relates two variables and has an equation of the form Y = mX + n, where m is slope and n is intercept. X in the formula is the independent variable, whereas Y is the dependent variable. This study uses Linear Regression, Linear Regression Angle, Linear Regression Intercept and Linear Regression Slope in addition to TSF.

3.5 Versatile indicator: Ichimoku Cloud

Ichimoku Cloud is an all-in-one indicator which was developed by Goichi Hosoda in 1948 (Patel, 2010). It both gauges trend and momentum direction of stock prices. It

includes five different technical indicators. Each of these indicators uses averaging in their calculations, and some of them make use of other Ichimoku Cloud components. These indicators are implemented and used in this study because of their popularity. This study uses four lines of Ichimoku Cloud by excluding Chikou Span since it causes loss of too many data points. They are explained in detail in this section.

3.5.1 Tenkan Sen

This subsection mentions Tenkan Sen first by giving its formula in Equation 3.37. If its formula is considered, it can be said that it uses average highest high and lowest low instead of simple moving average. According to (Patel, 2010), Tenkan Sen measures intraday volatility by using these calculations instead of averaging close prices. Moreover, it reflects short term price movements better as compared to the one that uses simple moving average. Since the indicator gauges volatility and follows the trend, it is a lagging volatility indicator.

$$Tenkan\ Sen = \frac{Highest\ High\ of\ 9\ period + Lowest\ Low\ of\ 9\ period}{2} \quad (3.37)$$

3.5.2 Kijun Sen

While Tenkan Sen determines the short term price movement, Kijun Sen determines the medium term movement of prices (Patel, 2010). The formula of Kijun Sen is similar to Tenkan Sen except for the default time period used. This similarity makes Kijun Sen also a trend-following volatility indicator. Equation 3.38 shows the formula of Kijun Sen.

$$Kijun\ Sen = \frac{Highest\ High\ of\ 26\ period + Lowest\ Low\ of\ 26\ period}{2} \tag{3.38}$$

3.5.3 Senkou Span A and Senkou Span B

Senkou Span A and Senkou Span B together form a kumo cloud and crossovers between these two indicators signal the change in trend direction. Since the signal is followed by a change in direction, these two indicators might be considered as leading indicators. In fact, they are also named as Leading Span A and B. However, the indicators are slow to react the change since their formulas uses 26 and 52-day periods. As a result, they are considered as lagging indicators. The indicators measure support and resistance levels and momentum as well. Equation 3.39 shows the formula of Senkou Span A. As the formula indicates, it uses other two components of Ichimoku Cloud. Equation 3.40 shows the formula of Senkou Span B.

$$Senkou\ Span\ A = \frac{Tenkan\ Sen + Kijun\ Sen}{2} \tag{3.39}$$

$$Senkou\ Span\ B = \frac{Highest\ High\ of\ 52\ period - Lowest\ Low\ of\ 52\ period}{2} \tag{3.40}$$

CHAPTER 4

RESEARCH METHODOLOGY

Previous chapters give background information and exemplify previous applications related to deep learning models, data augmentation and oversampling methods, and technical indicators. This section explains proposed methodology of this research which brings the given information together in order to solve a research problem.

Deep learning is a popular subset of machine learning. It is used to address many research problems specifically in speech recognition, computer vision and natural language processing. Deep learning models are built by using multilayered neural networks. A neural network consists of input, hidden and output layers. They take example subset of input data and try to predict output by passing the input data through hidden layers. The aim of deep learning model is to minimize difference between expected and predicted output. Weights are adjusted by neural network during training in order to make better predictions.

The accuracy of prediction is affected by neural network architecture. However, quality of input data set plays the most important role to increase performance of neural network. Data augmentation, oversampling and feature engineering are three main concepts to enhance data quality. Training DNNs requires huge amount of training data. If sufficient data is not available, various data augmentation methods can be used in order to increase data amount without collecting new data. Furthermore, feature engineering methods which extract new features from original ones that can be used to increase the performance of the models. According to Long, Lu, and Cui (2018), technical analysis and statistical methods are mostly used feature engineering methods.

Technical indicators are tools for technical analysis and they can be implemented for extracting new features from the original stock market data. The goal of this research is to improve performance and accuracy of deep learning models by using technical indicators, data augmentation and oversampling methods.

The purpose of this chapter is to introduce the steps followed in this research. The steps mentioned in the following are; data collection, labelling process, calculating technical indicators as new features, data preprocessing and preparation, building deep learning models and action prediction. Data Preprocessing and Preparation step consists of sub-steps which are feature scaling, feature selection, feature ablation study, dealing with imbalanced data, data cleaning and reshaping data for creating images before giving data as an input to deep learning models. Figure 1 shows the steps that belong to the methodology of this research

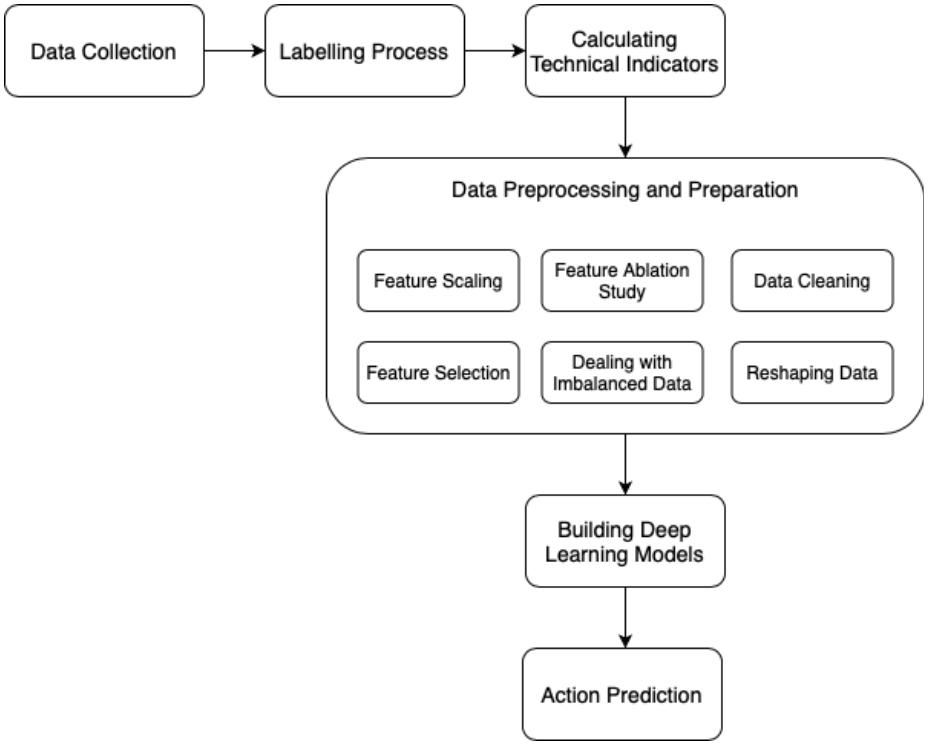

Figure 1. Steps of research methodology

4.1 Collection of data

In this study, predicting trading actions for the cryptocurrency market is targeted. This section explains the reasons behind selecting cryptocurrency market as the target market and gives further detail about collected cryptocurrency input data.

The cryptocurrency market differs from the traditional stock market with its unique features. Two characteristics of cryptocurrencies, decentralization and openness, make their market one of the best test candidates for deep learning experiments (Jiang & Liang, 2017). The decentralization makes cryptocurrencies not regulated by central authorities. According to Jiang and Liang (2017), the decentralization gives traders an opportunity that they can trade with low entrance prerequisites. On the other hand, the openness makes cryptocurrencies more accessible since there is no trading limit and the market is open for 24 hours a day and 7 days a week. In addition to these two characteristics, the cryptocurrency market can be considered as younger market compared to traditional stock market. Being immature in terms of scientific studies, it is inevitable that there is a need for further research on this field in the literature.

Bitcoin is the best known cryptocurrency in the market and it has the highest market capital. Market capitalization of Bitcoin is 163.4 out of 249.2 billion in USD, which is total market capitalization of all cryptocurrencies according to a report published in May 2020 ("Cryptocurrency market capitalizations", n.d.). Based on this information, it can be said that the most dominant cryptocurrency of the market is Bitcoin. Furthermore, it has the longest historical data in cryptocurrency market since it is the first known cryptocurrency. As a result, historical Bitcoin data is used in this research.

Historical data of cryptocurrencies are obtained by application programming interfaces (APIs). The APIs can provide historical data on daily, hourly and even minute time frames. Since historical data of cryptocurrencies can be acquired only for a decade, hourly data is used instead of daily data in order to increase data size. The data is acquired between 18-07-2019 07:00 and 01-01-2020 00:00. Data has 4002 records in this time interval. Each record has open, high, close, low prices and volume data. OHLCV is a common abbreviation used to refer to these five data points. The open and close prices of data show the first and last prices in an hour. The high and low prices represent the highest and lowest prices during an hour. The volume is total number of securities traded during an hour. While Figure 2 shows candlestick chart of data, Figure 3 is a random 24-hour slice of the data chart.

Figure 2. Candlestick chart of Bitcoin data

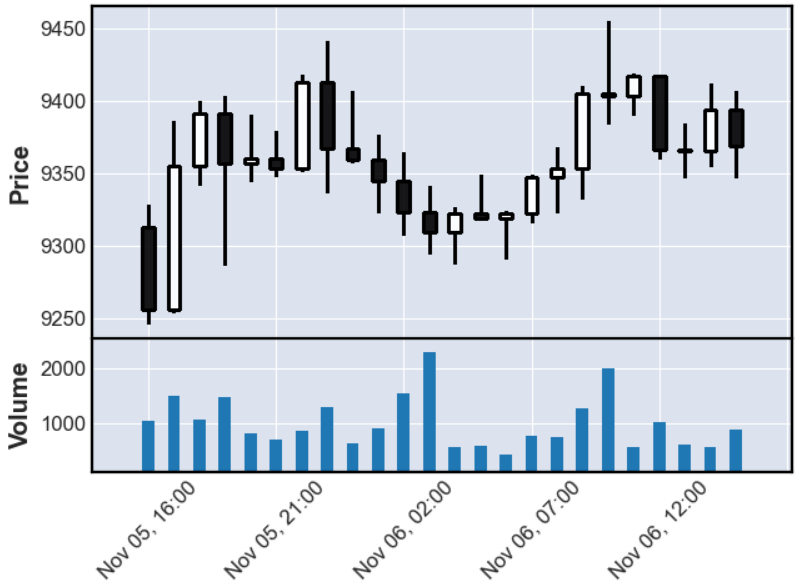

Figure 3. Random 24-hour slice of Bitcoin data

4.2 Labelling process

Time series forecasting problem is a supervised machine learning problem and one of the most important part of forecasting is defining output spaces. Sliding window approach is a common approach used to label time series data. The algorithm and parameters used by Sezer and Özbayoglu (2018) is selected for applying sliding window since it has been observed to achieve successful results in trading simulations. When the labels given by the algorithm are used for financial trading, the gain is considerably high and it is considered as a success criteria of labelling. A simple "buy/sell/hold all" financial trading algorithm similar to the one used by Sezer and Özbayoglu (2018) is employed for further evaluation of the labelling process. If the "buy/sell/hold all" financial trading algorithm is used with the labels provided in the test sets of the five

cross validation runs explained in Chapter 5, the average amount of money obtained as a result is approximately $14,600.

The algorithm starts with a predetermined amount of cash. It then basically buys stocks with all of the cash it has if the predicted label is a buy. Then, it waits until the model predicts a sell and then the trading algorithm sells all the stocks it has and convert them into cash. And this procedure is followed until the end of the test period. A transaction cost is also incurred when a buy or a sell transaction occurs. The initial amount of cash is determined as 10,000$. Transaction cost is taken as 0.001 of the traded amount (“Komisyonlar”, n.d.). All the stocks at hand or all the money in cash is traded if a transaction occurs. If the trading algorithm does not encounter with any consecutive predictions of buy and sell, then no transaction occurs within the financial trading simulation. The success of the financial trading algorithm depends on and highly sensitive to the order of buy and sell labels. This is the main drawback of the simple "buy/sell/hold all" algorithm.

Figure 4 shows how close price data is labelled by using sliding window with window size 11. Middle point of windows is labelled according to the labelling algorithm. To be more clear, middle point is labelled as "BUY" if it is the lowest price among prices within the window. On the other hand, it is labelled as "SELL" if it is the highest one. First and last five days can not be labelled since the window size is 11. Because of this, they are labelled as "NaN". It can be said that there are w-1 NaN labelled time periods if the window size is w. Table 3 shows the class distribution after the algorithm is applied. It can be clearly seen that there is imbalance between output classes and imbalanced problem is addressed by using data augmentation or oversampling methods explained in Section 4.4.4.

Table 3. Output Classes and Number of Samples per Class

Output Class	Number of Samples
Buy	197
Sell	261
Hold	3534

Algorithm 1 Sliding Window

```
procedure LABELLING
    windowSize ← 11                                          ▷ hours
    data ← close prices
    dataSize ← number of close prices                        ▷ 4002 records
    rowCounter ← 0
    while rowCounter < dataSize do
        if rowCounter > dataSize − windowSize/2 then
            label = NaN
        else if rowCounter < windowSize/2 then
            label = NaN
        else
            windowMiddleIndex ← rowCounter
            windowBeginIndex ← rowCounter − windowSize/2
            windowEndIndex ← rowCounter + windowSize/2
            window ← data[windowBeginIndex : windowEndIndex]
            maxValue ← maximum price value among prices of window
            minValue ← minimum price value among prices of window
            if data[windowMiddleIndex] = maxValue then
                label = SELL
            else if data[windowMiddleIndex] = minValue then
                label = BUY
            else
                label = HOLD
            end if
        end if
        rowCounter + +
    end while
end procedure
```

Figure 4. Pseudocode of window sliding algorithm

4.3 Calculating technical indicators as new features

The collected data set contains bitcoin's open, high, close, low prices, its volume and the date as features. As it is mentioned before, technical indicators are calculated to create new input features from these raw input features. Before starting to calculate technical indicators, they are categorized in four groups. Table 1 and 2 shows the technical indicators and the groups they belong to. Most of the technical indicators are calculated by using python library TA-Lib (https://www.ta-lib.org). However, those that are not implemented in this library are implemented in Python programming language.

Technical indicator calculation mostly requires close price data. However, there are functions which also need combinations of OHCLV. In addition, almost all functions take a specified time period as an input. When technical indicators are introduced to the financial world, the optimum time period needed for their calculation is commonly determined. Consequently, functions implemented in TA-Lib library have default time periods set to 9 day for short periods and 14 day for long periods.

Shynkevich, McGinnity, Coleman, Belatreche, and Li (2017) investigated the effects of calculating technical indicators with varying time periods on predictive models. They calculated technical indicators with 1, 3, 5, 7, 10, 15, 20, 25 and 30-day time periods and selected the one which resulted in best performance values among them. Moreover, Tsang and Chong (2009) calculated OBV indicator with 10, 20, 50 and 100-day time periods and compared the effects of each period. In contrast to Shynkevich et al. (2017) and Tsang and Chong (2009), the purpose of this study is not to investigate the time period that performs best. Instead, all technical indicators are calculated with different time periods and use them as features that are fed to the deep learning models. The specified time interval for this study is between 6 hours and 20 hours since this

range includes most of the successful time periods which are defined by developers of the indicators. To sum up, each technical indicator is calculated 15 times with 15 different time periods.

The outputs of each indicator between 6 and 20 hour time periods form a new feature structure and it can be called as a feature block. For example, CMF indicator is calculated with volume, high, low and close price data and for 15 different time periods. The results are kept together side by side and they form a CMF feature block. Appendix C shows the CMF feature block with labels and original features which are date and OHCLV. It is noticeable that there are NaN values in feature blocks since indicators can not be calculated for shorter time periods than given time period. To illustrate, first five values are NaN for CMF calculation with 6-hour time period and similarly, first 19 values are NaN for 20-hour calculation. The number of NaN values in the first lines can vary from indicator to indicator. Even some indicator blocks do not have NaN values. It depends on indicator's formula. Then, these new features are used to create images that is explained in detail in Section 4.4.6, and the newly created images also have same label as their corresponding close prices since the labelling process decides on the action to be taken just after seeing the current close price of the cryptocurrency.

4.4 Data preprocessing and preparation

Data preprocessing techniques in machine learning are the ways to transform input data set into a meaningful or useful form so that prediction process becomes more efficient and its results become more accurate. Huang, Li, and Xie (2015) conducted a study in which feature and case selection, missing data treatment and data transformation steps were performed as data preprocessing methods in order to increase accuracy of machine

learning techniques. This research uses similar data preprocessing techniques and this section explains them in detail.

4.4.1 Feature scaling

The goal of data transformation methods is converting data into categorical or encoded inputs to be fed into machine learning models. Feature scaling is a data transformation method and its goal is bringing the input features to the same degree of influence (Angelis & Stamelos, 2000). In this research, features that are extracted by using technical indicators vary in their ranges and normalization is needed in order to get them in a particular range. Min-max scaling is applied to each feature block since they are treated as an image later. Min-max scaling ranges the features into [0, 1] or [-1,1] if there are negative values.

$$x' = \frac{x - min(x)}{max(x) - min(x)} \tag{4.1}$$

Equation 4.1 is used for min-max scaling of the features. While x shows the feature block values, x' shows the normalized feature block values. min(x) and max(x) in the formula shows minimum and maximum values of each indicator block. Since features are not cleaned yet, minimum and maximum values are determined after eliminating NaN values.

4.4.2 Feature selection

It is a considerable point to extract relevant features when large number of features are used as input of deep learning models. Features should have strong relations with

targets. If there exist features which are not strong enough, it might make the model to train on noisy data. While data cleaning process is used for eliminating missing values, the irrelevant features can be eliminated by feature selection.

Having large number of features might increase accuracy of deep learning model. But then, it might encounter with overfitting and high computational cost problems. Data cleaning and feature selection methods can avoid overfitting. Section 4.4.5 elaborates on data cleaning. This section explains the feature selection, its methods and the methods used in this research.

It is important to have only the relevant features in input data set in order to obtain more accurate results with machine learning models. Feature selection is a method which reduces dimensionality of the input data set by selecting relevant group of features. The purpose is to increase performance of the model while decreasing computational cost of the model.

The main methods of feature selection can be categorized into three groups which are filter methods, wrapper methods and embedded methods. According to Jovic, Brkic, and Bogunovic (2015), the three methods are explained as follows: filter methods are only dependent on features and target, wrapper methods select feature groups by considering the performance of a model, and finally embedded methods select features as an internal part of the execution of the model.

The filter method is used as feature selection method in this research. The metrics for selecting features can be correlation or other statistical measures between features and the target. For example, 15 features with the highest correlation can be given as an example of feature group if correlation between features and target is selected as a metric.

In order to select features, all technical indicators are calculated with only 14-hour time period at first step since TA-Lib functions have 14-day time period by default. Then, SelectKBest function of python library scikit-learn is used to select top k best features among the technical indicators and mutual information is used as the score function's criterion (Pedregosa et al., 2011). Table 4 shows top 15 features which are chosen by SelectKBest function.

Table 4. Technical Indicators Chosen by SelectKBest Function

Top 15 Features
Balance of Power
Commodity Channel Index
Chande Momentum Oscillator
Rate of Change
Rate of Change Percentage
Rate of Change Ratio
Rate of Change Ratio 100 Scale
Relative Strength Index
Fast Stochastic Oscillator's %K
Fast Stochastic Oscillator's %D (same as Slow Stochastic Oscillator's %K)
Fast Stochastic Relative Strength Index's %K
Fast Stochastic Relative Strength Index's %D
Ultimate Oscillator
Williams %R
Money Flow Index

4.4.3 Feature ablation Study

Feature ablation study in machine learning refers to the removal of some group of input features in order to investigate how it affects the performance of the model. For this study, the ablation is applied on technical indicator groups. The new features which are generated by calculating technical indicators are grouped according to their types and categories as provided in Tables 1 and 2. For training the model, technical indicators other than one specific type are removed and the remaining features are used to train the classification model. To exemplify, deep learning models are trained with only trend

indicators or with only momentum indicators. So, trend, momentum, volume and volatility indicators are used as separate feature groups. Section 4.5 gives details of training models with these feature groups.

4.4.4 Dealing with imbalanced data

As it is mentioned earlier in Section 4.2, labelling process used in this study resulted in imbalanced class distribution. As it can be seen from Table 3, the number of hold labels is approximately 14 times the number of buy labels while it is almost 18 times the number of sell labels. In order to achieve a balance between the number of samples in each label, the data augmentation and oversampling methods are used. Data augmentation and oversampling methods are considered as two different approaches. Data augmentation methods create new time series data similar to the original one and use this newly created data to construct new features and corresponding labels. On the other hand, oversampling methods create new samples for each class from already existing ones.

Data augmentation methods are explained in Chapter 2. Some image augmentation methods like jittering, scaling, cropping, rotating, magnitude and time warping are used as time series augmentation methods by Um et al. (2017). In this research, jittering and time warping among these augmentation methods are used to overcome the class imbalance problem and increase the number of samples.

Jittering method implies adding element-wise noise and the noise is obtained from normal distributions of 0 mean with different values of standard deviations. On the other hand, time warping method implies distorting time in order to speed it up or down by using different warping ratios (Um et al., 2017). Figure 5 shows 80-hour section of

time warped close prices. The standard deviation of normal distribution is set to 0.01 in jittering, and same value is used as warping ratio in time warping. Moreover, values of 0.1 and 1.0 are also used as parameter values in order to see their corresponding effects on the performance of the generated classification models.

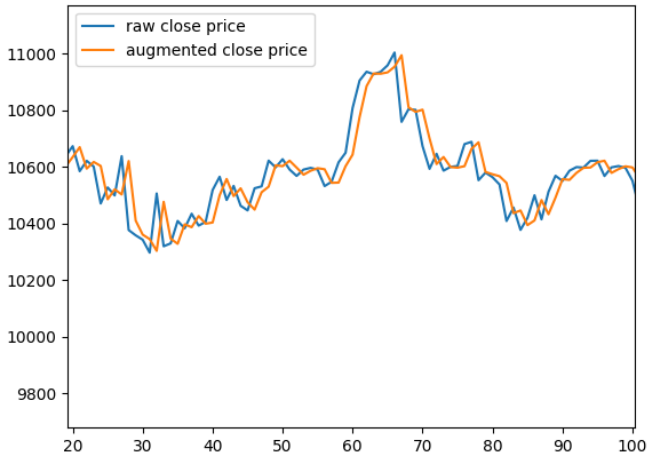

Figure 5. 80-hour section of augmented close prices generated by time warping

Besides augmentation methods, three oversampling methods are investigated in this study. SMOTE and ADASYN synthesizes new examples from minority output classes. Since Durahim (2016) obtained high accuracy and better performance results with SMOTE and ADAYSN oversampling methods, these two methods are also used in this study. Moreover, direct copying method is added to the scope of data oversampling.

SMOTE was first introduced by Chawla, Bowyer, Hall, and Kegelmeyer (2002). The technique chooses an instance from minority class and then, chooses one of its k-nearest neighbors which also belongs to same class. The aim is to connect these two instances in order to form line segment in feature space rather than in data space. Synthetic examples are generated along the line segment by joining different

combinations of other k-nearest neighbors of the same minority class (He & Ma, 2013). In order to perform the algorithm, SMOTE function of python library named imblearn is used in this research (Lemaître, Nogueira, & Aridas, 2017).

The second oversampling method, ADASYN was first introduced by He, Bai, Garcia, and Li (2008). This approach also generates synthetic samples but it uses a weighted distribution for different minority class depending on their learning levels. To be more clear, synthetic examples are generated more if minority class samples are harder to learn (He et al., 2008). In order to perform this technique, ADASYN function of imblearn python library is used in this research (Lemaître et al., 2017).

Direct copying method directly copies buy and sell data as its name suggests. Each of the original sell and buy labelled data is duplicated until the number of sell, buy and hold labelled data become equal to each other. It is important to mention once again that already generated features are oversampled by SMOTE, ADASYN and direct copying whereas raw OHLC price and volume data is augmented by using jittering and time warping methods. One more difference between these methods is that ADASYN generates less numbers of synthetic examples from minority classes while the other methods generate instances by making the number of instances equal in each class.

In this study, these oversampling and augmentation methods are used to only to balance the training data set and keep the number of instances in the test data set same. So, evaluation of the models has been performed with imbalanced test data set.

4.4.5 Data cleaning

As it is mentioned in Section 4.4.2, missing values should be cleaned before the data is fed to the deep learning algorithm. Missing values can be handled either by imputing

their values or dropping them. According to Jerez et al. (2010), imputation methods are grouped into statistical techniques like mean, hot-deck, multiple imputation and machine learning techniques like MLP, self-organisation maps (SOM), kNN. In contrast to Jerez et al. (2010), this study chooses to drop missing values. As it can be seen in Appendix C, there are NaN values among both labels and generated features. Whole feature set and corresponding label for the time periods having NaN values are dropped before applying oversampling methods. However, these NaN values are dropped after applying methods, jittering and time warping since these two methods augment raw price and volume data which do not have any NaN values.

4.4.6 Reshaping data for creating images

Reshaping data is the last step of this section. The data is scaled and cleaned so that it is ready to be used in creating images. In order to create images, width is taken as the number of technical indicators that are selected and height is taken as time period interval. Images can also have depths which correspond to channels which may represent the number of historical feature sets to be used. However, the created images have only one channel so that depth is taken as 1. To sum up, the final data become images with shape (Number of Technical Indicators, Time Period Interval, 1).

4.5 Building deep learning models

While quality of input features is important to achieve more accurate results with deep learning models, designing deep learning architectures by considering different approaches proposed in the literature is also considerably important. Different deep

learning models with different architectures investigated in this study are discussed in this section.

4.5.1 Convolutional neural network

Previous researches have shown that CNNs are successful in image recognition and image classification problems. On the other hand, there are several researches which use CNNs for solving time series forecasting and natural language processing problems. In this study, CNN is selected to be investigated as a classification model because of its success on image classification.

A CNN is composed of multiple layers consisting of convolution layers, pooling layers, and fully connected layers. In a convolutional layer, each neuron gets input from subarea of the previous layer. The subarea of previous layer is called as a receptive field. Each neuron applies a specific function to its input values coming from receptive field of previous layer and computes an output value in order to pass it to next neuron. The applied function is determined by weights and bias. The vector of weights and the bias are called filters. Weights and biases are trainable parameters of neural network and the learning process iteratively adjusts these weights and biases (Dheir, Mettleq, Elsharif, & Abu-Naser, 2019). Filter size of convolution layer specifies the dimension of output. Kernel size of convolution layer shows width and height parameters of convolution window. The pooling layer uses a function to reduce the number of parameters and computation by down-sampling (Ke et al., 2018).

The proposed CNN model in this study is given in Figure 6. Input layer of the model has shape (15, n) where 15 is the time period interval for calculating technical indicators and n is the number of technical indicators given to the model. In other words,

input layer has same shape as reshaped data. The proposed model consists of two convolutional layers which are followed by one max pooling layer. First convolutional layer has filter size 32 and kernel size 3 with RELU activation, and second one has same kernel size and activation function but different filter size which is 64. Max pooling layer has pooling window size of 2. The max pooling window finds max value of its corresponding area. Dropout is applied after max pooling layer with rate 0.25 in order to reduce overfitting. Dropout removes a given percentage of the nodes in order avoid overfitting. These removed nodes join the network again with their original weights. While the dropout reduces overfitting, it also increases training speed.

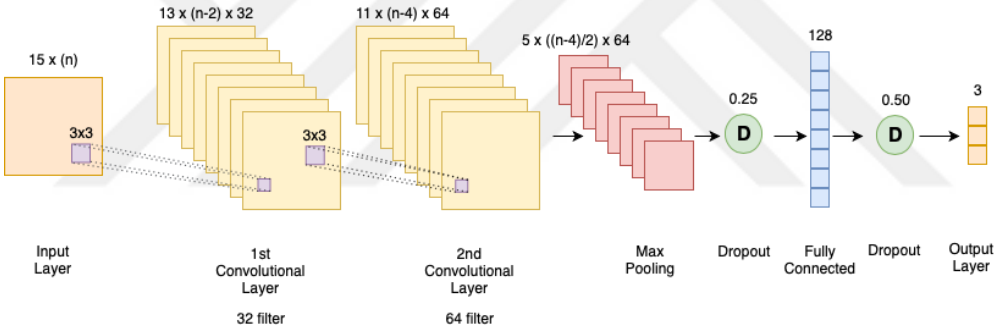

Figure 6. CNN as base model

After output of these layers are flattened, two dense layers are used with dropout layer between them. The activation functions of dense layers are RELU and softmax, respectively. Last layer with softmax activation makes size of its output equal to the number of output classes which is 3. The dropout applied between them has rate 0.5.

4.5.2 Attention mechanism

Attention mechanism is a component of neural network architecture and it achieves a significant breakthrough in neuroscience, natural language processing and image

captioning. There are various types of attention mechanism. Visual attention is the one widely used for image captioning and it is used for image classification in this study. Attention mechanism, as its name suggests, pays attention on specific parts of input instead of using all parts of input. This section explains how to use an attention-based encoder-decoder network for visual attention.

Attention mechanism divides the image into predetermined number of parts and CNN computes representations of each part. CNNs are used as encoder of attention mechanism. In other words, CNN model encodes the image and produces hidden states. On the other hand, the decoder of attention decodes the hidden states by using a RNNs or more specifically, LSTM units. As it is stated before, the attention mechanism is focusing on the relevant parts of the image. As a result, the decoder only uses specific parts of the image and it generates an output (Heuritech, 2019).

The study conducted by Xu et al. (2015) uses soft and hard attention mechanisms separately. However, in this research only soft attention mechanism is used which was proposed by Bahdanau, Cho, and Bengio (2015). Bahdanau et al. (2015) is one of the earliest studies on attention mechanisms and it has been used by many researchers.

In order to visualize attention-based encoder-decoder network used in this study, Figure 7 is given. Moreover, Figure 8 shows the soft attention mechanism in detail. The CNN model used as an encoder is the model that is given in Figure 6. The layers before flattening, including flatten layer itself, are used as encoder of attention mechanism. In

other words, the dense layers and dropout layer between them are excluded.

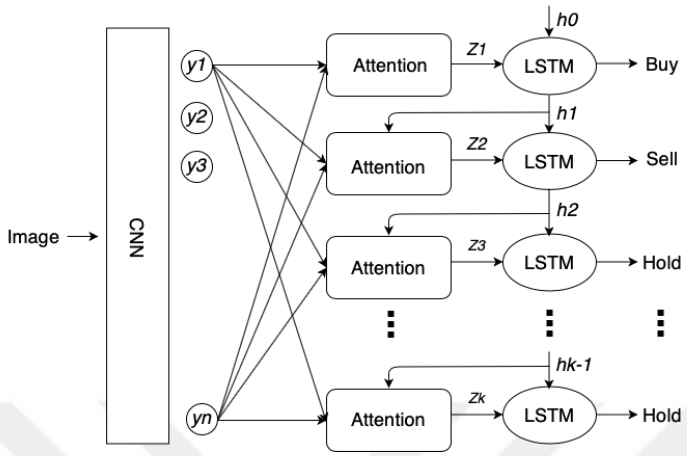

Figure 7. Attention mechanism with CNN encoder and LSTM decoder

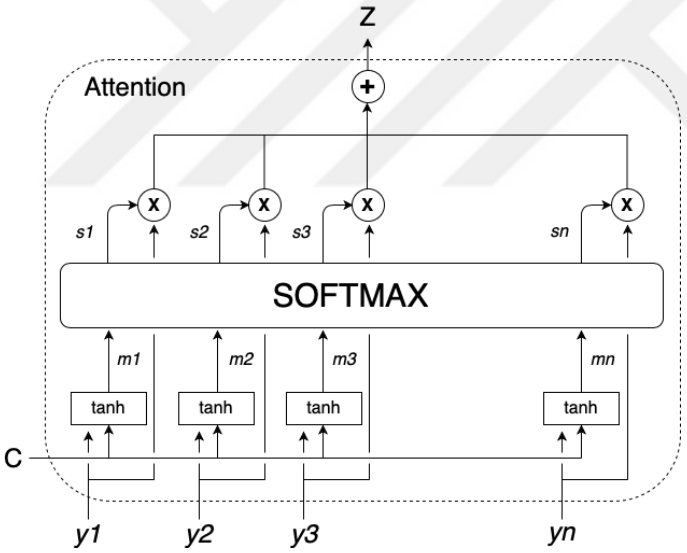

Figure 8. Soft attention mechanism

The second encoder model is shown in Figure 9. The second encoder model has three sequential convolution layers with same kernel size which is 3 and different filter sizes which are 64, 256 and 512. The activation function used in each convolutional layer is RELU. The filter map size of the final convolution layer shows the size of that output layer and 512 outputs are flattened before attention mechanism is applied on this

layer.

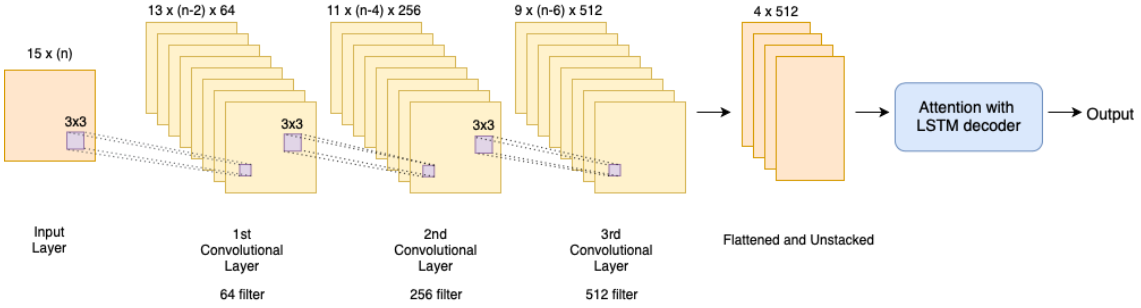

Figure 9. Second CNN as an encoder of attention

4.5.3 Model evaluation

There are several different deep learning algorithms with several different parameters, and one needs to evaluate the performance of the models generated with a particular deep learning architecture for a given set of parameters. In order to come up with the best machine learning model, performances of those models have to be evaluated using a holdout data set. In order to reach reliable conclusions in model selection, time series cross validation is employed in this study.

Cross validation is a re-sampling procedure used to evaluate machine learning models in a robust way. Its main goal is to test the generalization capability of created machine learning models and in order to flag problems like overfitting. In cross validation, data is split into pre-determined number of separate folds to obtain data sets for training and testing. In this study, the data set is split into 10 folds according to their date times, where each of the last 5 folds is used once as test set and previous 5 folds are used as train set. Figure 10 shows the time series cross validation model and the dates which are the starting date time of the train set and ending date time of the corresponding test set. As it can be seen, five different models are trained with these data

sets and corresponding performances of these five models are obtained and then aggregated to conclude about the performance of the specified deep learning algorithm. Since collected data has 4002 records and data set is split into 10-folds, train data has 2002 records and test data has 400 records for each validation run. Size of the training data set might change since cleaning, oversampling and augmentation methods are applied on these records.

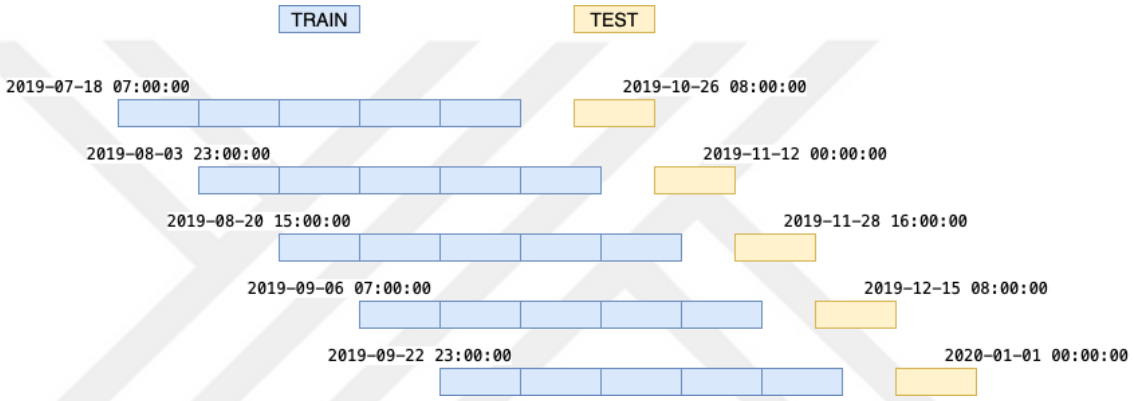

Figure 10. Time series cross validation model with train and test

In order to split the data set as train and test data, TimeSeriesSplit function of python library scikit-learn is used (Pedregosa et al., 2011). Since hourly data is collected, 400 records allocated for testing corresponds to approximately 16 days. Each block in the Figure 10 has approximately 16-day records. Therefore, the test result shows the 16 day accuracy of the model.

Table 6 shows the formulas the performance measures used in this research. In order to explain performance metrics and True Positive, False Negative, False Positive, True Negative in their formulas, confusion matrix of the Hold output class is given in Table 5. As it can be seen in Table 5, True Positive (TP) means the hold labelled instance is predicted as hold which is its ground truth label. False Negative (FN) means

the hold labelled instance is predicted incorrectly as buy or sell. False Positive (FP) means the buy or sell labelled instance is again incorrectly predicted as hold. Moreover, True Negative (TN) in Table 5 means the buy or sell labelled instance is predicted correctly as buy or sell.

Table 5. Confusion Matrix for the Hold Class

		PREDICTED		
		HOLD	BUY	SELL
ACTUAL	HOLD	TP_{HH}	FN_{HB}	FN_{HS}
	BUY	FP_{BH}	TN_{BB}	
	SELL	FP_{SH}		TN_{SS}

Table 6. Formulas of Performance Metrics

Metric	Formula
Precision	$\frac{TP}{TP + FP}$
Recall	$\frac{TP}{TP + FN}$
F1 Score	$2 * \frac{Precision * Recall}{Precision + Recall}$
Accuracy	$\frac{TP + TN}{TP + TN + FP + FN}$

Precision gives the number of correctly predicted hold instances. Recall shows the correctly predicted hold instances among all instances whose ground truth labels are hold. F1 score is harmonic mean of recall and precision. Accuracy is the percentage of correctly predicted instances. In order to compare the performances of the deep learning

models, this study uses macro average F1 score. This is due to the fact that the labelling algorithm results in imbalance data set. Even training data set become balanced with oversampling and augmentation methods, test performance metrics are still obtained by evaluating the models on imbalanced test data set.

CHAPTER 5

EXPERIMENTS AND RESULTS

Different deep learning algorithms are introduced in Section 4.5. This chapter explains experiments conducted by training different deep learning models with six feature groups that are composed of different set of technical indicators. Each model is trained with these groups for once. First of all, each model is trained with all technical indicators and this feature group is called all indicators. Then, each model is trained only with the technical indicators that belong to the trend, momentum, volume and volatility types separately. Finally, models are trained with top 15 best technical indicators that are selected by considering mutual information and these indicators are given in Table 4. By training with different feature groups, the effects of each indicator group on the models are examined.

Furthermore, effects of oversampling and data augmentation methods are investigated by further experiments. All deep learning models used by this study are trained with batch size of 1028 and for 200 epochs. As it is mentioned in Section 4.4.4, the data is imbalanced and number of buy and sell data is considerably low as compared to the number of hold data. First, the models are trained with imbalanced data set. While majority class is predicted almost hundred percent accurate, most of sell and buy data are also predicted as hold even though all features are present in training. It is the side effect of training any model with imbalanced data. In order to solve the problem, the imbalanced data is either oversampled or augmented by using the methods explained in Section 4.4.4.

In this study, evaluation of the models is performed by considering macro average F1 score. However, other accuracy measures are also provided for some models in order to compare their performances in more detail.

5.1 Convolutional neural network

The CNN model used in this research is explained in Section 4.5 and visualized in Figure 6. Initially, this model is trained with all technical indicators without using any oversampling or augmentation methods. The results obtained by the best performing one among the five test sets is given as its confusion matrix in Table 7. If the confusion matrix is analyzed, it can be said that almost all hold data are predicted correctly. But in contrast, almost all buy and sell data are predicted incorrectly as hold. Other confusion matrices obtained are similar to this one. As it is previously mentioned, the model trained with imbalanced data tends to predict minority classes as a member of majority class.

Table 7. Sample Confusion Matrix of CNN Model Trained with Imbalanced Data

		PREDICTED		
		HOLD	BUY	SELL
ACTUAL	HOLD	332	1	7
	BUY	25	1	0
	SELL	25	0	4

In order to improve the performance of the models trained with imbalanced data set, oversampling and augmentation methods are applied separately. At first, three oversampling methods, direct copying, SMOTE and ADASYN, are applied respectively

in order to analyze and compare their effects on the model performance. Table 8 shows model performances averaged over cross validation runs obtained by applying oversampling methods. It also includes the performance of the model which is trained with imbalanced data, named as None.

If scores of the models trained by applying three oversampling methods are compared to the one trained with imbalanced data set, None, the scores obtained by None are low as compared to others except for the accuracy. Since the number of hold data is considerably high in imbalanced data set and almost all of them are predicted correctly, it obtains higher accuracy score than other models. However, considering macro average F1 metric, model trained with imbalanced data set performed far worse than all the others.

According to the scores given in Table 8, direct copying obtained the highest macro average F1 score while SMOTE obtained the highest accuracy but the second highest macro average F1 score. ADASYN technique has the lowest macro average F1 score but it has the second highest accuracy value.

Table 8. The Results of CNN Models Trained with Oversampled Data

		None	Direct Copying	SMOTE	ADASYN
	Accuracy	0.8616	0.7660	0.8171	0.7840
Macro Average	F1 Score	0.3523	0.5573	0.5529	0.5523
	Recall	0.3599	0.6681	0.5893	0.6317
	Precision	0.4348	0.5213	0.5469	0.5206
Weighted Average	F1 Score	0.7930	0.8066	0.8230	0.8040
	Recall	0.8616	0.7660	0.8171	0.7840
	Precision	0.7808	0.8460	0.8370	0.8386

After analyzing models oversampling methods, two augmentation methods, jittering and time warping, are applied with different standard deviation values of 0.01, 0.10 and 1.00. Table 9 shows model performances averaged over cross validation runs obtained by applying jittering and time warping methods. It also includes performance of the model trained with imbalanced data, None. Similar to the results obtained by applying oversampling methods, models trained with augmented data sets achieved better scores than score of None except for the accuracy.

Table 9. The Results of CNN Models Trained with Augmented Data

		None	Jittering			Time Warping		
	Std Dev	-	0.01	0.10	1.00	0.01	0.10	1.00
	Accuracy	0.8616	0.6650	0.6344	0.6386	0.6289	0.6460	0.6815
Macro Average	F1 Score	0.3523	0.4507	0.4501	0.4203	0.4317	0.4327	0.4627
	Recall	0.3599	0.5962	0.6241	0.5665	0.5958	0.5731	0.5813
	Precision	0.4348	0.4699	0.4588	0.4538	0.4615	0.4771	0.4721
Weighted Average	F1 Score	0.8092	0.6955	0.6768	0.6660	0.6609	0.6622	0.7127
	Recall	0.8616	0.6650	0.6344	0.6386	0.6289	0.6460	0.6815
	Precision	0.7808	0.8285	0.8285	0.8212	0.8293	0.8297	0.8223

Jittering applied with standard deviation of 0.01 shows the highest performance among other jittering methods taking macro average F1 score as the basis. On the other hand, time warping with standard deviation of 1.00 achieves the best performance among other time warping methods. Furthermore, the best performed time warping method outperforms the best performed jittering method. Nevertheless, models trained utilizing augmentation methods obtained approximately 0.1 less macro average F1 score than the ones trained utilizing oversampling methods with CNN architecture.

5.2 Attention mechanism

In this section, two different attention mechanisms which use different CNN architectures as their encoders are explained and experimental results regarding models trained with these architectures are given. The data set used in these experiments is the same data set that was used to train models with CNN-only architectures. Due to the imbalanced class distribution observed in the labelled data set, the effects of applying three oversampling and two augmentation methods are also analyzed. Performance results obtained from the model trained with imbalanced data are not reported since the results are poor similar to ones obtained by CNN-only models. First, experiments are performed with attention mechanism which uses CNN model given in Figure 9 as its encoder. The first attention mechanism is named as Attention 1. Then, CNN model in Figure 6 is used as the encoder of the second attention mechanism proposed in this study. The second attention mechanism is called as Attention 2.

5.2.1 Experiments with the first attention mechanism

Table 10 shows the performance results obtained from the Attention 1 models trained with data set oversampled using three oversampling algorithms. ADASYN technique achieves the highest macro average F1 score and SMOTE follows it. As compared with the macro average F1 scores obtained by the CNN-only models, Attention 1 models achieve higher scores.

If Table 11 is investigated, jittering with the highest standard deviation and time warping with the lowest standard deviation result in the highest accuracy and macro average F1 scores among others. The results shown by the CNN-only models were exact opposites with respect to the method and size of the standard deviation. On the other

hand, the best performed time warping method again outperforms the best performed jittering one.

Table 10. The Results of Attention 1 Models Trained with Oversampled Data

		Direct Copying	SMOTE	ADASYN
	Accuracy	0.6903	0.7624	0.7354
Macro Average	F1 Score	0.5435	0.5566	0.5579
	Recall	0.7625	0.6763	0.7193
	Precision	0.5118	0.5208	0.5188
Weighted Average	F1 Score	0.7379	0.7913	0.7736
	Recall	0.6903	0.7624	0.7354
	Precision	0.8674	0.8473	0.8576

Table 11. The Results of Attention 1 Models Trained with Augmented Data

		Jittering			Time Warping		
	Std Dev	0.01	0.10	1.00	0.01	0.10	1.00
	Accuracy	0.5885	0.6170	0.6355	0.6471	0.5899	0.5941
Macro Average	F1 Score	0.3938	0.4058	0.4098	0.4110	0.3908	0.4083
	Recall	0.5645	0.5524	0.5550	0.5337	0.5583	0.5789
	Precision	0.4323	0.4382	0.4342	0.4460	0.4682	0.4410
Weighted Average	F1 Score	0.6139	0.6445	0.6623	0.6659	0.5985	0.6291
	Recall	0.5885	0.6170	0.6355	0.6471	0.5899	0.5941
	Precision	0.8142	0.8141	0.8070	0.8108	0.8233	0.8190

5.2.2 Experiments with the second attention mechanism

This section discusses the results obtained from the last model, Attention 2 where the attention mechanism uses the first five layers of CNN architecture depicted in Figure 6

as its encoder. The overall performance of Attention 2 is worse than first two models when all the oversampling and augmentation methods are considered.

Table 12 and 13 shows the averages calculated over five test cross validation runs for each method. It can be seen that SMOTE and ADASYN methods outperform direct copying method. If augmentation methods are considered, standard deviation value of 1.0 achieves the best result for the jittering method is 1.00 which is the same value used for training Attention 1 models. On the other hand, time warping methods achieve the best result with standard deviation of 1.00. For Attention 2 models highest score achieved by utilizing time warping method is higher than the best score achieved by utilizing the jittering method.

Table 12. The Results of Attention 2 Models Trained with Oversampled Data

		Direct Copying	SMOTE	ADASYN
	Accuracy	0.4692	0.5404	0.5509
Macro Average	F1 Score	0.4155	0.4504	0.4550
	Recall	0.7704	0.7445	0.7554
	Precision	0.4699	0.4701	0.4702
Weighted Average	F1 Score	0.5308	0.6074	0.6165
	Recall	0.4692	0.5404	0.5509
	Precision	0.8818	0.8628	0.8651

When a detailed analysis is performed on the performances of models on individual classes, it is observed that Attention 2 models predict almost all buy and sell labelled data correctly. The side effect of this situation is that accuracy on hold labelled data substantially decreased since most of them are incorrectly predicted as buy or sell.

Table 13. The Results of Attention 2 Models Trained with Augmented Data

		Jittering			Time Warping		
	Std Dev	0.01	0.10	1.00	0.01	0.10	1.00
	Accuracy	0.4959	0.5115	0.5039	0.6152	0.5788	0.6608
Macro Average	F1 Score	0.3585	0.3930	0.4053	0.4214	0.4342	0.4658
	Recall	0.6499	0.6507	0.6926	0.6172	0.6524	0.6214
	Precision	0.4233	0.4464	0.4537	0.4600	0.4599	0.4646
Weighted Average	F1 Score	0.5232	0.5542	0.5435	0.6505	0.6178	0.6979
	Recall	0.4959	0.5115	0.5039	0.6152	0.5788	0.6608
	Precision	0.8458	0.8354	0.8493	0.8348	0.8416	0.8293

5.3 Feature selection

Feature selection is the process of selecting a subset of features most relevant to the machine learning task in order to create a successful model. In this section, performance results obtained from the models trained by feeding the best 15 technical indicators selected using mutual information are analyzed. Similarly, Sezer and Özbayoglu (2018) used 15 different technical indicators in their study and these technical indicators are given in Chapter 2.

Scores achieved by CNN-only and Attention 1 models are reported in Table 14. Since oversampling methods outperform augmentation methods, this section only provides results of the experiments made with direct copying, SMOTE and ADASYN oversampling methods. Moreover, only CNN-only and Attention 1 models are trained and tested in these experiments since Attention 2 models do not show remarkable performance as compared to the models generated by considering other two architectures. ATT1 in Table 14 is an abbreviation used for Attention 1 model.

Table 14. The Results of the Models Trained with Top 15 Features

		Direct copying		SMOTE		ADASYN	
	Model	CNN	ATT1	CNN	ATT1	CNN	ATT1
	Accuracy	0.6745	0.6432	0.7068	0.7087	0.6942	0.7052
Macro Average	F1 Score	0.5377	0.5299	0.5516	0.5648	0.5446	0.5458
	Recall	0.7867	0.8080	0.7618	0.8024	0.7708	0.7538
	Precision	0.5091	0.5129	0.5172	0.5243	0.5113	0.5129
Weighted Average	F1 Score	0.7258	0.6990	0.7516	0.7543	0.7419	0.7504
	Recall	0.6745	0.6432	0.7068	0.7087	0.6942	0.7052
	Precision	0.8744	0.8840	0.8685	0.8791	0.8693	0.8660

Even if there is a small difference between performances of CNN and Attention 1 models, Attention 1 shows higher performance than CNN model both with SMOTE and ADASYN oversampling methods. To conclude, the best performance score is obtained by training an Attention 1 deep learning model with data set oversampled with SMOTE when the best 15 features are used.

5.4 Feature ablation study

Technical indicators are grouped into four types according to their characteristics as discussed in Chapter 3. Table 1 and 2 show the indicators used in this research and the types they belong to. In order to further analyze the effects of indicator types on action prediction, feature ablation study is conducted on technical indicators. This section provides in depth analysis of the effects of four indicator types, namely momentum, trend, volatility and trend indicators. As it is indicated in Section 5.3, the following subsections only provide results of CNN and Attention 1 models trained utilizing three oversampling methods. ATT1 in following tables is as an abbreviation for Attention 1.

5.4.1 Momentum indicators

This section provides performance results of CNN and Attention 1 models which use 33 momentum indicators as their features. Moreover, the models oversample data with direct copying, SMOTE and ADASYN methods. Table 15 shows averages of the scores obtained from five cross validation runs of each model. Attention 1 with SMOTE outperforms all other models.

Table 15. The Results of the Models Trained with Momentum Indicators

		Direct copying		SMOTE		ADASYN	
	Model	CNN	ATT1	CNN	ATT1	CNN	ATT1
	Accuracy	0.7735	0.6803	0.7629	0.7430	0.7613	0.7229
Macro Average	F1 Score	0.5556	0.5401	0.5615	0.5640	0.5540	0.5429
	Recall	0.6493	0.7744	0.6877	0.7257	0.6708	0.7142
	Precision	0.5177	0.5152	0.5252	0.5286	0.5194	0.5114
Weighted Average	F1 Score	0.7985	0.7248	0.7916	0.7771	0.7897	0.7623
	Recall	0.7735	0.6803	0.7629	0.7430	0.7613	0.7229
	Precision	0.8416	0.8739	0.8515	0.8618	0.8472	0.8557

5.4.2 Volume indicators

In this section, prediction performances of Attention 1 models which uses volume indicators as features are given. Table 16 shows performances of the models trained with volume indicators.

Results of CNN model for volume indicators can not be acquired since five features (volume indicators) are not sufficient for training a CNN model with the underlying architecture given in Figure 6. In order to train a CNN model with volume indicators, either number of volume indicators needs to be increased or architecture of

CNN model needs to be changed. Attention 1 with SMOTE outperforms other two oversampling methods when volume indicators are used.

Table 16. The Results of the Models Trained with Volume Indicators

		Direct copying	SMOTE	ADASYN
	Accuracy	0.5533	0.6724	0.5919
Macro Average	F1 Score	0.3588	0.3766	0.3579
	Recall	0.4733	0.4094	0.4334
	Precision	0.3892	0.3873	0.3871
Weighted Average	F1 Score	0.6111	0.7122	0.6443
	Recall	0.5533	0.6724	0.5919
	Precision	0.7759	0.7717	0.7694

5.4.3 Volatility indicators

This section shows results of CNN and Attention 1 models which use 12 volatility indicators as features. Table 17 shows performances of the models where Attention 1 model trained with data set oversampled with ADASYN outperforms other two models, but it has close results with Attention 1 with SMOTE.

Table 17. The Results of the Models Trained with Volatility Indicators

		Direct copying		SMOTE		ADASYN	
	Model	CNN	ATT1	CNN	ATT1	CNN	ATT1
	Accuracy	0.1599	0.4111	0.4572	0.4990	0.3937	0.5055
Macro Average	F1 Score	0.1439	0.2601	0.2724	0.2970	0.2451	0.3073
	Recall	0.3585	0.3606	0.3749	0.3837	0.3561	0.3708
	Precision	0.3502	0.3440	0.3549	0.3600	0.3458	0.3518
Weighted Average	F1 Score	0.1824	0.4893	0.4965	0.5550	0.4522	0.5863
	Recall	0.1600	0.4111	0.4572	0.4990	0.3937	0.5055
	Precision	0.7698	0.7772	0.7731	0.7806	0.7711	0.7663

5.4.4 Trend indicators

This section shows results of CNN and Attention 1 models which use 21 trend indicators as features. Table 18 gives the performance results of the models. Attention 1 with SMOTE is the best performed model among the others.

Table 18. The Results of the Models Trained with Trend Indicators

		Direct copying		SMOTE		ADASYN	
	Model	CNN	ATT1	CNN	ATT1	CNN	ATT1
	Accuracy	0.4075	0.4202	0.4569	0.5032	0.4472	0.4325
Macro Average	F1 Score	0.3364	0.3312	0.3539	0.3647	0.3503	0.3423
	Recall	0.5977	0.5573	0.5693	0.5306	0.5725	0.5852
	Precision	0.4000	0.3910	0.3998	0.3939	0.3979	0.3996
Weighted Average	F1 Score	0.4754	0.4899	0.5253	0.5792	0.5211	0.4977
	Recall	0.4075	0.4202	0.4569	0.5032	0.4472	0.4325
	Precision	0.7947	0.7790	0.7884	0.7794	0.7910	0.7950

If results of feature ablation study are further analyzed, training models with momentum indicators achieves the best performance. The highest macro average F1 score 56.40% of momentum indicators are very close to score of top 15 features using Attention 1 with SMOTE which shows the highest macro average F1 score in this study. On the other hand, volatility indicators show the worst performance.

CHAPTER 6

CONCLUSIONS AND MANAGERIAL IMPLICATIONS

This study is conducted for increasing the accuracy of forecasting cryptocurrency price movements represented as trading decisions made by deep learning models. Forth is purpose, numerous technical indicators are investigated and they are used for generating new features to be fed into deep learning algorithms. The technical indicators are grouped into six feature groups which include four technical indicator types and these are used as features for training deep learning models. One of these groups contains 15 indicators which are the top listed indicators related to the target variable according to the mutual information metric. The last group contains all the technical indicators considered in the study as features. By these approaches, effects of different technical indicator types and groups are investigated together with different oversampling techniques, augmentation methods and deep learning architectures.

Time series financial data is converted into images and classified into trading decisions by a sliding window algorithm. Oversampling and augmentation methods are adapted to this study since the classification algorithm generates imbalanced dataset. The data set has huge amounts of data belonging to the hold decision class as compared to buy and sell classes which results in imbalance. Thus, buy and sell labelled data become minority classes while hold labelled data becomes the majority class. When deep learning models are trained purely with imbalanced data, the models predict almost all hold labelled data correctly but they also predict buy and sell labelled data incorrectly as hold. That is why, in this thesis, several different oversampling methods as well as

augmentation methods are investigated to deal with the imbalanced class distribution problem.

The deep learning models used in this research are CNN and two soft attention mechanisms, named as Attention 1 and Attention 2. Generated deep learning models are evaluated by using time series cross validation which contains five runs. For each model, aggregated values of accuracy, F1 score, recall and precision scores are reported as performance metrics but evaluation is done using macro average F1 score.

The model with Attention 1 and SMOTE oversampling method using top 15indicators achieves the highest macro average F1 score of %56.48 in this study. If the top 15 indicators are analyzed, it can be said that types of all top 15 indicators are momentum, but MFI. MFI in top 15 indicators is a volume indicator. The model with Attention 1 and SMOTE using momentum indicators has the second highest macro average F1 score of %56.40. The reason behind the success of models with momentum indicators and top 15 indicators is a shallow market problem. If large amount of Bitcoin is traded, it is inevitable to see the effects on market price since cryptocurrency market has relatively low trade volume compared to traditional stock markets (Böhme, Christin, Edelman, & Moore, 2015). Moreover, herding effect in cryptocurrency market has an impact on the success of models with momentum indicators and top 15 indicators. Herding occurs when traders follow the market consensus instead of private information or knowledge (Poyser, 2018). Traders trade in the same direction as a result of herding effect and this behavior determines the price changes. Traders influenced by herding effect mostly follow the momentum behaviors.

Oversampling methods have more promising results than augmentation methods. Attention 1 and CNN models both give good results with oversampling methods but

performances of them varies according to feature group and oversampling method. CNN outperforms Attention 1 when it is used with direct copying method with different feature groups except volatility indicators. Moreover, for the models generated with momentum and trend indicator feature groups and utilizing ADASYN, the performance of CNN is better than Attention 1.

Augmentation methods are not successful enough as compared to oversampling methods to overcome the problems occurred due to the imbalanced data set. Besides, their computation time is so long compared to oversampling methods that they might not be preferable methods in real-life trading. Jittering and time warping augmentation methods with different standard deviation values are performed. Among the two augmentation methods, it can be said that the best performed time model created using warping method always outperforms best performed model created using jittering method.

While oversampling methods are successful in mitigating the imbalanced data problem, applying augmentation methods end up less successful. The results of augmentation methods show that models using the augmented training data mostly predict buy and sell so that most of the hold data is predicted as buy or sell as well. Same results are acquired by using Attention 2 since it also mostly predicts data as buy or sell, hence reducing the precision of these classes and recall of hold class. As a result, it can be said that Attention 1 and CNN outperform Attention 2.

The two attention mechanisms use same LSTM architecture as their decoders. The difference between Attention 1 and Attention 2 is that they use different architectures of CNN as their encoders. Performance of Attention 1 is higher than Attention 2. It is clear that changing architecture of attention mechanism affects the

performance of resulting model. As a result, the study shows the importance of investigating different architectures that has to similar deep learning architecture in order to increase performance of the models.

As it is indicated in Section 5.3, Sezer and Özbayoglu (2018) train their CNN model with 15 technical indicators in their study. They use two different daily price data set which belong to stocks of Dow 30 and ETFs. They resample their imbalanced data set before training using the direct copying method. Evaluating the trained model on each test data results in accuracy values 58% and 62% respectively. If only their data set is changed to data set collected in this study and use the 15 technical indicators, CNN model and resampling method selected by Sezer and Özbayoglu (2018), the accuracy value become 60%. However, the accuracy values increase to 64% and 62% respectively if resampling method is changed to SMOTE and ADASYN. Even if Attention 1 architecture is used for training, accuracy values of direct copying, SMOTE and ADASYN become 67%, 68% and 66% respectively. Moreover, the accuracy values become approximately 70% if top 15 technical indicators selected according to the mutual information criteria are used. Finally, the means of accuracy value are 81% and 76% respectively if all 71 technical indicators are used for training CNN and Attention 1 models. Since Sezer and Özbayoglu (2018) use accuracy value as a performance evaluation metric, the comparison to their study is done with accuracy values.

The results show that increasing the number of features is important for increasing the accuracy whereas selecting the best features among all these features also gives considerable results. On the other hand, different feature groups of technical indicators have varying results on contrast to results of the study conducted by Vargas et al. (2018) although their feature groups are totally different than this study. Vargas et al.

(2018) conclude their study by the claim that different set of technical indicators do not have different effects on accuracy results. However, this study shows that momentum indicators have considerably higher results than other types of indicators. Macro average F1 score of momentum indicators is 56.40% which are trained by using Attention 1 and SMOTE oversampling method. This result is really close to the result obtained by top 15 indicators using Attention 1 and SMOTE which has the best result in this study. On the other hand, volatility indicators show the worst results among four technical indicator types.

Finally, current study also shows the importance of data preprocessing steps. Data cleaning and min-max scaling also have effects on the performance of deep learning models.

The methods used in this research can be subject of researches related to timeseries forecasting and classification problems. Moreover, the results are promising for many traders, specifically the ones who are interested in cryptocurrency markets. Technical analysts can also take advantage of this study in terms of technical indicator categorization and selection. Since the financial trading approach proposed in Chapter4 is used with real world transaction cost in the trading simulation, there is no obstacle to use it in real world. However, the financial trading algorithm works with consecutive buy and sell outputs, which is a simple approach, so this makes the model depending on the order of output labels. As a result, a more robust financial trading algorithm can be constructed by traders, investors or financiers.

CHAPTER 7

FURTHER RESEARCH

The results of this study are promising since it has high accuracy values as a results of deep learning models with oversampling techniques. However, there are further research possibilities in order to increase the performance of the models even higher.

The attention mechanism used by this study is soft attention mechanism which was used for image captioning previously by Xu et al. (2015). Using different architectures of CNN model as an encoder already shows its effects on performance. In order to take a step forward, different LSTM architectures can also be experimented as a decoder. Furthermore, hard attention mechanism which is also proposed by Xu et al. (2015) can also be trained and evaluated with images converted from time series data. Last but not least, the model with temporal attention for forecasting time series data which is built by Fan et al. (2019) can be studied on financial data.

Two of the augmentation methods proposed by Um et al. (2017) are adapted to this study but they do not give the expected results since their performances on financial time series data are low. Um et al. (2017) also propose different augmentation methods which can be exemplified as scaling, rotation and permutation. The methods can also be experimented with financial time series data. Moreover, there are different types of SMOTE for oversampling the imbalanced data. Even using SMOTE in this study has innovative results, other types of SMOTE can also be experimented.

As J. Chen and Tsai (2020) use, candlestick charts can also be included in this study as another technical analysis tool. Moreover, fundamental analysis tools can be used instead of technical analysis tools or together with technical analysis tools in order

to assess the performance of them on forecasting price movements. To illustrate, using financial news proposed by Vargas et al. (2018) can be further investigated.

Furthermore, the order of technical indicators and their mutual correlations can be taken into consideration while data set is converted into images. These can be seen as the steps for increasing the quality of training data. In respect of data, different cryptocurrencies can also be used in order to see effects of different data types as Alonso-Monsalve et al. (2020) suggested.

As it is indicated in Chapter 4, the simple financial trading algorithm mentioned in this study is not robust. Success of financial trading algorithm might depend on the order of predicted labels since consecutive buy and sell outputs are considered for transaction. As a result, more powerful trading approaches can be used or proposed to improve the financial performance of the deep learning model predictions.

APPENDIX A

FORMULAS OF MOMENTUM INDICATORS

$$Aroon\ Down = 100 * \frac{n\ period - period\ number\ since\ n\ period\ low}{n\ period} \quad (A.1)$$

$$Aroon\ Up = 100 * \frac{n\ period - period\ number\ since\ n\ period\ high}{n\ period} \quad (A.2)$$

$$Aroon\ Oscillator\ =\ Aroon\ Up\ -\ Aroon\ Down \quad (A.3)$$

$$Balance\ of\ Power\ =\ \frac{Close\ -\ Open}{High\ -\ Low} \quad (A.4)$$

$$Chande\ Momentum\ Oscillator\ =\ 100\ *\ \frac{sU\ -\ sD}{sU\ + sD} \quad (A.5)$$

$$sD\ =\ Sum\ of\ n\ period's\ down\ closes$$

$$sD\ =\ Sum\ of\ n\ period's\ down\ closes$$

$$Commodity\ Channel\ Index\ =\ \frac{TP\ -\ SMA\ of\ TP}{0.015\ *\ Mean\ Deviation\ of\ TP} \quad (A.6)$$

$$TP\ =\ Typical\ Price$$

$$SMA\ =\ Simple\ Moving\ Average$$

$$Mean\ Deviation\ =\ \frac{\sum_{p=1}^{n}(TP\ -\ SMA)}{n\ period}$$

$$Momentum\ =\ Price\ -\ Price_{-n} \quad (A.7)$$

$$Price_{-n}\ =\ Close\ price\ n\ period\ ago$$

$$Smoothed(+DM)\ =\ \sum_{p=1}^{n}(+DM) - \frac{\sum_{p=1}^{n}(+DM)}{n}\ + Current(+DM) \quad (A.8)$$

$$Plus\ Directional\ Movement\ (+DM)\ = Current\ High - Previous\ High$$

$$Smoothed\ (\text{-}DM) = \sum_{p=1}^{n}(\text{-}DM) - \frac{\sum_{p=1}^{n}(\text{-}DM)}{n} + Current\ (\text{-}DM) \tag{A.9}$$

$$Minus\ Directional\ Movement\ (\text{-}DM) = Current\ Low - Previous\ Low$$

$$Plus\ Directional\ Indicator\ (+DI) = 100 * \frac{Smoothed\ (+DM)}{ATR} \tag{A.10}$$

$$Minus\ Directional\ Indicator\ (\text{-}DI) = 100 * \frac{Smoothed\ (\text{-}DM)}{ATR} \tag{A.11}$$

$$ATR = Average\ True\ Range$$

$$Directional\ Movement\ Index\ (DX) = 100 * \frac{+DI - \text{-}DI}{+DI + \text{-}DI} \tag{A.12}$$

$$ADX(Average\ DX) = \frac{\left(Previous\ DX * (n-1)\right) + Current\ DX}{n} \tag{A.13}$$

$$ADX\ Rating = \frac{\left(Current\ ADX * (n-1)\right) + ADX_{-n}}{2} \tag{A.14}$$

$$Percentage\ Price\ Oscillator = 100 * \frac{12\ period\ MA - 26\ period\ MA}{26\ period\ MA} \tag{A.15}$$

$$Rate\ of\ Change = 100 * (\frac{Price}{Previous\ Price} - 1) \tag{A.16}$$

$$Rate\ of\ Change\ Percentage = \frac{Price - Previous\ Price}{Previous\ Price} \tag{A.17}$$

$$Rate\ of\ Change\ Ratio = \frac{Price}{Previous\ Price} \tag{A.18}$$

$$Rate\ of\ Change\ Ratio\ 100\ Scale = 100 * \frac{Price}{Previous\ Price} \tag{A.19}$$

$$Stochastic\ RSI\ \%K = 100 * \frac{Current\ RSI - Lowest\ RSI}{Highest\ RSI - Lowest\ RSI} \tag{A.20}$$

$$Lowest\ RSI = the\ lowest\ RSI\ over\ n\ period$$

$$Highest\ RSI = the\ highest\ RSI\ over\ n\ period$$

$$Stochastic\ RSI\ \%D = 3\ period\ moving\ average\ of\ Stochastic\ RSI\ \%K \quad \text{(A.21)}$$

$$Triple\ Exponential\ Average = \frac{Current\ EMA_3 - Previous\ EMA_3}{Previous\ EMA_3} \quad \text{(A.22)}$$

$$EMA_3 = n\ period\ EMA_2$$

$$EMA_2 = n\ period\ EMA_1$$

$$EMA_1 = n\ period\ EMA$$

APPENDIX B

FORMULAS OF TREND INDICATORS

$$Midpont\ over\ Period\ =\ \frac{Highest\ +\ Lowest}{2} \tag{B.1}$$

$$Midpont\ Price\ over\ Period\ =\ \frac{Highest\ High\ +\ Lowest\ Low}{2} \tag{B.2}$$

$$Typical\ Price\ =\ \frac{High\ +\ Low\ +\ Close}{3} \tag{B.3}$$

$$Average\ Price\ =\ \frac{High\ +\ Low\ +\ Open\ +\ Close}{4} \tag{B.4}$$

$$Median\ Price\ =\ \frac{High\ +\ Low}{2} \tag{B.5}$$

$$Weighted\ Close\ Price\ =\ \frac{High\ +\ Low\ +\ Close\ *\ 2}{4} \tag{B.6}$$

$$Double\ Exponential\ Moving\ Average = 2 * EMA_1 - EMA_2 \tag{B.7}$$

$$Triple\ Exponential\ Moving\ Average = 3 * EMA_1 - 3 * EMA_2 + EMA_3 \tag{B.8}$$

$$\begin{aligned} &if\ n\ period\ is\ even \\ &Triangular\ Moving\ Average\ =\ SMA_{\frac{n}{2}}\ of\ SMA_{\frac{n}{2}+1} \\ &SMA_{\frac{n}{2}}\ =\ \frac{n}{2}\ period\ simple\ moving\ average \\ &SMA_{\frac{n}{2}+1}\ = (\frac{n}{2} + 1)\ period\ simple\ moving\ average \end{aligned} \tag{B.9}$$

$$\begin{aligned} &if\ n\ period\ is\ odd \\ &Triangular\ Moving\ Average\ =\ SMA_{\frac{n+1}{2}}\ of\ SMA_{\frac{n+1}{2}} \\ &SMA_{\frac{n+1}{2}}\ = = \left(\frac{n+1}{2}\right)\ period\ simple\ moving\ average \end{aligned} \tag{B.10}$$

$$Hull\ Moving\ Average\ =\ WMA_{\sqrt{n}}\ of\ (2\ *\ WMA_{\frac{n}{2}}\ -\ WMA_n) \tag{B.11}$$

$$WMA_n\ =\ n\ period\ weighted\ moving\ average$$

$$WMA_{\frac{n}{2}}\ =\ \frac{n}{2}\ period\ weighted\ moving\ average$$

$$WMA_{\sqrt{n}}\ =\ \sqrt{n}\ period\ weighted\ moving\ average$$

APPENDIX C

DATA SAMPLE

Label	Date	Open	High	Low	Close	Volume	CMF6	CMF7	CMF8	CMF9
nan	18.07.2019 07:00	9726.9	9907.92	9673.15	9907.92	1963.84	nan	nan	nan	nan
nan	18.07.2019 08:00	9907.92	9911.64	9814.45	9834.58	1447.89	nan	nan	nan	nan
nan	18.07.2019 09:00	9834.58	10014.99	9803.99	9899.84	4128.15	nan	nan	nan	nan
nan	18.07.2019 10:00	9899.84	9905.35	9793.45	9803.97	1606.64	nan	nan	nan	nan
nan	18.07.2019 11:00	9803.97	9848.11	9703.24	9719.93	1873.62	nan	nan	nan	nan
0.0	18.07.2019 12:00	9719.93	9815.07	9700.71	9751.82	1732.84	0.4233	nan	nan	nan
0.0	18.07.2019 13:00	9751.82	9883.56	9736.04	9847.93	1550.9	0.3577	0.4723	nan	nan
0.0	18.07.2019 14:00	9847.93	9861.53	9758.75	9770.62	1471.46	0.3431	0.323	0.4312	nan
1.0	18.07.2019 15:00	9770.62	9771.57	9336.48	9398.21	10509.25	0.1586	0.2159	0.2121	0.2853
0.0	18.07.2019 16:00	9398.21	9493.35	9363.83	9407.49	4259.91	0.2028	0.1889	0.2326	0.2285
0.0	18.07.2019 17:00	9407.49	10371.77	9292.61	10305.6	20465.93	0.6759	0.647	0.6233	0.6105
0.0	18.07.2019 18:00	10305.6	10466.74	10254.65	10455.2	10027.94	0.7758	0.765	0.7386	0.7165
0.0	18.07.2019 19:00	10455.2	10662.05	10377.1	10501.89	10018.0	0.7164	0.7206	0.7132	0.6926
0.0	18.07.2019 20:00	10501.89	10584.31	10452.31	10532.22	3234.69	0.7313	0.7141	0.7182	0.7113
0.0	18.07.2019 21:00	10532.22	10732.85	10498.12	10503.68	5358.17	0.7799	0.6625	0.6483	0.6535
0.0	18.07.2019 22:00	10503.68	10627.96	10482.34	10518.24	3506.18	0.7785	0.7443	0.6386	0.6256
0.0	18.07.2019 23:00	10518.24	10664.57	10516.21	10611.05	2836.48	0.5789	0.7755	0.7431	0.6418
0.0	19.07.2019 00:00	10611.05	10719.87	10531.58	10594.66	2610.81	0.3572	0.5608	0.755	0.7254
0.0	19.07.2019 01:00	10594.66	10620.06	10519.26	10602.94	1457.67	0.3528	0.388	0.5761	0.7602
0.0	19.07.2019 02:00	10602.94	10776.54	10601.39	10638.35	3962.39	0.2608	0.319	0.3602	0.5375
2.0	19.07.2019 03:00	10638.35	10679.16	10586.26	10673.45	2016.67	0.4797	0.3396	0.383	0.4035
0.0	19.07.2019 04:00	10673.45	10718.97	10566.52	10585.03	1852.33	0.4901	0.435	0.3161	0.3593

CMF10	CMF11	CMF12	CMF13	CMF14	CMF15	CMF16	CMF17	CMF18	CMF19	CMF20
nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan
nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan
nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan
nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan
nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan
nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan
nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan
nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan
nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan
0.2904	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan
0.597	0.6199	nan	nan	nan	nan	nan	nan	nan	nan	nan
0.6993	0.6858	0.7021	nan	nan	nan	nan	nan	nan	nan	nan
0.6752	0.663	0.6523	0.6672	nan	nan	nan	nan	nan	nan	nan
0.6917	0.6751	0.6635	0.6533	0.6675	nan	nan	nan	nan	nan	nan
0.6488	0.6324	0.6184	0.6107	0.6021	0.6167	nan	nan	nan	nan	nan
0.6311	0.6272	0.6121	0.5991	0.5928	0.5849	0.5992	nan	nan	nan	nan
0.6293	0.6345	0.6306	0.616	0.6035	0.5971	0.5894	0.6032	nan	nan	nan
0.6303	0.6184	0.6237	0.6202	0.6063	0.5943	0.5886	0.5813	0.5948	nan	nan
0.731	0.6369	0.6252	0.6302	0.6266	0.6128	0.6009	0.595	0.5877	0.6009	nan
0.7225	0.6974	0.6126	0.6019	0.6071	0.6042	0.5915	0.5806	0.5758	0.5692	0.5822
0.5632	0.7345	0.7093	0.6251	0.6145	0.6193	0.6162	0.6036	0.5927	0.5875	0.5808
0.3852	0.5425	0.7154	0.692	0.6119	0.6017	0.6067	0.6039	0.5918	0.5813	0.5767

REFERENCES

Achelis, S. B. (2001). *Technical analysis from A to Z* (2nd ed.). New York, NY: McGraw-Hill.

Alonso Monsalve, S., Suárez-Cetrulo, A., Cervantes, A., & Quintana, D. (2020). Convolution on neural networks for high-frequency trend prediction of cryptocurrency exchange rates using technical indicators. *Expert Systems with Applications, 149*, 113250. doi: https://doi.org/10.1016/j.eswa.2020.113250

Angelis, L., & Stamelos, I. (2000). A simulation tool for efficient analogy based cost estimation. *Empirical Software Engineering, 5*, 35–68. doi: https://doi.org/10.1023/A:1009897800559

Appel, G., & Dobson, E. (2007). *Understanding MACD (Moving average convergence divergence)*. Milwaukee, WI: Traders Press.

Bahdanau, D., Cho, K., & Bengio, Y. (2015). *Neural machine translation by jointly learning to align and translate*. Paper presented at the 3rd International Conference on Learning Representations (ICLR 2015), San Diego, CA, USA. Retrieved from https://arxiv.org/abs/1409.0473

Böhme, R., Christin, N., Edelman, B., & Moore, T. (2015). Bitcoin: Economics, technology, and governance. *Journal of Economic Perspectives, 29*(2),213–38. doi: https://doi.org/10.1257/jep.29.2.21

Bollinger, J. (1992). Using Bollinger Bands. *Stocks & Commodities, 10*(2), 47–51. Retrieved from http://traders.com

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research, 16*, 321–357. doi: https://doi.org/10.1613/jair.953

Chen, A., Leung, M. T., & Daouk, H. (2003). Application of neural networks to an emerging financial market: Forecasting and trading the Taiwan stock index. *Computers & Operations Research, 30*(6), 901–923. doi: https://doi.org/10.1.1.472.2854

Chen, J. (2020, March 12). Understanding Volatility Measurements [Blog post]. Retrieved from https://www.investopedia.com/investing/understanding-volatility-measurements

Chen, J., & Tsai, Y. (2020). Encoding candlesticks as images for pattern classification using convolutional neural networks. *Financial Innovation, 6*(26). doi: https://doi.org/10.1186/s40854-020-00187-0

Colby, R. W. (2003). *The encyclopedia of technical market indicators* (2nd ed.). New York, NY: McGraw-Hill.

Cryptocurrency market capitalizations. (n.d.). Retrieved May 25, 2020, from CoinMarketCap Website: https://coinmarketcap.com

Dheir, I. M., Mettleq, A. S. A., Elsharif, A. A., & Abu-Naser, S. S. (2019). Classifying nuts types using convolutional neural network. *International Journal of Academic Information Systems Research (IJAISR)*, 12–18. Retrieved from https://philarchive.org

Durahim, A. O. (2016). Comparison of sampling techniques for imbalanced learning.*Yönetim Bilişim Sistemleri Dergisi, 2*(2), 181–191. Retrieved from https://dergipark.org.tr

Ebens, H. (1999). *Realized stock volatility* (Economics Working Paper Archive No. 420). Baltimore: The Johns Hopkins University, Department of Economics.

Ehlers, J. F. (2001). *Rocket science for traders: Digital signal processing applications*. Hoboken, NJ: John Wiley & Sons.

Etzkorn, M. (1997). *Trading with oscillators: Pinpointing market extremes – theory and practice.* Hoboken, NJ: John Wiley & Sons.

Falk, A., & Moberg, J. (2014). *Algorithmic trading using MACD signals* (Bachelor's Thesis). Retrieved from https://kth.diva-portal.org

Fan, C., Zhang, Y., Pan, Y., Li, X., Zhang, C., Yuan, R., . . . Huang, H. (2019). *Multi-horizon time series forecasting with temporal attention learning*. Paper presented at the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, Anchorage, AK, USA. Retrieved from https://www.semanticscholar.org/paper/Multi-Horizon-Time-Series-Forecasting-with-Temporal-Fan-Zhang/5d8272d3ee319cf205ec96597f7a828eecde1da8

Fischer, T., & Krauss, C. (2018). Deep learning with long short-term memory networks for financial market predictions. *European Journal of Operational Research, 270*(2), 654–669. doi: https://doi.org/10.1016/j.ejor.2017.11.054

Forman, J. (2004). Cross-market evaluations with Normalized Average True Range. *Stocks & Commodities, 24*(6), 60. Retrieved from http://traders.com

Granville, J. E. (1963). *New key to stock market profits*. Upper Saddle River, NJ: Prentice-Hall.

Guennec, A. L., Malinowski, S., & Tavenard, R. (2016). *Data augmentation for time series classification using convolutional neural networks*. Paper presented at ECML/PKDD Workshop on Advanced Analytics and Learning on Temporal Data, Riva Del Garda, Italy. Retrieved from https://halshs.archives-ouvertes.fr

Güreşen, E., Kayakutlu, G., & Daim, T. (2011). Using artificial neural network models in stock market index prediction. *Expert Systems with Applications, 38*(8), 10389–10397. doi: https://doi.org/10.1016/j.eswa.2011.02.068

He, H., Bai, Y., Garcia, E. A., & Li, S. (2008). *Adasyn: Adaptive synthetic sampling approach for imbalanced learning*. Paper presented at the 2008 International Joint Conference on Neural Networks (IJCNN 2008), Hong Kong, China. Retrieved from https://www.researchgate.net/publication/224330873_ADASYN_Adaptive_Synthetic_Sampling_Approach_for_Imbalanced_Learning

He, H., & Ma, Y. (2013). *Imbalanced learning: Foundations, algorithms, and applications*. Hoboken, NJ: John Wiley & Sons.

Heuritech. (2019, December 5). Attention mechanism [Blog post]. Retrieved from https://medium.com/heuritech/attention-mechanism-5aba9a2d4727

Huang, J., Li, Y. F., & Xie, M. (2015). A systematic analysis of data preprocessing for machine learning-based software cost estimation. *Information and Software Technology, 67*, 108–127. doi: https://doi.org/10.1016/j.infsof.2015.07.004

Jerez, J. M., Molina, I., García-Laencina, P. J., Alba, E., Ribelles, N., Martín, M., & Franco, L. (2010). Missing data imputation using statistical and machine learning methods in a real breast cancer problem. *Artificial Intelligence in Medicine, 50*(2), 105–115. doi: https://doi.org/10.1016/j.artmed.2010.05.002

Jiang, Z., & Liang, J. (2017). *Cryptocurrency portfolio management with deep reinforcement learning*. Paper presented at the Intelligent Systems Conference (IntelliSys), London, UK. Retrieved from https://arxiv.org/abs/1612.01277

Jovic, A., Brkic, K., & Bogunovic, N. (2015). *A review of feature selection methods with applications*. Paper presented at the 38th International Convention on Information, Communication Technology, Electronics, and Microelectronics (MIPRO 2015), Opatija, Crotia. Retrieved from https://www.semanticscholar.org/paper/A-review-of-feature-selection-methods-with-Jovic-Brkic/6dc7cc7ec2c016ddcf341f868bafe459ca156ab9

Kaufman, P. (1995). *Smarter trading*. New York, NY: McGraw-Hill.

Ke, Q., Liu, J., Bennamoun, M., An, S., Sohel, F., & Boussaid, F. (2018). Computer vision for human-machine interaction. *Computer Vision for Assistive Healthcare*, 127–145. doi: https://doi.org/10.1016/B978-0-12-813445-0.00005-8

Komisyonlar. (n.d.). Retrieved Jul 28, 2020, from BtcTurk Website: https://pro.btcturk.com/yardim/komisyonlar

Lane, G. C. (1984). Lane's Stochastics. *Stocks & Commodities, 2*(3), 87–90. Retrieved from https://forexstarmoon.com

Lemaître, G., Nogueira, F., & Aridas, C. K. (2017). Imbalanced-learn: A Python toolbox to tackle the curse of imbalanced datasets in machine learning. *Journal of Machine Learning Research, 18*(17), 1–5. Retrieved from http://jmlr.org

Long, W., Lu, Z., & Cui, L. (2018). Deep learning-based feature engineering for stock price movement prediction. *Knowledge-Based Systems, 164*, 163–173. doi: https://doi.org/10.1016/j.knosys.2018.10.034

Lui, Y., & Mole, D. (1998). The use of fundamental and technical analyses by foreign exchange dealers: Hong Kong evidence. *Journal of International Money and Finance, 17*(3), 535–545. doi: https://doi.org/10.1016/S0261-5606(98)00011-4

Murphy, C. (2019, June 29). The Difference Between Fast and Slow Stochastics [Blog post]. Retrieved from https://www.investopedia.com/ask/answers/05/062405.asp

Murphy, J. J. (1999*). Technical analysis of the financial markets: A comprehensive guide to trading methods and applications*. London, England: Penguin.

Naik, N., & Mohan, B. (2019). *Optimal feature selection of technical indicator and stock prediction using machine learning technique*. Paper presented at the International Conference on Emerging Technologies in Computer Engineering, Jaipur, India. Retrieved from https://www.researchgate.net/publication/333166969_Optimal_Feature_Selection_of_Technical_Indicator_and_Stock_Prediction_Using_Machine_Learning_Technique

Oberlechner, T. (2001). Importance of technical and fundamental analysis in the European foreign exchange market. *International Journal of Finance & Economics, 6*(1), 81–93. doi: https://doi.org/10.1002/ijfe.145

Patel, M. (2010*). Trading with Ichimoku Clouds: The essential guide to Ichimoku Kinko Hyo*. Hoboken, NJ: John Wiley & Sons.

Peachavanish, R. (2016). *Stock selection and trading based on cluster analysis of trend and momentum indicators*. Paper presented at the 24th International MultiConference of Engineers and Computer Scientists (IMECS 2016), Hong Kong, China. Retrieved from http://www.iaeng.org/publication/IMECS2016/IMECS2016_pp317-321.pdf

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., . . . Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research, 12*, 2825–2830. Retrieved from https://jmlr.csail.mit.edu

Poyser, O. (2018). *Herding behavior in cryptocurrency market* (Working Paper). Universitat Autònoma de Barcelona Department of Applied Economics.

Qin, Y., Song, D., Chen, H., Cheng, W., Jiang, G., & Cottrell, G. W. (2017). *A dual-stage attention-based recurrent neural network for time series prediction*. In Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, Melbourne, Australia. Retrieved from https://www.ijcai.org/Proceedings/2017/0366.pdf

Rashid, K. M., & Louis, J. (2019). *Window-warping: A time series data augmentation of IMU data for construction equipment activity identification*. Paper presented at the 36th International Symposium on Automation and Robotics in Construction (ISARC 2019), Banff, Canada. Retrieved from https://www.iaarc.org/publications/fulltext/ISARC_2019_Paper_152.pdf

Sezer, Ö. B., & Özbayoglu, A. (2020). Financial trading model with stock bar chart image time series with deep convolutional neural networks. *Intelligent Automation and Soft Computing, 26*(2). doi: https://doi.org/10.31209/2018.100000065

Sezer, Ö. B., & Özbayoglu, A. M. (2018). Algorithmic financial trading with deep convolutional neural networks: Time series to image conversion approach. *Applied Soft Computing, 70*, 525–538. doi: https://doi.org/10.1016/j.asoc.2018.04.024

Shynkevich, Y., McGinnity, T. M., Coleman, S. A., Belatreche, A., & Li, Y. (2017). Forecasting price movements using technical indicators: Investigating the impact of varying input window length. *Neurocomputing, 264*, 71–88. doi: https://doi.org/10.1016/j.neucom.2016.11.095

Thomsett, M. C. (2010). *Cmf–chaikin money flow: Changes anticipating price reversal*. Upper Saddle River, NJ: FT Press.

Tsang, W., & Chong, T. (2009). Profitability of the On-Balance Volume indicator. *Economics Bulletin, 29*(3), 2424–2431. Retrieved from http://www.accessecon.com/pubs/eb/

Um, T. T., Pfister, F. M. J., Pichler, D., Endo, S., Lang, M., Hirche, S., . . . Kulic, D. (2017). *Data augmentation of wearable sensor data for Parkinson's disease monitoring using convolutional neural networks*. Paper presented at the 19th ACM International Conference on Multimodal Interaction, Glasgow, Scotland. Retrieved from https://arxiv.org/abs/1706.00527

Vargas, M. R., Dos Anjos, C. E. M., Bichara, G. L. G., & Evsukoff, A. G. (2018). *Deep learning for stock market prediction using technical indicators and financial news articles*. Paper presented at the 2018 International Joint Conference on Neural Networks (IJCNN 2018), Rio, Brazil. Retrieved from https://www.researchgate.net/publication/328400101_Deep_Leaming_for_Stock_Market_Prediction_Using_Technical_Indicators_and_Financial_News_Articles

Wilder, J. W. (1978). *New concepts in technical trading systems*. McLeansville, NC: Trend Research.

Williams, L. (1985). The Ultimate Oscillator. *Stocks & Commodities, 3*(4), 140–141. Retrieved from http://williamspercentr.com

Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhudinov, R., . . . Bengio, Y. (2015). *Show, attend and tell: Neural image caption generation with visual attention*. Paper presented at the International Conference on Machine Learning, Lille, France. Retrieved from http://proceedings.mlr.press/v37/xuc15.pdf

Yazdi, S. H. M., & Lashkari, Z. H. (2013). Technical analysis of Forex by MACD indicator. *International Journal of Humanities and Management Sciences, 1*(2), 159–165. Retrieved from http://www.isaet.org/proceeding.php?type=4