Neural networks for validation of technical analysis for FOREX market price forecasting

Data Science Bsc. 6th Semester 2021

Bachelor Project Report

Christian Emil Haldan

IT-University

17th May, 2021

Table of Contents

1 Abstract

This research paper explores the validity of technical indicators for forecasting price fluctuations in the FOREX market. The overall research design was conducted by performing an extensive data preparation stage, followed by supervised machine learning to create labels for trading simulations, which is used to define the performance of each model. The models used in this research include the time series model Long Short-Term Memory (LSTM), alongside Convolutional Neural Networks (CNN). The base model used for performance comparison is a multivariate linear regression. The primary currency pair used in the project is the EURUSD, which is among the most popular FX-pairs in modern FOREX trading history. Data from 2012 to 2019 was used as training data, saving 2020 as a test set. Models were created using hyperparameter tuning and evaluated by using predicted labels for trading simulations. To further validate the performance of models, trading simulations were performed on the EURGBP, GBPUSD for the years 2015 - 2020. Trading simulations do not include trading strategies as the intention of the report is to emphasize a model's ability to predict future price fluctuations in the market.

Correlations of the feature and the target were arguably poor; however, trading simulations using neural network models for the EURUSD 2020 yielded results that were considered better than the baseline performance and random guessing. Additionally, applying the magnitude of predicted labels as a metric defining the confidence of predicting future price fluctuations was shown to yield results worth exploring in future research. Because of this, technical indicators were found mildly informative for neural networks, with the intent of predicting forex market volatility.

2 Introduction

The domain of artificial intelligence and machine learning has gained exceptional popularity within the research community over the past 20 years (2021, Artificial Intelligence Index report), with a near exponential growth in popularity over the past five years (see figure 1).

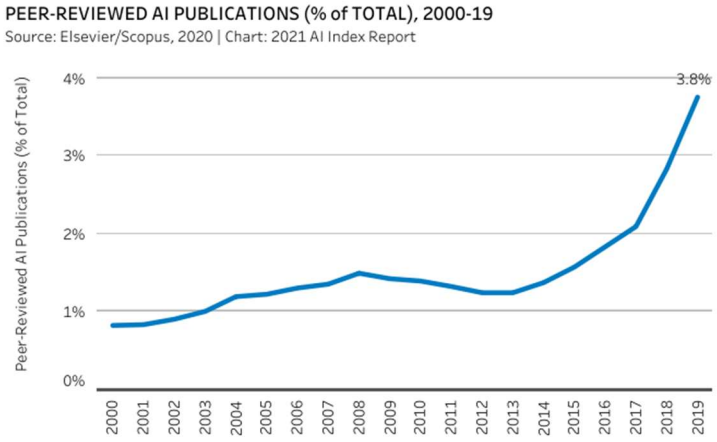


Figure 1: Peer-reviewed AI publications

The popular open access archive for scholarly articles, primarily within the field of physics, mathematics, and computer science, arXiv has seen a tremendous increase in published AI-related research papers (see figure 2) (2021, Artificial Intelligence Index report). The number of AI-related research papers published has almost tripled in the open-source archive since 2017, with 34 thousand AI-related research papers published in 2020. Companies, corporations, and institutions have started to adopt interest by incorporating AI using the business' available data to increase profit, gain business intelligence, or even enhance in-house solutions.

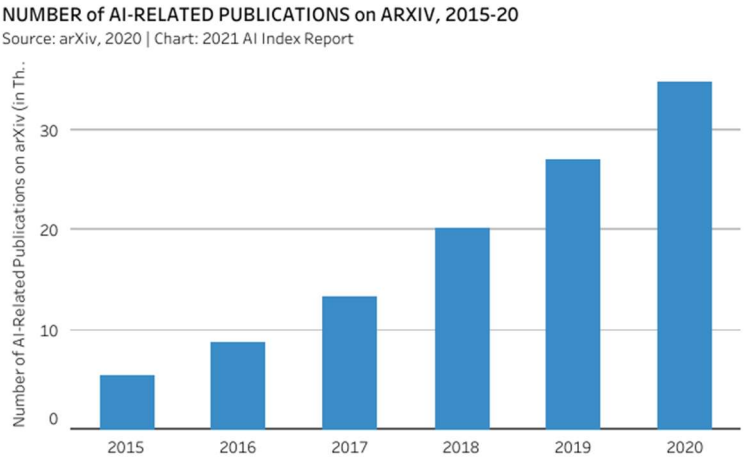


Figure 2: number of AI related publications pr. year on arXiv

Naturally, this includes corporations working within investments, trading, and market speculation. (2018, Barclay Hedge AI/Machine Learning in Investment Strategies) released a survey showing that 56% of hedge funds use applied AI or/and machine learning for investment and trading decision making. Within the population of fund managers who use AI for decision making, more than half let models dictate more than 40% or more of all investment decisions, and 19% of the fund managers within the previously stated population let models dictate 80%-100% of investment decisions.

One financial market is the Foreign Exchange market, which is the market where the data used in this project originates. The FOREX market is a marketplace that is open 24 hours a day, 5.5 days a week, and known for high volatility and high trading volumes, both due to leveraged trading. Leveraged trading is the act of trading with borrowed money. Trading volume defines the amount of money changing hands for a given period. Trading on the FOREX market is considered a zero-sum game if the cost of trading a currency pair is neglected.
For this reason, FOREX falls within the category of high risk, high reward investments. The market has a notorious history of retail investors ending up losing money. A common method of taking decisions during FOREX trading is using technical indicators for decision making. Technical analysis is a controversial topic within finance, with the debate being whether future prices are dependent on the price history, along with patterns shaped by it.
A brief introduction of the basic concepts of FOREX trading will be further elaborated in section 4 of this paper.

With a clear incentive to use machine learning for investment decision making and algorithmic trading, the motivation for this research comes from the popularity of machine learning used within the financial markets. Additionally, the scepticism and controversy regarding the validity of the predictive capabilities of technical analysis was a further motivation for the following research question: **To what extent are technical indicators informative for neural networks, with the intent of predicting forex market volatility?**

The statement of past price values for a currency pair, defining the future price of said pair, is arguably doubtful, however, some individuals claim to consistently make profitable investment decisions based on technical indicators. Data with some amount of correlation with target values is a task suitable for machine learning. With the assumption that the previous claim is valid, the hypothesis for the research question is as followed:

Technical indicators will not consistently define future price fluctuations. That being said, it is expected that the performance of models during a trading simulation will perform better than random guessing.

3 FOREX glossary

FOREX/FX Market: Foreign Exchange Markets; a decentralized market for trading currencies. In the perspective of a trader, the objective is exchanging one currency into another, expecting to gain profit, when exchanging the second-mentioned currency, back into the initial currency, due to a difference in exchange rates.

Currency pairs: A combination of two national currencies, defining an exchange rate.

Technical indicator: A statistical measure of previous OHLC prices. Different technical indicators exist, with different objectives.

Position: Buying or selling the first currency using the second of a currency pair. E.g. buying EUR with USD. A position can be placed, closed, and held, respectively defining the beginning of a trade, the end of a trade, and the time between the two events.

Order: An order defines the request for a position. Trading currency pairs requires an opposing buyer or seller for said pair.

OHLC: Open, High, Low, and Close prices for a given period. When visualized on a candle chart, a single candle represents said values for a given period. The data format is further described in section 5.1.

Pip: Although short for "point in percentage", a pip is represented as a 0.0001 difference in price. The number of pips gained in a trade is often the describing factor of the success of a trade, without referring to one's profit or loss in monetary units relative to the initial investment.

Spread: The difference between the asking price and bidding price of a currency pair.

Long: A long position is an act of placing a position on a currency pair, where an increase in the value of the currency pair will result in a profit.

Short: A short position is an act of placing a position on a currency pair, where a decrease in the value of the currency pair will result in a profit.

Free lunch: An investment or a series of investments with riskless profit.

4 Information regarding the FOREX market

This section will introduce information regarding FOREX trading, necessary to understand the incentive of exploring this market, the basics of how the market works, and the results section.

The FOREX market is a marketplace that is open 24 hours a day, 5.5 days a week. Individuals and institutions exchange currencies at an exchange rate, defined by the supply and demand for the currencies involved. A report from Triennial Central Bank states that the FOREX market exchange currencies worth $2 trillion per day as of April 2019. The market is very liquid, meaning that trade orders are executed instantly, which creates a position. The prior statement assumes that the order is placed within market hours due to an opposing buyer or seller nearly always being available.

An example of how a currency pair is traded will briefly be demonstrated below to give a basic understanding of the domain of data used for this paper.

Example: A retail trader wishes to buy the EURUSD at an exchange rate of 1.1, meaning that 1€ will cost the trader $1.1. Holding the position means that instead of owning $1.1, they now own 1€.
This position is held for some time until the trader decides to close their position by selling the EURUSD. When the trader closes their position, the new exchange rate is now 1.2, which means they exchange their 1€ for $1.2, effectively gaining $0.1.

In practice, much more happens, including leverages, margin impacts, lots, interest rates, spreads and more, which is un-important for the research in this paper. The subsection below will introduce the unit “pip”, which is the unit defining price changes for a currency pair. This unit keeps the point of the paper clear and concise.

4.1 FOREX unit of price difference (Pip)

In the FOREX market, the term pip is used as a standard unit measuring price differences. The term will be used frequently in this paper, motivating a further explanation of the term. A pip is an abbreviation of "point in percentage". Pips are calculated by the difference of two prices, multiplied by 10000. Another way of considering a pip is the change in the value of the 4^{th} decimal place in the exchange rate. This definition is currency independent except for currencies paired with the Japenese Yen (JPY), where a pip is the 2^{nd} decimal place in the exchange rate; however, that particular currency is not used in this paper.

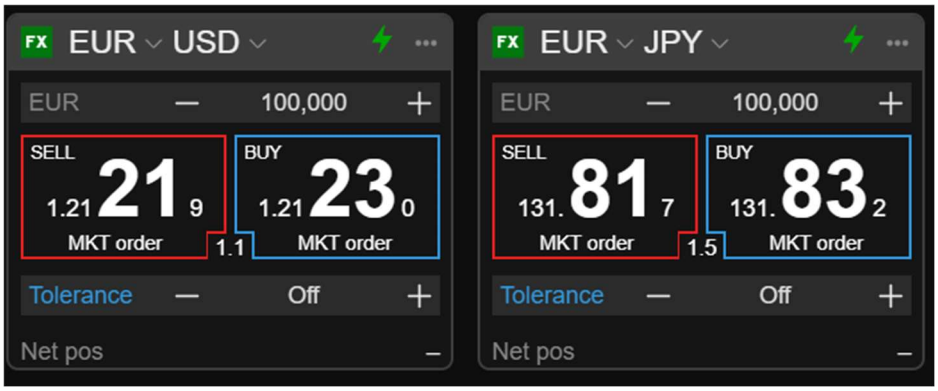


Figure 3: Screenshot of tool from SaxoTaderPro, illustrating a pip for EURUSD and EURJPY

Figure 3 shows a screenshot of an order placement tool found on the trading platform SaxoTraderPro. Two currency pairs (EURUSD and EURJPY) are displayed, containing the price of the currency pairs in the red and blue encapsulations, respectively being the bid and an asked price. In between the red and blue encapsulations is a small value of 1.1 on the EURUSD and 1.5 EURJPY. This value is the spread, which is the difference between the bid and asked price represented in pips.

4.2 FOREX market volatility

Exchange rates are often reasonably stable for short time frames, which introduces the need for leveraged trading. Leveraged trading within FOREX is a crucial concept for conjuring volatility from small price changes. A survey from Statista shows that roughly 75% of retail FOREX traders hold trading positions between 1 hour to 1 week (Statista research department, 2016), emphasizing the rate at which trades are taken .

As popular currency pairs consist of currencies from first world nations, the exchange rate fluctuates consistently around said rate, essentially deeming short term FOREX trading as a zero-sum game. A zero-sum game is when the expected profit of randomly placing trades, should equate to 0, given that the costs of trading from a broker are neglected.

The ratio of pips gained and pips traded for a period of time, such as a year, will be referred to as the “pip ratio”. The pip ratio ranges from 1 to -1. This metric will be used frequently to measure performance in the results and discussion section of this paper. Sharing similarities with the expected value of profit, a pip ratio of 0 is the expected value of randomly placing trades if the spread and cost of trading are neglected.

Synthesized high market volatility, leveraged trading, high market liquidity, and a passive thought of the law of large numbers is the foundation of interest for this particular market. The act of creating an algorithm for decision making that consistently maintains a high pip ratio while placing multiple trades on a weekly basis, could be considered a free lunch. Naturally, a free lunch is a bold statement, and risks are associated with trading on the financial markets. The information in this paper is not considered financial advice, as this bachelor project falls within the domain of data science and supervised machine learning, not economics.

4.3 Trading strategies

Although not included in the trading simulations discussed later in this paper, trading strategies are of high importance for the performance of trading algorithms.

Traders or algorithms placing short term positions often do so with a stop-loss. Stop losses act as a lower bound for a positions price development, which will close a position if the price decreases to a certain level, with the intent of minimizing losses. Stop-losses may also be trailing, called trailing stop-loss. This measure works by automatically defining a new stop loss level after the price of a position increases a certain amount. This measure intends to lock in profits on a position before the market moves in an unfavourable direction.

In terms of algorithmic trading, a well-known approach is the countertrend strategy. This strategy utilizes the signals of an algorithm predicting price movements. An opposite predicted price movement of a currently held position results in the closure of said position, along with the opening of a reverse position. Figure 4 illustrates that the algorithm places its first position at arrow 1, which is a short position. At arrow 2, the algorithm detects a buy signal, leading the algorithm to close the short position, followed by a near-instant placement of a long position. The long position is held until an opposing signal occurs. Arrow 3 and 4 repeat the same events described prior. This example is naturally a very optimistic outcome of the predictive capability of a trading algorithm, which is not expected to be seen during the following experiments.

Regardless of expectations, algorithmic trading in practice is presumably, applied using trading strategies, risk management, a predictive algorithm for decision making, and a deep fundamental understanding of economics.

Figure 4: Example of near perfect countertrend strategy execution.

5 Data

The following subsections will be elaborate on the format and source of the original data. Additionally, any terms specific for this type of data deemed necessary to explained will be explained.

Technical indicators are statistical measures of historical data for a given currency pair with a defined window of time limiting the data included for calculating the indicator value. Technical indicators have a specific value for a given time, similarly to OHLC values.

Descriptions of each technical indicator are provided below. The formulas for calculating the values of the technical indicators used for this project are mimicked as closely as possible to the standard definitions found on trading platforms and sources. The measures mentioned are done with the research question in mind, aiming to uncover whether technical indicators are informative for predicting FOREX market volatility.

The data created by the indicators will be referred to as columns of indicators or indicator data, while indicator data after undergoing feature engineering and scaling will be referred to as features.

Design choices for creating the targets and features are explained. An analysis of the feature's correlation and importance is found in the final subsection.

The data used for this project range from 2012 to 2020, using three different currency pairs, with the EURUSD as the primary currency pair as it accounts for 24% of the FOREX markets transactions (Triennial Central Bank Survey). The two other currency pairs used for the results in the project are the EURGBP and GBPUSD.

5.1 FOREX data representation (OHLC)

Visualizing market data is often done by candle charts, which consists of aggregating a sequence of prices for a defined period, such as 10 minutes. A single aggregation is called a "candle", where a candle consists of 4 data points, namely Open, High, Low, Close. The aggregation is determined by a timeframe, where the first price within the timeframe corresponds to the Open value, the highest price corresponds to the High value, the lowest price to the Low value, and the last price within the timeframe corresponds to the Close value. When discussing prices, there is technically both a bid and an asked price. The source of data used for this project used bid prices.

In figure 5, the x-axis represents the dimension of time, while the y axis represents the exchange rate of a currency pair. The colour of the candles dictates the placement of the Open and Close price. On red candles, the Open price is the top part of the box, and the Close price is the bottom part of the box, meaning that within that time frame, the price fell. These are also called bearish candles. For green candles, the opposite applies, meaning that the Open price is the bottom of the candle, and the Close price is the top of the candle. These candles are also called bullish candles. Figure 6 is of the EURUSD during the evening (GMT+1) 8th December 2020. The six arrows found in the figure define points of the OHLC data.

1. Open on a bearish (red) candle
2. Close on a bearish (red) candle
3. Open on a bullish (green) candle
4. Close on a bullish (green) candle
5. Low on candle
6. High on candle

Figure 5: Description of candle charts

5.2 Data source

A group of traders and analysts have set up a website called histdata.com, hosting an open-source archive of historical FOREX data. 1-minute timeframe OHLC formatted data for 66 different currency pairs is openly available, using EST time zone.

The website has a disclaimer stating, "*since the data is freely available, we will not provide any kind of warranty or certification. Use the data at your own will and risk*".

Attached within the zip folder containing a year worth of OHLC data is a text file with all market gaps in the dataset, which are larger than 1 minute that the provider could not document. A gap is defined as a period where market prices are unavailable, resulting in some time stamps not being present.

When comparing samples from the dataset with pricing data provided from Saxo Banks platform (SaxoTraderPro), it is noticeable that the Open and Close prices are shifted by a small magnitude (between 1-5 pips); however, the times of volatility in price is consistent with that of the broker's data. Time zones are taken into consideration during this comparison.

The price difference is most likely caused by the following factors:

1. A delay in recording/reporting the price from the providers relative to each other.
2. The foreign exchange market is decentralized; thus, different data providers/brokers charge different exchange rates (similarly to how physical FOREX dealers and ATM's charge "high" exchange rates).

Table 1 shows the structure of the raw data. Unfortunately, histdata.com require users to purchase data of market volatility. The OHLC data is suitable for this project as the technical indicators used, only rely on historical OHLC data.

Date Time	Open	High	Low	Close	Volatility
2020-05-07 13:55	1,082670000	1,082680000	1,082550000	1,082550000	,0000
2020-05-07 13:56	1,082550000	1,08260000	1,082430000	1,082510000	,0000
2020-05-07 13:57	1,08250000	1,082940000	1,082410000	1,08290000	,0000
2020-05-07 13:58	1,082920000	1,082980000	1,082910000	1,082980000	,0000
2020-05-07 13:59	1,082980000	1,083130000	1,082830000	1,083070000	,0000
2020-05-07 14:00	1,083080000	1,083330000	1,083050000	1,083330000	,0000
2020-05-07 14:01	1,083340000	1,083340000	1,083030000	1,083270000	,0000
2020-05-07 14:02	1,083270000	1,083330000	1,082940000	1,082940000	,0000

Table 1: Raw, uncleaned CSV data, histdata.com

5.3 Data cleaning

Due to known gaps in the data mentioned in the prior subsection, it was decided to fill the missing entries. Adding the data was done by using the previous Close price to define the missing Open price. Likewise, the Close price was determined by the Opening price of the next available data entry. The High and the Low of the missing entry, was determined by the maximum and minimum of the mentioned Open and Close price.

Most time gaps found throughout the entire dataset ranged between 1-3 minutes, with very few exceptions of longer periods. During the cleaning, gaps of time greater than 30 minutes were not extrapolated, as it was deemed inappropriate to synthesize the mentioned amount of data.

For time gaps greater than 1 minute, the first OHLC instance was defined as mentioned above, however, the following entries were defined by the next available Open price, leading to a non-volitile period.

Table 2 displays a sbusection of the statistics of newly added rows.

The resources available in terms of computational power relative to the dataset and task, it was decided to aggregate the OHLC data from 1-minute entries to 10-minute entries, reducing all future computations by an order of magnitude.

```
Statistics for 2015
2015 EURUSD Added rows total: 585  as a percentage: 0.15716976214975995%
2015 EURGBP Added rows total: 2124  as a percentage: 0.5730210674270977%
2015 GBPUSD Added rows total: 474  as a percentage: 0.12734233326616964%

Statistics for 2016
2016 EURUSD Added rows total: 718  as a percentage: 0.19265961500276377%
2016 EURGBP Added rows total: 1681  as a percentage: 0.45224643529728276%
2016 GBPUSD Added rows total: 945  as a percentage: 0.2537055780325871%

Statistics for 2017
2017 EURUSD Added rows total: 783  as a percentage: 0.21069116388705017%
2017 EURGBP Added rows total: 2159  as a percentage: 0.5831163639899312%
2017 GBPUSD Added rows total: 1213  as a percentage: 0.3267753762765926%

Statistics for 2018
2018 EURUSD Added rows total: 951  as a percentage: 0.2552293843899454%
2018 EURGBP Added rows total: 4033  as a percentage: 1.091820684763511%
2018 GBPUSD Added rows total: 1235  as a percentage: 0.33175648268284885%

Statistics for 2019
2019 EURUSD Added rows total: 233  as a percentage: 0.06254546625900266%
2019 EURGBP Added rows total: 472  as a percentage: 0.12682921586225057%
2019 GBPUSD Added rows total: 292  as a percentage: 0.07841157483740 34%

Statistics for 2020
2020 EURUSD Added rows total: 1494  as a percentage: 0.40125263875982314%
2020 EURGBP Added rows total: 1157  as a percentage: 0.3106248741526277%
2020 GBPUSD Added rows total: 647  as a percentage: 0.17343240693086293%
```

Table 2: percentages of rows added to the cleaned dataset

5.4 Exploring price fluctuations in original OHLC 1-minute data

An interesting point to explore is the validity of the dataset. Section 5.1 explains how OHLC data is represented on a candle chart. For any given candle, the open price is supposed to match the previous close price. In both the datasets and the visualizations of candle charts, this statement is true, with possibilities of slight price variations of 1 to 2 pips.

As a method of checking the consistency of the dataset, an analysis was performed with the objective of validating the consistency of the original 1-minute OHLC data for the EURUSD. Additionaly, the same analysis is performed on the cleaned 10-minute OHLC data. The definition of price fluctuations are as followed:

1. For any given close price, a price change of 10 pips relative to the previous close price is considered a price fluctuation.
2. If the previous close price is not within 10 pips of the current open price, the entry is considered inconsistent, as there is a price gap.
3. The first entry of each week is considered a consistent price fluctuation, to avoid classifying a change in price occurring after a weekend

Original 1-minute EURUSD 2012-2019	Consistent Fluctuations	Inconsistent Fluctuations
Size of dataset	2970910 rows	
Number of fluctuations	**5727**	**12**
Mean change in close price	14.9 pips	27.1 pips
Standard deviation of change in close price	6.9 pips	23 pips
Minimum change in close price	10 pips	11.9 pips
Maximum change in close price	74.7 pips	92.4 pips

Original 1-minute EURUSD 2020	Consistent Fluctuations	Inconsistent Fluctuations
Size of dataset	372335 rows	
Number of fluctuations	**588**	**0**
Mean change in close price	12.9 pips	0 pips
Standard deviation of change in close price	7.4 pips	0 pips
Minimum change in close price	10 pips	0 pips
Maximum change in close price	0 pips	0 pips

Cleaned 10-minute EURUSD 2012-2019	Consistent Fluctuations	Inconsistent Fluctuations
Size of dataset	298126 rows	
Number of fluctuations	**14417**	**5**
Mean change in close price	15.4 pips	50.6 pips
Standard deviation of change in close price	7.4 pips	56 pips
Minimum change in close price	10 pips	11.3 pips
Maximum change in close price	76.9 pips	147.6 pips

Cleaned 10-minute EURUSD 2020	Consistent Fluctuations	Inconsistent Fluctuations
Size of dataset	37377 rows	
Number of fluctuations	**1919**	**0**
Mean change in close price	15.1 pips	0 pips
Standard deviation of change in close price	4.9 pips	0 pips
Minimum change in close price	10.2 pips	0 pips
Maximum change in close price	34.3 pips	0 pips

The tables above state that 5 out of almost 300,000 entries within the cleaned 10-minute training dataset are considered inconsistent. Looking further for an explanation of the observed phenomenon, an article on Investopedia explains this occurrence as "gapping" (https://www.investopedia.com/terms/g/gapping.asp). Gaps found during the opening of the market after a weekend or holiday is known as common gapping, which in this case is considered a consistent fluctuation, while the gaps considered inconsistent fluctuations are either called "runaway gaps", "exhaustion gaps", or "breakaway gaps", where a noticeable gap exists between the new open price and the previous close price. An example of the gap types is shown in figure 6 to the right.

Figure 6: Breakaway gap and runaway gap from Investopedia article

Although the definitions and labelling of the mentioned gaps seem slightly subjective, their existence is defined and are therefore presumably well known among technical analysts. Removing these outliers from the dataset will not be done, as these presumable outliers could potentially be part of FOREX market data. As the quality of the data does not appear to be of concern, the results of the experiments will enable conclusions of the research question.

5.5 Targets for machine learning models

The research question for this paper aims to clarify whether technical indicators are informative for FOREX market volatility. As initial consideration was using the future close price as the target, which was quickly discarded as the value of the future close price is not informative for determining price volatility on its own. To numerically describe the magnitude and directionality of the difference in close prices inbetween rows, the percentage difference of a previous close price is a viable option. A similar measure mentioned in section 4.1, the standard unit of measuring price differences in the FOREX market is a pip.

The two images display the close price difference as a percentage and the price difference in terms of pips. The two figures are of the GBPUSD (left) and the EURUSD (right). The measuring tool from SaxoTraderPRO confirms the definition of a pip defined in section 4.1. A price change of -5285 pips for the GBPUSD is the difference between the initial price of 1.98406 and the future price of 1.45560, multiplied by 10000. This understanding is further validated with the EURUSD, with a price difference of 1.38645 and 1.04617 being -3403 pips. The price differences are this large, as each candle represents OHLC values for an entire month, rather than 10-minutes, as they do in the rest of this project.

Figure 7: GBPUSD: Difference in price

Figure 8: EURUSD: Difference in price

The targets defined for the machine learning experiments in this project will be the future change in pips, as a value to be found using regression. As the optimal amount of time in the future to predict for is an unknown variable, multiple target datasets for each currency pair for each year have been created, with the said variable being referred to as "future candles". The values for future candles will be set as 3, 6, 12 and 18. The expectation for the optimal value for future candles is that it will be a low value, as it seems more plausible to predict. On the contrary, larger future candle values may allow for a better overall accumulation of pips traded in a year during the trading simulations, which will be described further in section 7.4.

	Future Candle 3	Future Candle 6	Future Candle 12	Future Candle 18
Mean	-0.020056	-0.040039	-0.079861	-0.117923
Std	5.298691	7.494573	10.503823	12.719829
Maximum	69.2	100.9	122.1	130
75th percentile	2.2	3.1	4.4	5.4
Median	0.0	0.0	0.1	0.1
25th percentile	-2.2	-3.2	-4.5	-5.6
Minimum	-76.5	-88.7	-89.2	-95.7

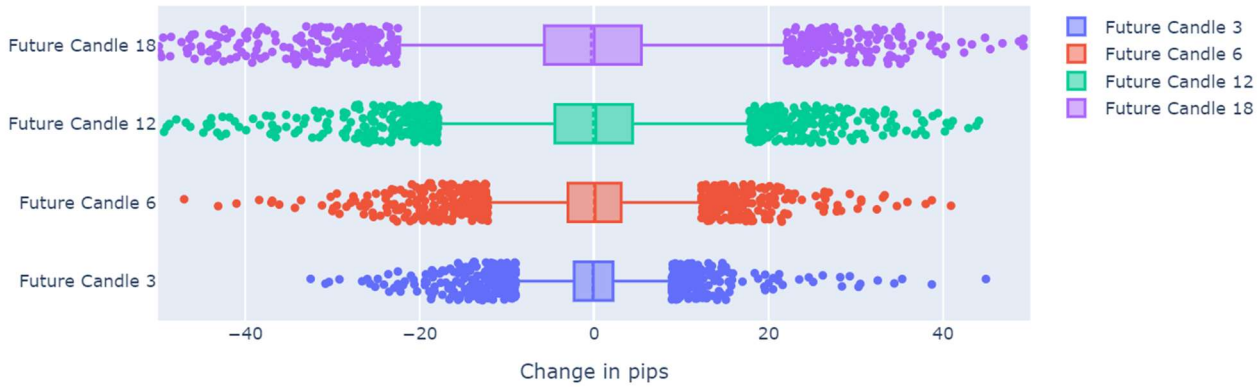


Figure 9: Box Plot of 5000 samples of target values, from EURUSD 2012 - 2020

The table above displays statistics for the different targets for 2012 to 2020 for the EURUSD. Below the table, figure 9 shows a randomly sampled subset of the same data, displayed as a box plot. The visualization shows the 1st and 99th percentile as the end of the whiskers, the Median, 25th, and the 75th percentile as lines divided the box, and the dotted line representing the mean. The data points on each side of the boxplot highlight outliers of the sample (the data points). The outliers of the dataset are the points of interest for this particular task, defining price future fluctuations.

To further elaborate on the format of the target dataset, let the value of future candles be set to 12 for the following visualizations. As shown in figure 10 below, the distribution of target values for

the EURUSD 2019, also follows a normal distirbution, nearly identical to that of the remaing dataset.

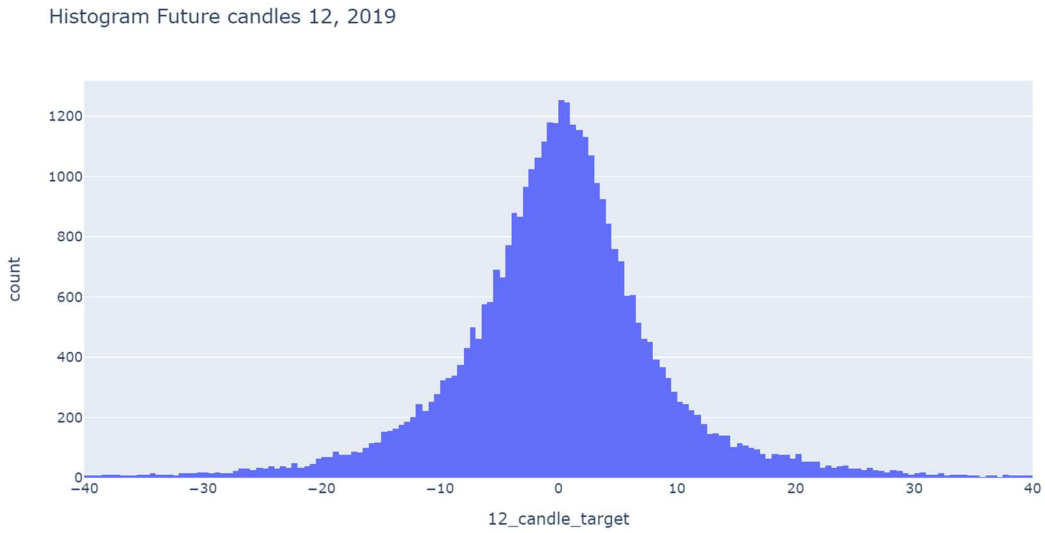


Figure 10: Histrogram of future candles 12, 2019

Figure 11 shows the targets values with their respective timestamp on a line chart. The bridge-like connections show gaps of data during the weekends, which is due to the market being open, 24 hours a day, 5.5 days a week.

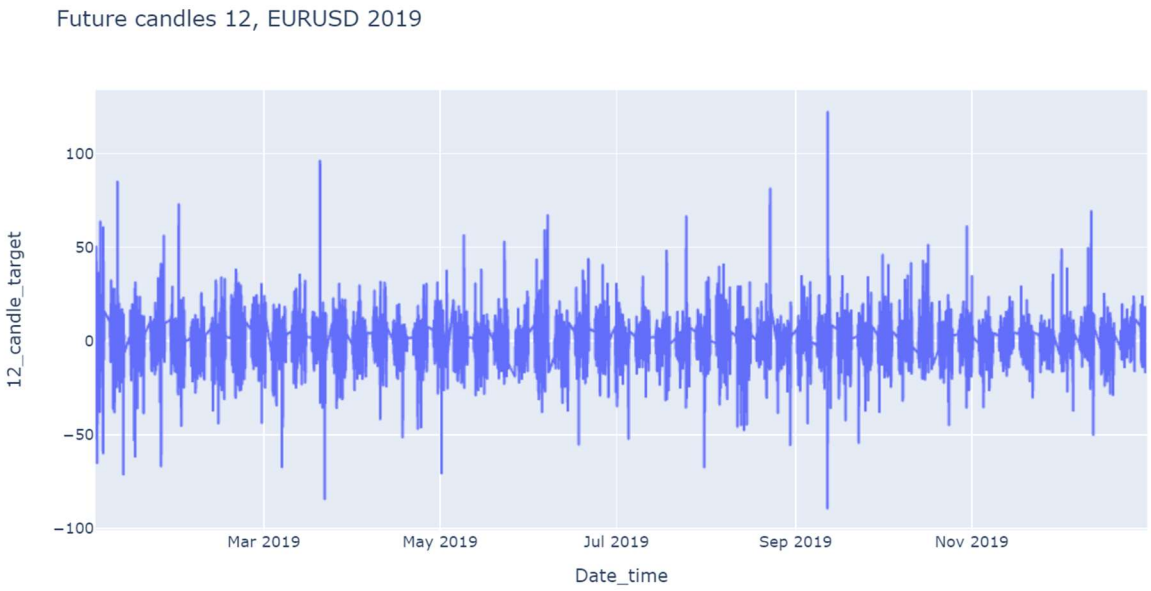


Figure 11: Line chart of future candles 12, 2019

Figure 12 displays the same target data for an arbitrary week of 2019, intending to visually enhance the price fluctuations. The date and time denotes the value of the future difference in price. For example, the label for the date and time 7th October, 06:00, has a value of 21.5, meaning that the price will increase by 21.5 pips from 06:00 to 08:00.

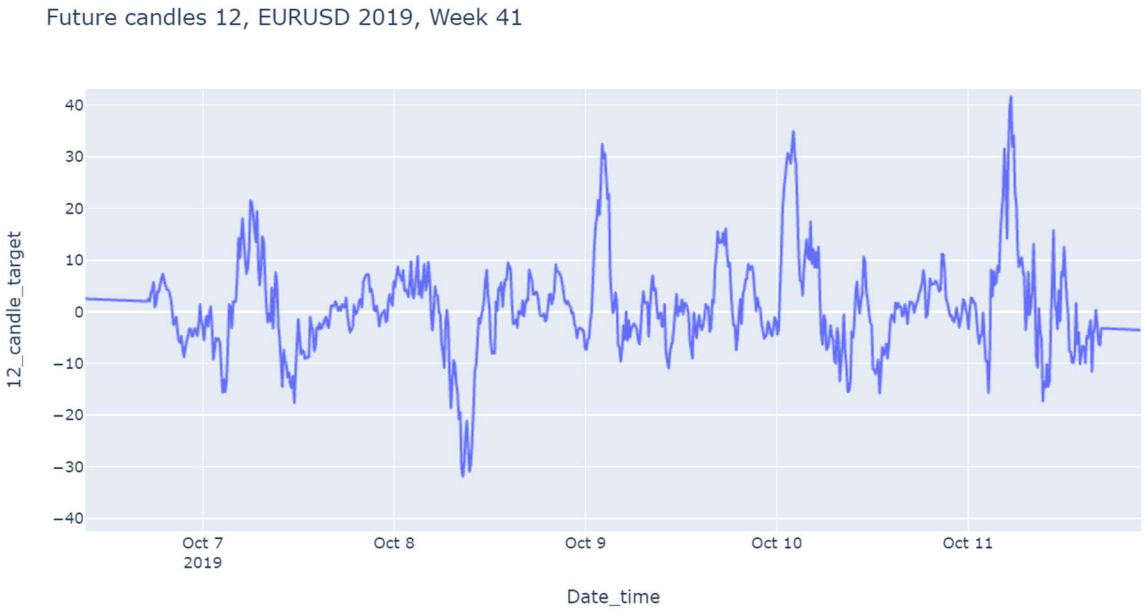


Figure 12: : Line chart of future candles 12, 2019, week 41

5.6 Technical indicators

5.6.1. What is technical analysis?

When financial analysts perform technical analysis, they view candle charts of OHLC data with indicators attached to their trading view, as displayed in figure 13. Technical analysts are notorious for drawing support and trend lines on top of their trading view to determine trading decisions. Whether or not the practice of drawing support lines is valid, the dataset created for this research will not include said methods. Drawing support line is primarily avoided due to subjectivity in extrapolating information by drawing lines. Furthermore, instructions on how to initiate trend lines were found to be somewhat subjective. Past data is objective, and technical indicators are defined by mathematical formulas derived from past data; hence the values are objectively defined.

The technical indicators used in this research will be defined in the latter subsubsections. Technical analysis is by far the most controversial topic when determining market price predictions, where most analysts' opinions are divided into two categories. The practice is either considered a pseudoscience or a valid strategy for decision making.

Technical analysis is defined by Investopedia as: "*... a trading discipline employed to evaluate investments and identify trading opportunities by analyzing statistical trends gathered from trading activity, such as price movement and volume*". (source)

Figure 13: Trading view of all indicators included for the data set viewed on SaxoTraderPRO

5.6.2. Technical indicator specifications

The trading view shown in figure 13 is a representation of what the trading view could look like if the technical indicators for this project were used in the same chart. As there are multiple parameters in some of the indicators, the default parameters will be used, except for the time windows.

Multiple time windows for the same indicator will be defined, as it is assumed to be a debatable parameter amongst analysts due to it being a changeable parameter on most trading platforms.

The time windows used are; 60, 90, 120, 240, 360, 480, and 600 minutes, which will be denoted with the notation n in the following subsections. $n=\{6,9,12,24,36,48,60\}$. In total, this results in 98 unique columns of technical indicators, which will be used as the input data for the machine learning experiments. Some indicators have multiple columns for each value of n, which is mentioned in each indicator's description.

The notation for the formulas for each indicator has been altered slightly, letting the notation variables share the same meaning throughout. Some technical indicators include other indicators embedded within, such as the MACD, which uses the EMA. In such case, embedded indicators will be referred to by their abbreviation, such as the simple moving average being called SMA.

Some technical indicators require $k \geq n$, which happens for the first n, OHLC values for each year. A description of handling the lack of initial OHLC values for each indicator will be mentioned in their description. The index of the first value in the dataset is 1, when referred to in mathematical notation. The fixed notation variables are:

x: the close price.
t: the point of time for the value it is subscripted.
n: the time window used when the indicator is calculated.
i: the index for an element in the sum.

5.6.3. Simple Moving Average (SMA)

The simple moving average (SMA) for any given time t is the mean of all the past n closing prices, where x is the closing price. The value of k is set to 1 if $k < 1$.

$$SMA_t = \frac{\sum_{i=t-n}^{t} x_i}{n}$$

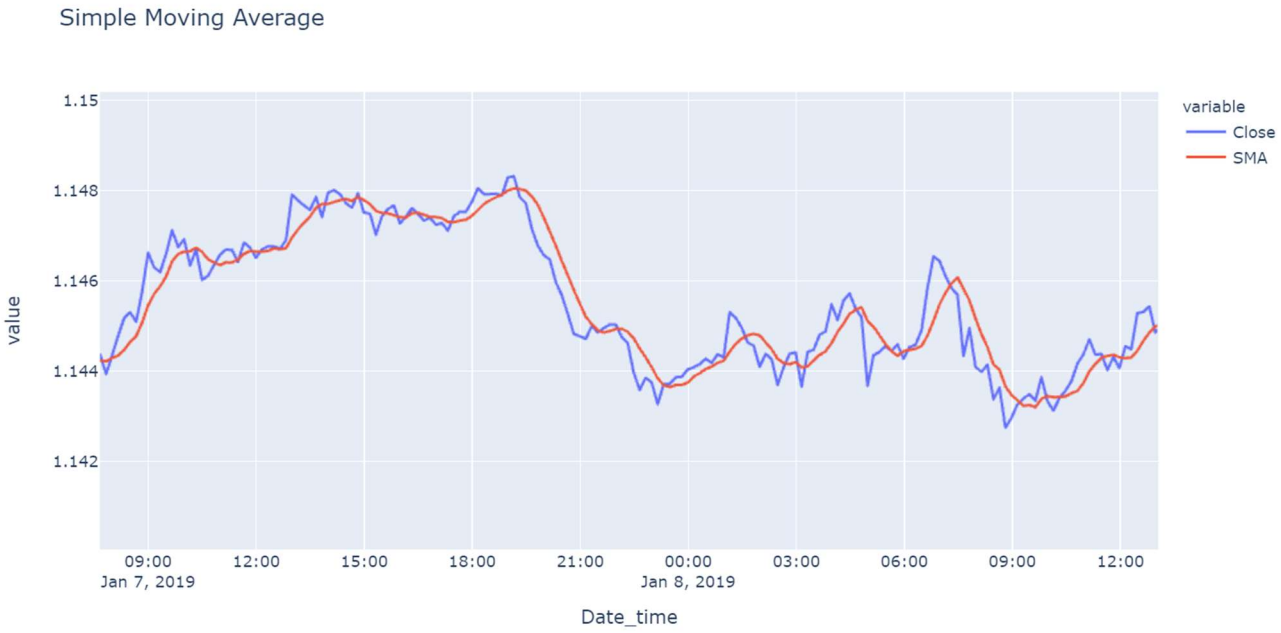


Figure 14: Simple Moving Average

5.6.4. High & Low

Price ceilings and floors commonly drawn on trading views. The feature "High" represents the maximum value found in the OHLC data for the given time window, and the "Low" represents the minimum value found in the OHLC data for the given time window.

$$High = max(x_{t-n}, x_{t-(n-1)}, x_{t-(n-2)}, \ldots, , x_t)$$
$$Low = min(x_{t-n}, x_{t-(n-1)}, x_{t-(n-2)}, \ldots, , x_t)$$

5.6.5. Bollinger bands (BB)

The Bollinger bands (BB) indicator uses the standard deviation of the difference between the current closing price, and the SMA. The indicator was created by John Bollinger in the early 1980s to be an indicator measuring volatility. In terms of the dataset, two features are created for each value of n: BB_upper and BB_lower, where d=2. The initial value of i is set to 1 if $i < 1$.

$$\sigma_t = \sqrt[2]{\frac{\sum_{i=t-n}^{t}(x_i - SMA_i)^2}{n}}$$

$$BB_t = x_t \pm \sigma_t \cdot d$$

Figure 15: Bollinger bands

5.6.6. Exponential Moving Average (EMA)

The Exponential Moving Average (EMA) is a weighted version of the SMA, which considers the current difference in price and the previous EMA value multiplied by the weight. The indicator intends to let the current price have a heavier weight on the indicator value. The EMA is only calculable after the first n values in the data. The first n values for EMA is calculated using SMA.

$$\alpha = \frac{2}{n+1}$$

$$EMA_t = EMA_{t-1} + \alpha \cdot (x_t - EMA_{t-1})$$

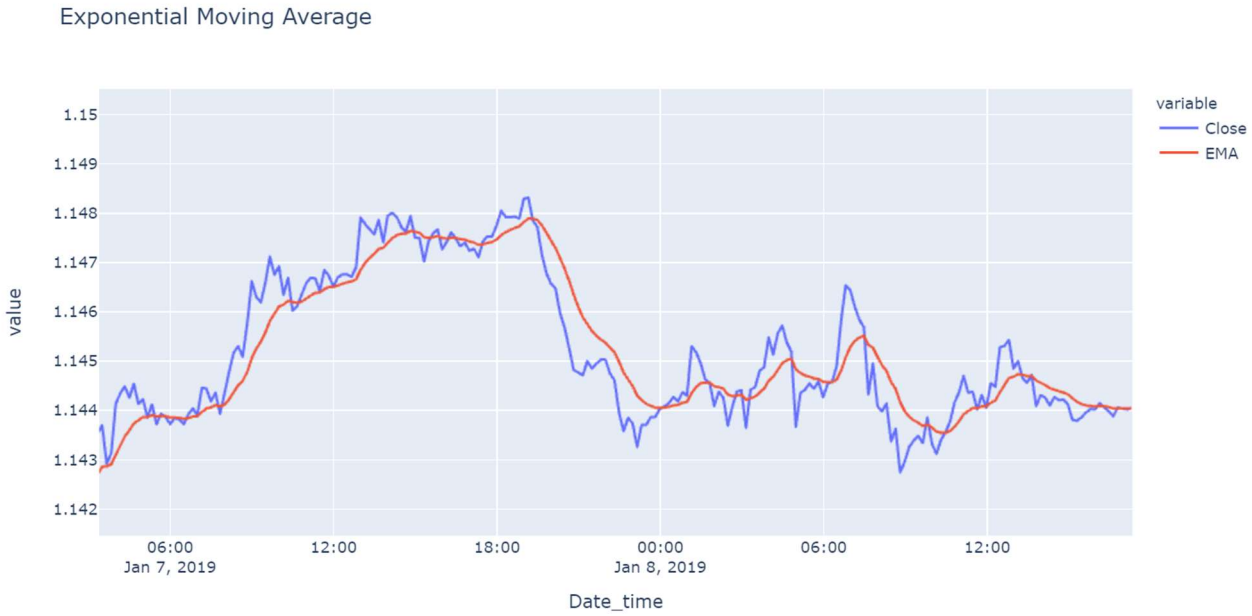


Figure 16: Exponential Moving Average

5.6.7. Moving Average Convergence Divergence (MACD)

The MACD, an abbreviation for the Moving Average Convergence Divergence which consists of three parts. The MACD, MACD Signal Line, and the MACD Histogram. As stated in the formula, the MACD is calculated by subtracting two EMA lines, having a time window of 26 (slow EMA) and 12 (fast EMA). The signal line is calculated by the EMA of the MACD line using a time window of 9. As the variable n denotes the time window, the default time windows for the MACD (26,12,9) will be multiplied by a fraction, such that their relative difference remains intact:

26: n
12: $n \cdot 12/26$
9: $n \cdot 9/26$

$$\alpha = \frac{2}{n+1}$$

$$\beta = \frac{2}{\lfloor n \cdot \frac{12}{26} \rfloor + 1}$$

$$slow\_EMA_t = EMA_{t-1} + \alpha \cdot (x_t - EMA_{t-1})$$
$$fast\_EMA_t = EMA_{t-1} + \beta \cdot (x_t - EMA_{t-1})$$
$$MACD_t = fast\_EMA_t - slow\_EMA_t$$

$$\gamma = \frac{2}{\lfloor n \cdot \frac{9}{26} \rfloor + 1}$$

$$MACD\_signal_t = MACD\_signal_{t-1} + \gamma \cdot (MACD_t - MACD\_signal_{t-1})$$
$$MACD\_histogram_t = MACD_t - MACD\_signal_t$$

The original MACD Histogram was not included as a feature. Instead, an altered version of the histogram was created called the MACD crossover. The MACD crossover is defined as the difference between the MACD and the MACD signal. Values are multiplied by 5 if two conditions are met. Firstly, the MACD line must have crossed the "zero-line", that being, the sign of the previous MACD value, differing from the current. Secondly, the value of the MACD crossover must be greater than the standard deviation of all values of the MACD crossover prior to the scaling. This is proposed to emphasize the value as a potential signal.

Three columns of data are generated for each value n named:
MACD, MACD_signal, MACD_crossover.

All three values oscillate around 0, with a standard deviation of 0.000834 for the EURUSD during the years 2009 to 2019.

The MACD, MACD signal line, and the MACD crossover are visualized on the next page.

Figure 17: MACD vs. MACD signal line

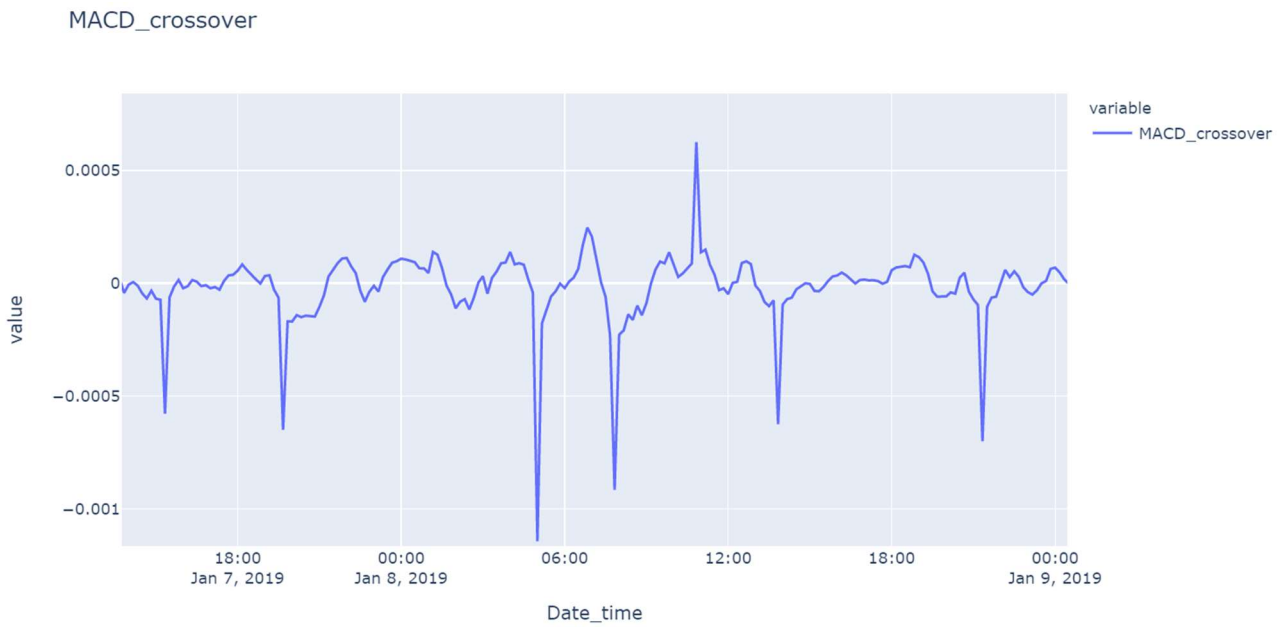


Figure 18: MACD crossover

5.6.8. Rate of change (ROC)

The rate of change is used in a variety of different fields and applications, including the field of finance as a technical indicator. The value is a percentage determining the rate of change of the current close price of t, and the close price at t-n. When t-$n < 1$, the ROC value expression is set to 0. In theory, the rate of change has a lower bound of -100 and an infinite upper bound.

The ROC values for the EURUSD during 2012 to 2019 fluctuates around 0, with a standard deviation of 0.334.

$$ROC_t = \frac{x_t - x_{t-n}}{x_{t-n}} \cdot 100$$

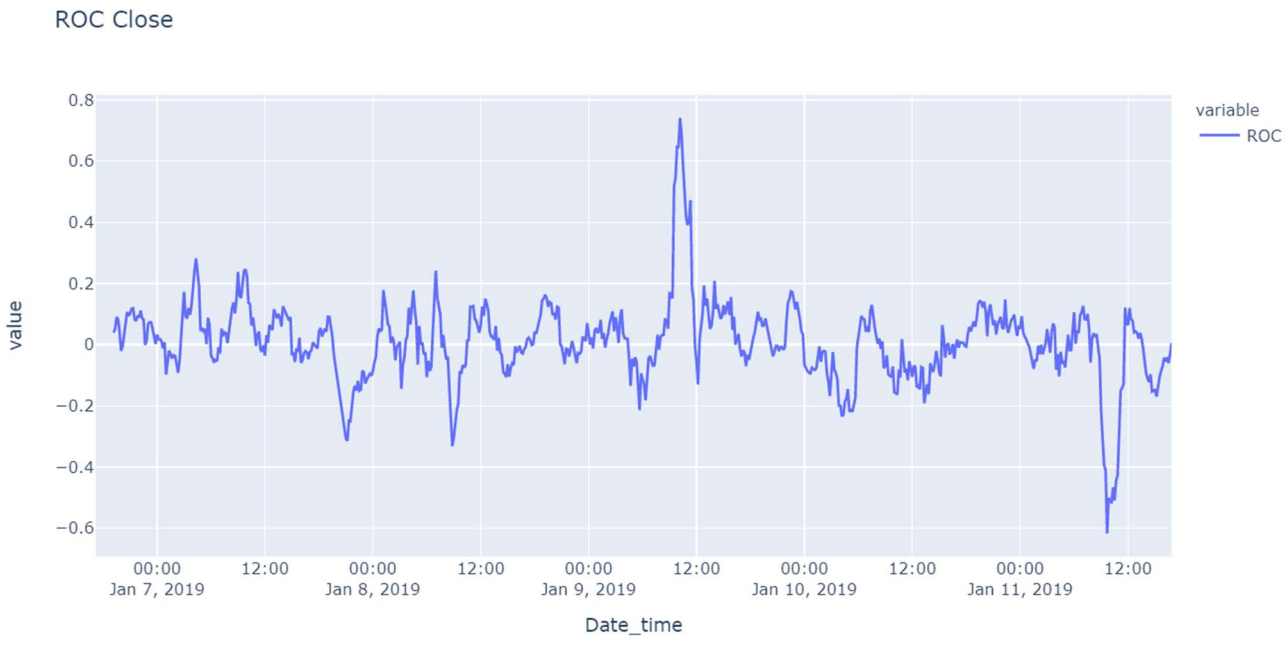


Figure 19: ROC

5.6.9. Stochastic Oscillator (STCO)

The Stochastic Oscillator is a measure of volatility relative to the defined time window. This indicator is calculated using the current close price, the High in the time window, and the Low in the time window. STCO oscillates between 0 and 1. The STCO has a value of 1 when the close price is equal to the High in the time window and a minimum value of 0 when the close price is equal to the Low in the time window.

$$STCO_t = \frac{x_t - L_n}{H_n - L_n}$$

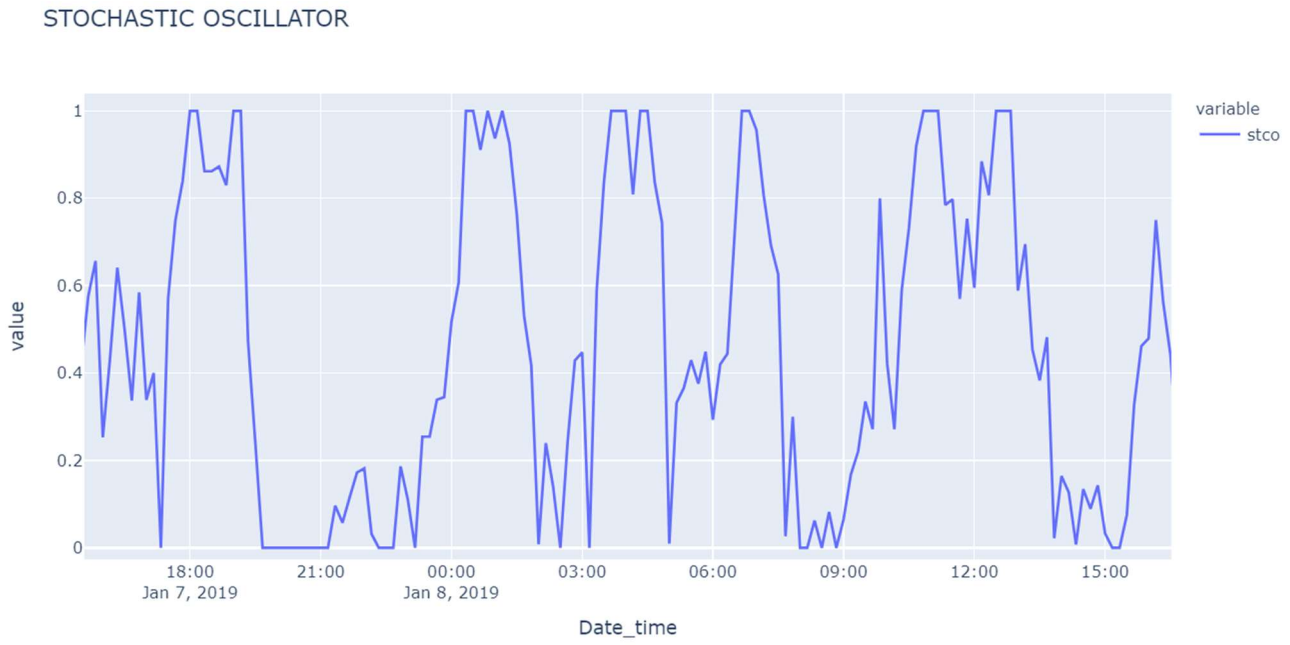


Figure 20: Stochastic Oscillator

5.6.10. Relative Strength Index (RSI)

The relative strength index is calculated in multiple steps. Firstly, two variables, denoted U and D, are calculated for all close prices in the dataset. The two variables consist of the absolute value of the difference in close prices at t and t-n.

$$U_t = \begin{cases} x_t - x_{t-n}, \text{ if } x_t > x_{t-n} \\ 0, \text{ if } x_t \leq x_{t-n} \end{cases}$$

$$D_t = \begin{cases} x_{t-n} - x_t, \text{ if } x_t < x_{t-n} \\ 0, \text{ if } x_t \geq x_{t-n} \end{cases}$$

When t-$n < 1$, the value of that expression is set to 1. The mean of U and D are calculated and used in the formula for RSI shown to the right.

$$\bar{U}_t = \frac{\sum_{i=t-n}^{t} U_i}{n}$$

$$\bar{D}_t = \frac{\sum_{i=t-n}^{t} D_i}{n}$$

The relative strength index oscillates between 0 and 1, where a value close to 1 indicates that the currency pair is overbought, whereas a value close to 0 indicates that the currency pair is oversold. The RSI technically never reaches 0 or 1, as the mean of U and D are set to 0.000001, to avoid division by zero. Although this measure is only necessary for values of t, where the mean of D *is 0*, the configuration has been applied for both U and D.

$$RSI_t = 1 - 1 \cdot \frac{1}{1 + \frac{\bar{U}_t}{\bar{D}_t}}$$

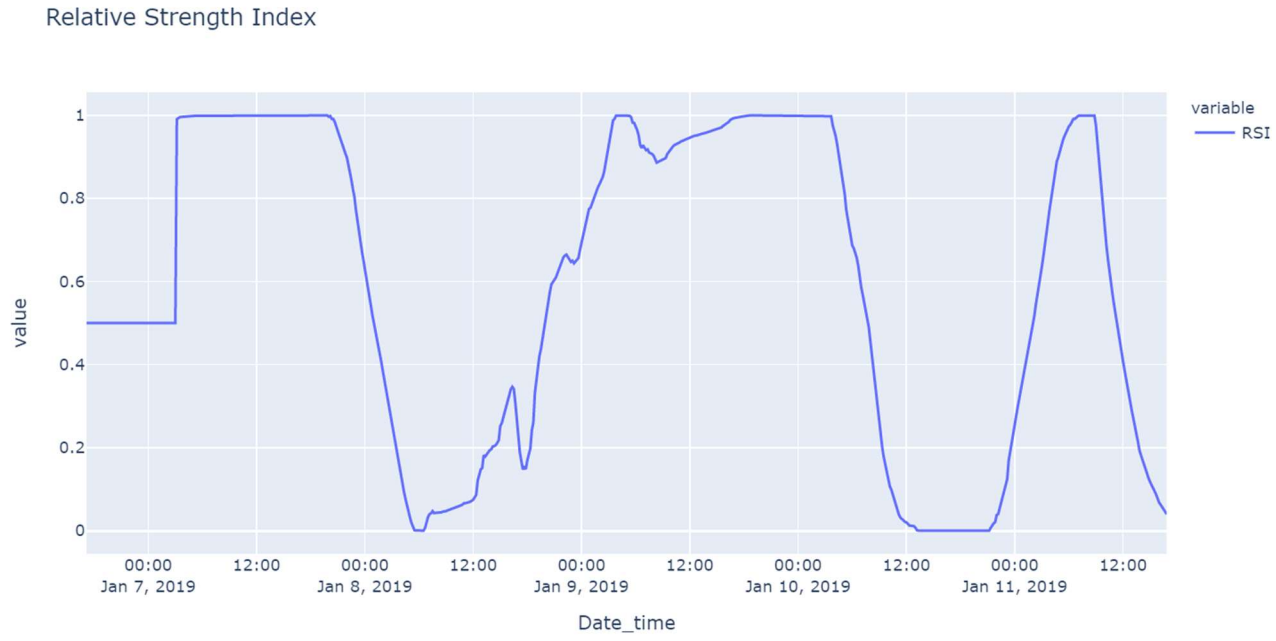


Figure 21: Relative Strength Index

5.6.11. Aroon indicator (AROON)

The Aroon indicator consists of two indicators: AROON_up and AROON_down. NH_t and NL_t refers to the amount of time, since the last High and Low for the given time window n. The indicator applies the variables in the formula such that the value of the indicator has a minimum of 0 and a maximum 1. A value 0 means that the most recent High (AROON_up) or Low (AROON_down) occurred at t-n, while a value of 1 occurs when the High or Low for the time window n is present in the OHLC data at time t.

$$AROON\_up_t = \frac{n - NH_t}{n}$$

$$AROON\_down_t = \frac{n - NL_t}{n}$$

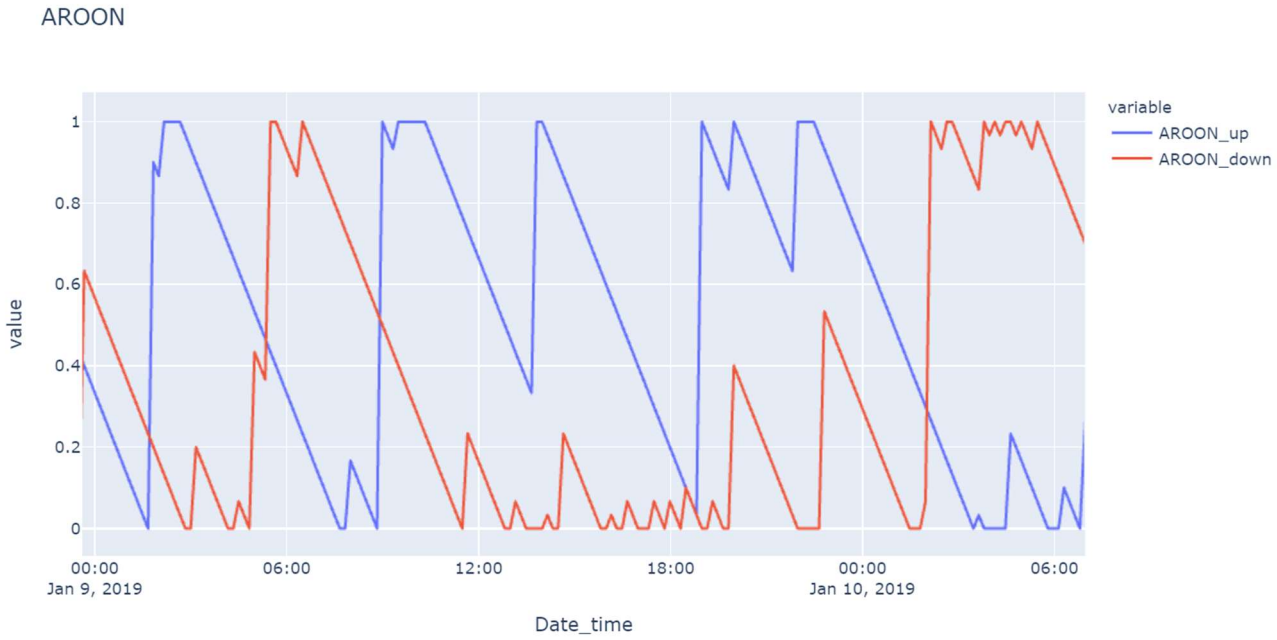


Figure 22: AROON

5.7 Transforming technical indicators to features

5.7.1. Indicators with a value of an exchange rate

Indicators with values that offset a currency pair's current market price, are subtracted with the close price for the respective time t. The decision has been taken as a feature engineering decision to reflect the information of interest, which is the indicator's value relative to the currency pairs market price.

While an analyst analyses a trading view, the relative difference of said indicator to the price is presumably the point of interest, not the absolute value. The measure has not been applied to the formulas or visualizations in the subsections above defining the indicators. This is due to their unaltered format being the intended information to communicate.

Consider the following example: A fictive analyst is fond of the Bollinger band indicator (see figure 23) and uses the indicator for their trading strategy. The analyst's strategy is to execute a long position if the close price hits the upper band (red line) and a short position if the close price hits the lower band (green line), with the intention of closing their position shortly after. Regardless of the questionable strategy, the point of interest is the value of the Bollinger bands relative to the close price. A similar logic applies to the SMA, EMA, BB, High and Low, which also have values that are slight offsets of the current close price.

Figure 23: Candle chart with Bollinger bands

To further argue for subtracting the close price from the value of the mentioned indicators, consider figure 24. The figure displays a candle chart for the EURUSD during 2010 to 2015. The blue line is the Simple Moving Average (SMA). The scalar value of the SMA has seemly little to no correlation with points in time where price fluctuations occur; however, the difference between the SMA and the closing price is likely to be more descriptive for predicting future price fluctuations.

Figure 24: Candle chart of 2010-2015 for the EURUSD with 1-week candles.

This falls under the mentioned controversial statement that future market prices are dependent on past market prices, which is the foundation for the argument of technical indicator's validity for decision making. As the research question aims to discover whether technical indicators are informative for neural networks, the prior statement is assumed to be correct, although it is hypothesized to be false.

Furthermore, when preparing indicator data as features, which will be described in detail in section 5.9, the measure will ensure that any value of an indicator at interest is represented at a point on a normal distribution, which is not affected by more significant changes in the price over a long period.

In other words, the numerical value of an indicator (for example, the SMA) as a feature for an artificial neural network, at any given point in time, is meant to reflect the sentiment of said indicator, which is assumed to be the displacement of the indicator value relative to the closing price at said time, rather than the indicator value relative to the entire set of values of said indicator, from 2012 to 2019.

5.7.2. Indicators represented by a percentage.

The indicators STCO, RSI and AROON, are represented through values oscillating between 0 and 1. As both the maximum and minimum values of these indicators represent some significance, the said significance is to be transformed such that it may be interpreted when used as input data for neural networks. If a feature x_0 has a value of 0, the weights attached to the corresponding input node for that feature in a neural network will be effectively be muted.

The indicators are often used with some estimated threshold, such as 0.75 and 0.25, with little interest mid-value. As representing significant values is of importance, these indicators are scaled with 2 and subtracted by 1, over the entire indicator column. The effect of this is shifting the minimum from 0, to -1, while the maximum remains at 1. The transformation's motivation is that values that prior were close to 0, have significance in predicting the target.

5.8 Training, validation and test data sets

The research paper aims to find a balance between using standard machine learning practices while maintaining the transparency of the experiments and the results. The training data used for the experiments is the range of 2012 to 2019 for the EURUSD, while the test set used is of the same currency pair for the year 2020. During model training, the training set was split and shuffled, leaving 80% to be used for training, and 20% to be used as a validation set. A visual representation of said split can be viewed below in figure (num), where the black proportion refers to validation data, the blue proportion training data, and the grey proportion being the test data.

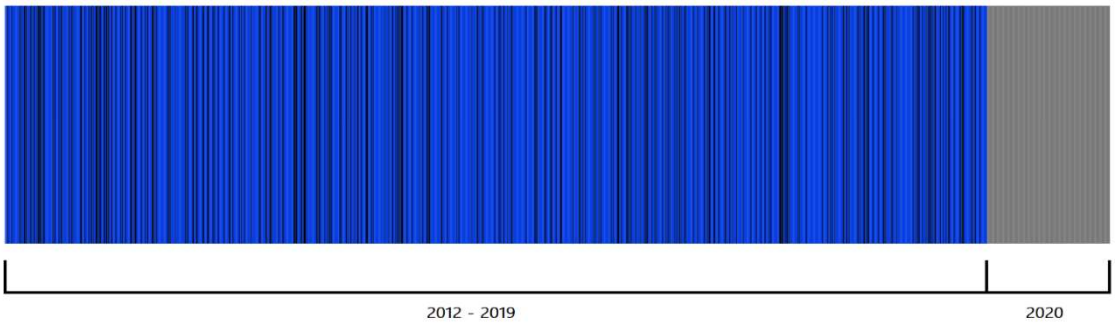


Figure 25: Test, Validation, and Training split for EURUSD

As the feature data for the neural network models is 2-dimensional, as shown in figure 26, a conscious decision was taken when defining the test set, as both features and targets of the test set were secluded from the training set.

The shape of input data of the LSTM and CNN models for this project uses 2D input data (disregarding the CNN's channel parameter as is set to 1). The window size is a parameter described in section 7.3, which refers to the number of entries from the past, included in the 2D input data, as visualized in figure 26.

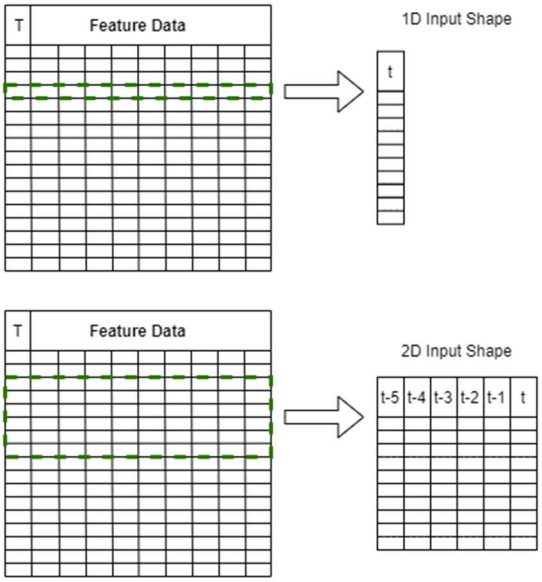


Figure 26: Input data shape

Another motivation for using the year 2020 as a test set was due to the high amount and scale of unprecedented events, drastically impacting economies worldwide. Most countries experienced negative economic growth rates, exceeding those of the financial crisis in 2008. The washing post has labelled this period as the worst year of economic growth since World War 2 (source). A chart of the United states' GDP is shown below in figure 27.

Although the pandemic has undoubtfully caused financial insecurities, personal grief, and struggling daily lives for many individuals, the events of 2020 define an interesting test set, emulating unexpected events in the financial market.

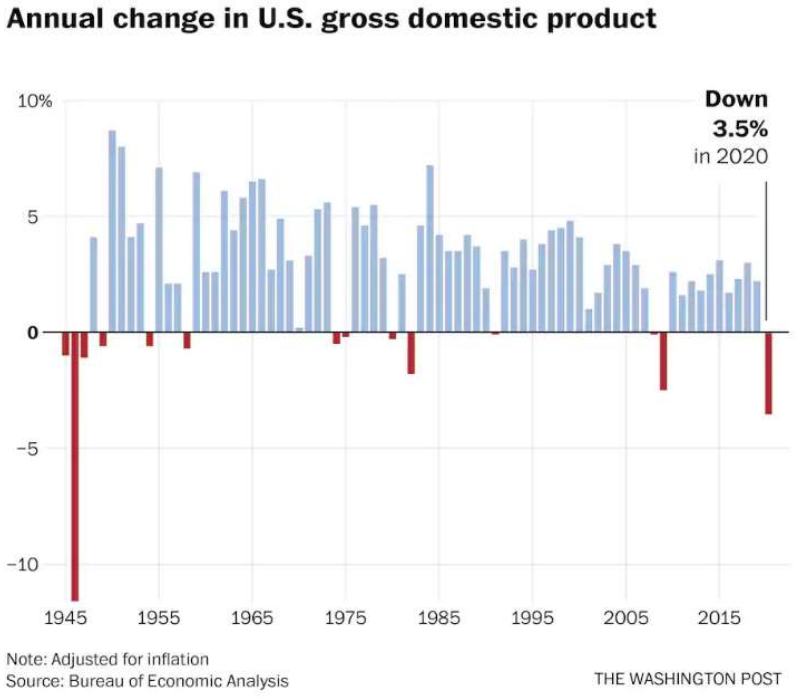


Figure 27: Bar chart of economic growth for each year

5.9 Feature scaling

Artificial neural networks respond best to normalized and scaled input data, which has become standard practice when preparing data for training neural networks. The research conducted by [2014, Jian Jin et al.] implementing normalization methods showed significant results in minimizing loss functions. The best normalization method for minimizing loss function scores for regression was the "*normal distribution normalization method*", more commonly known as z-score normalization.

The transformed indicator data, with values prior to the transformation being represented as an exchange rate, was scaled by a factor, of $\frac{1}{\sigma}$. The standard deviation for each indicator column was calculated using all of the said indicator data for a currency pair between the years 2012 to 2019, excluding the data from 2020. This was done due to 2020 being the test set for the experiments.

The mean of the feature dataset remained untouched, as the sentiment of the value representing 0 for multiple indicators should remain. The distribution of the feature data for the EURUSD 2019 is visualized below in figure 28.

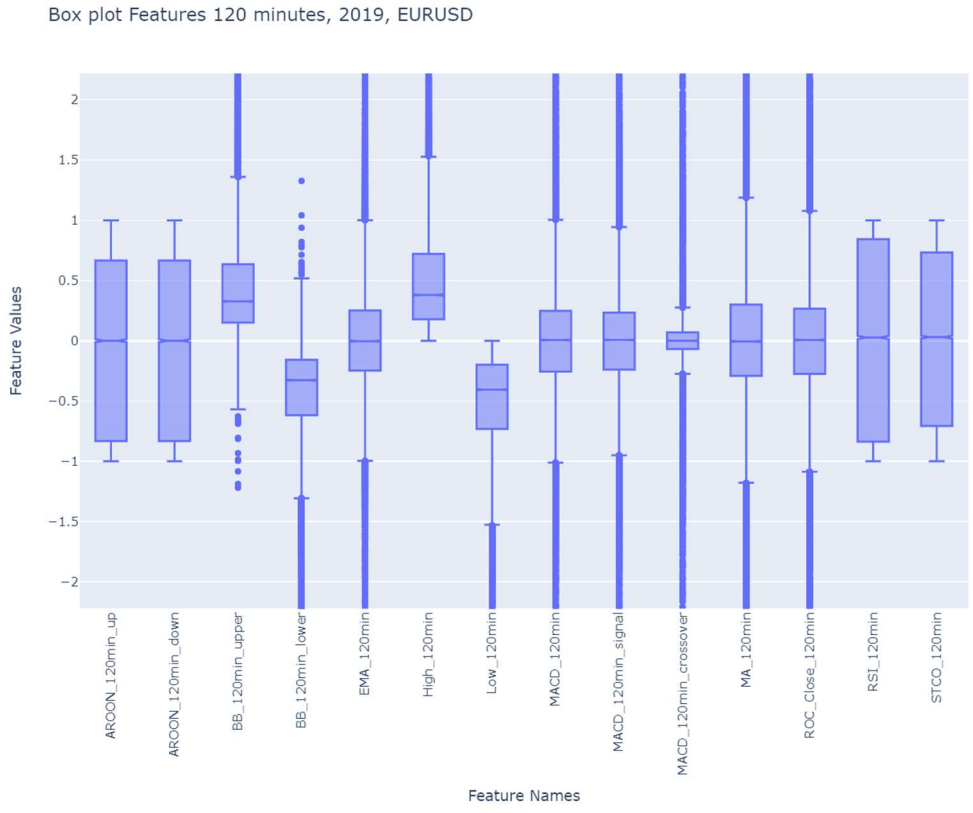


Figure 28: Box plot of feature data

5.10 Feature correlation and importance to target data

With the features defined, an analysis of the feature importance was conducted. The importance of a feature will be defined by two metrics. Firstly, Pearson's correlation coefficient. A measure of a given feature's correlation with the target, ranging from -1 to 1, with strong correlations being less than -0.7 or greater than 0.7.

Secondly, the feature present in the nodes of a trained decision tree. Decision trees are constructed by calculating the Gini-impurity or Gini-importance for each node in the decision tree, starting from the root node. The feature, which is most capable of separating the labels, is used as the root node. This continues for each node's child nodes until a decision has been made, returning a regressed value. The dataset used is the features of the EURUSD ranging from 2012 to 2019.

The correlation values are seen in figure 29, which determine that the correlation of the features and the target is absent, which is usually a sign of features poorly suited for a supervised learning task. Some features have a considerably higher correlation to the target than others, although still being significantly lower than the threshold defining a weak correlation of 0.3. The most informative indicators in terms of the correlation coefficient are the AROON indicator, along with the stochastic oscillator and RSI.

Figure 30 tells another story: the AROON indicator being the least significant feature for determining regression values. The RSI and ROC are primarily found as the most significant features for the regression decision tree.

The same exploration of feature and target correlation was performed on a subset of data, only including instances with targets having absolute values greater than 10. The intention of the said method was to discover correlations of feature data and targets considered price fluctuations of significance. The correlations were still poor, although the correlation values did increase significantly relative to the previous correlation values.

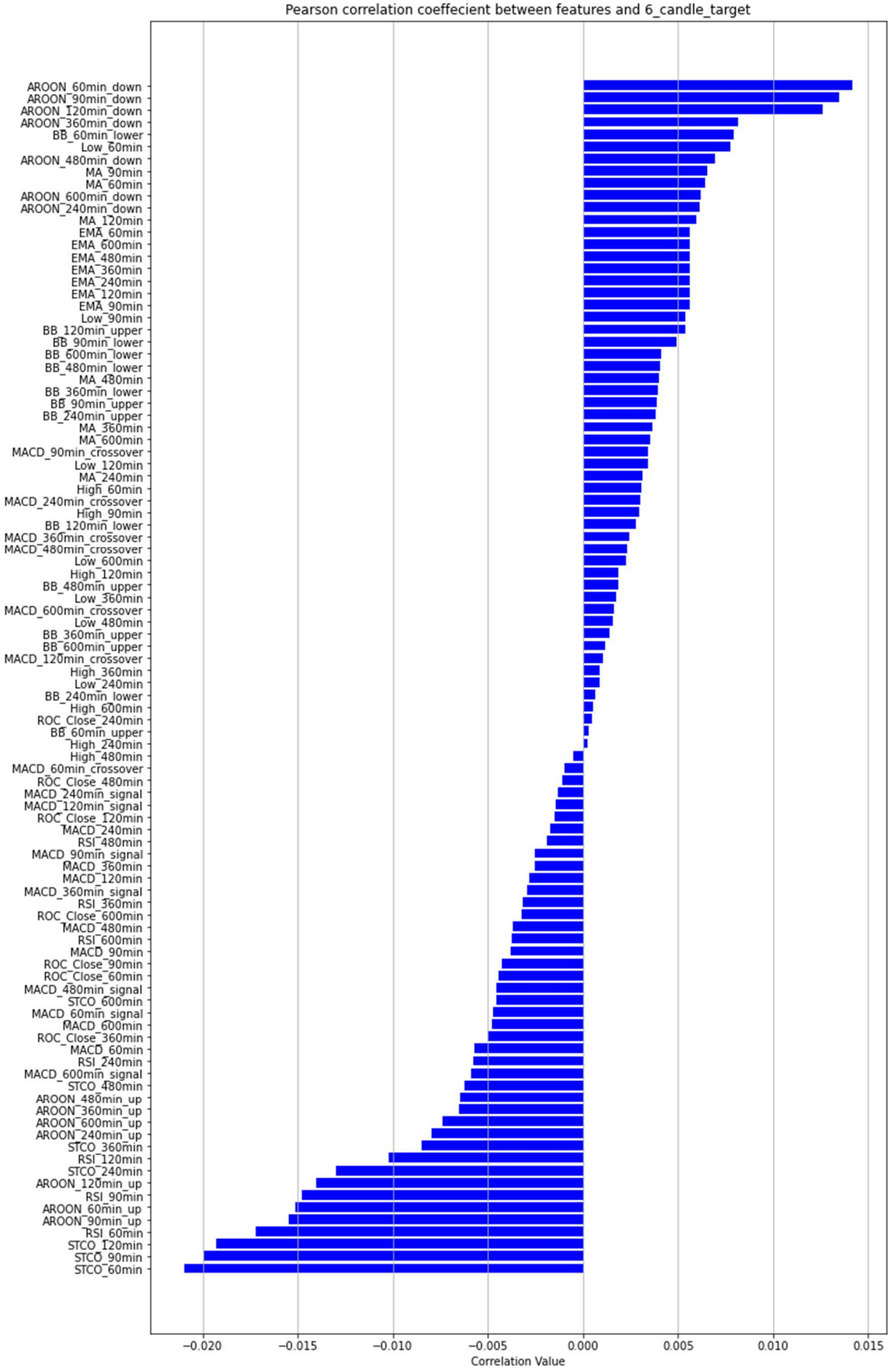


Figure 29: Correlation Co-efficient values of features with respect to future candles 6.

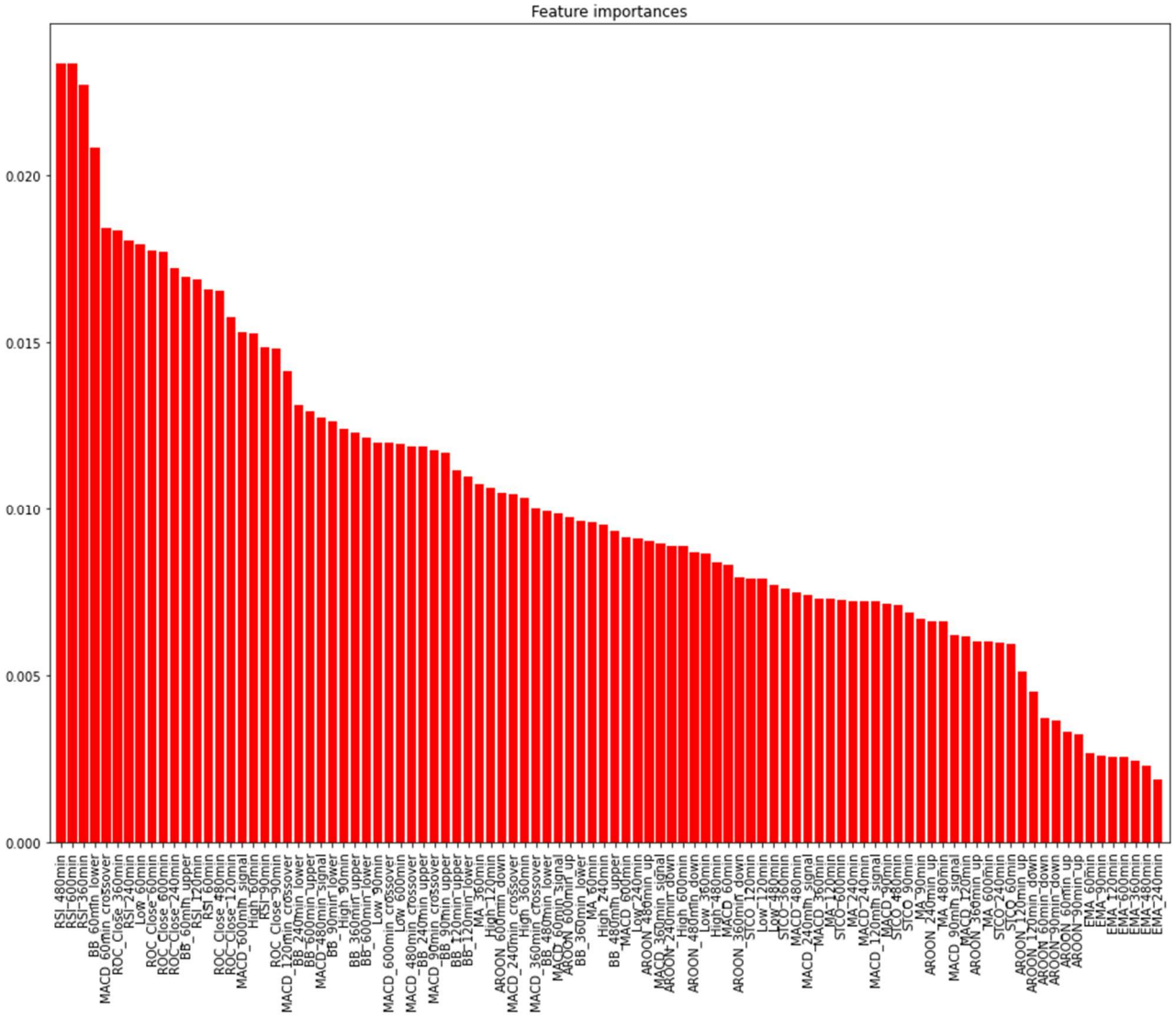


Figure 30: Feature importance found with Gini impurity with future candles 6 the target.

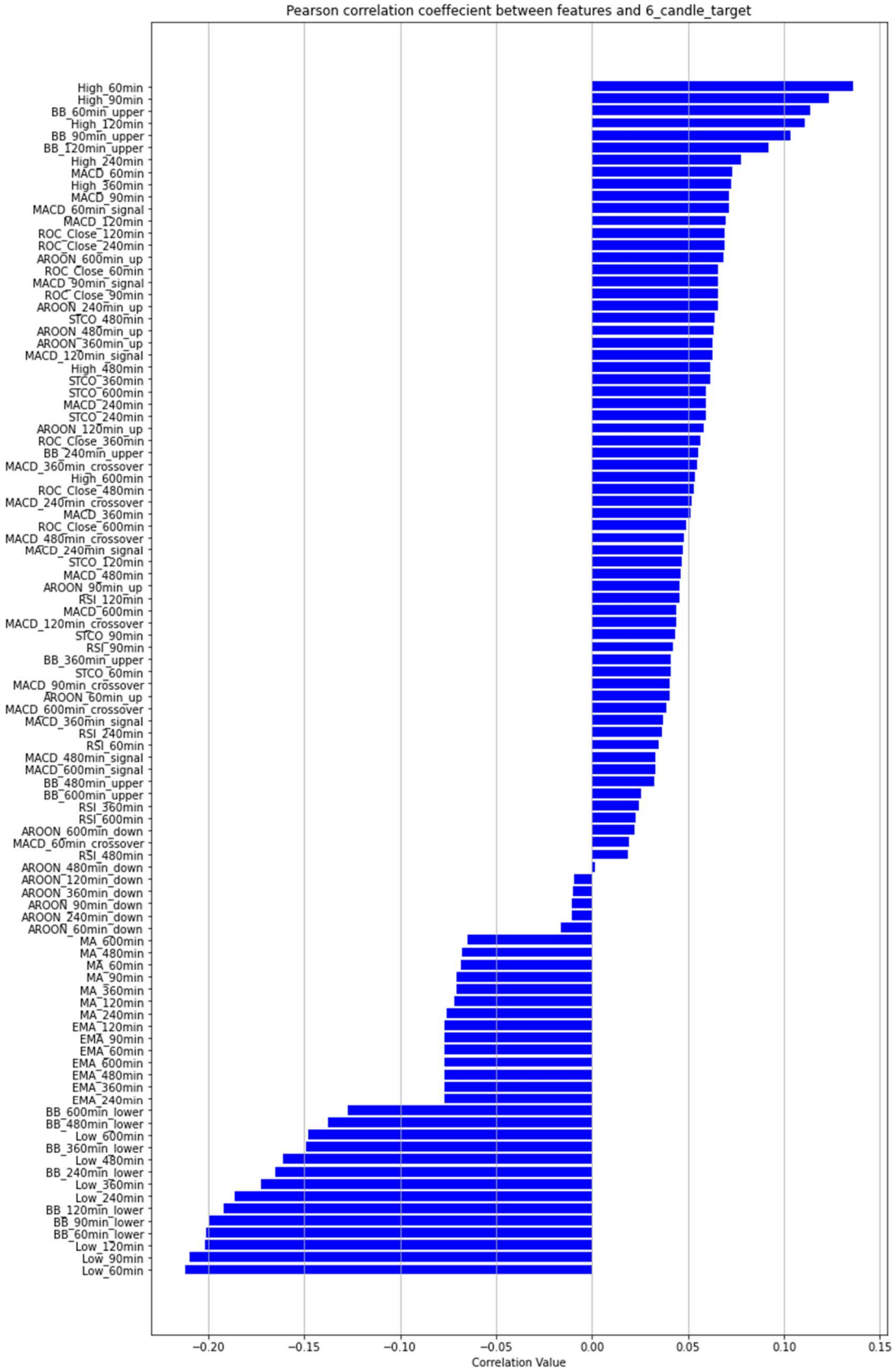


Figure 31: Correlation Co-efficient values of features with respect to future candles 6, only using instances with targets with absolute values greater than 10.

6 Machine learning models

Supervised learning for predicting financial market data using neural networks is a popular topic among machine learning practitioners. (2020, Sidra Mehtab et al.), applied CNN and LSTM models using 5-minute interval stock price data, motivated by the characteristics of the models, which will be discussed further in section 6.2 and 6.3.

For this research project, both the LSTM and CNN configurations will train up to 1000 epochs, with a patience of 3. Patience is a method used while training neural networks. After each epoch, the validation loss is evaluated. If the validation loss does not increase for n-epochs, n being the value of patience, the training process is stopped. The implementation of patience kept track of the weights yielding the lowest validation loss, which were re-applied before stopping the training. The intention of this practice is to stop a model from overfitting, which is characterized by a comparably large difference in training loss and test/validation loss. Overfitting is also characterized as a model lacking the ability to generalize.

Multiple configurations of the two mentioned model architectures will be trained, with randomized configurations defining each trained model. The options and ranges for hyperparameters will be discussed in detail in section 7.3. Although many hyperparameters are randomized, the overall model structures remain constant for both neural network architectures.

6.1 Baseline model

The baseline model for the experiment will be the classic implementation of multivariate linear regression, using the least squared error to determine the high dimensional affine subspace used to regress the corresponding predicted values for the test set.

As a baseline, the future candles will be set to 6, as this is hypothesized to be the best performing future candle value. The optimal value for future candles is a hyperparameter to be optimized for each model architecture during the training process.

The performance of a linear regression model will further determine whether the features and targets have a linear relationship, as some trading strategies imply. Examples of the said strategies include: a price breaking a previous High or Low, the MACD crossing the zero line, or the value of the STCO indicator reaching nearing a maximum or minimum value.

6.2 LSTM

6.2.1. Motivation behind LSTM's

Recurrent neural networks in various configurations and architectures have shown promising results in supervised learning when applied to time series or sequential based data. The LSTM network model structure was first proposed by (1997, Hochreiter & Schmidhuber) and has since gained much attraction in natural language processing, music generation, weather forecasting, market price prediction, alongside other sequence-based tasks. The primary motivation for using an LSTM is due to the features derived from FOREX market indicator data being sequential with suspected sequence patterns defining future price fluctuations. Furthermore, LSTM architecture's are additionally known to perform well on time-series data.

The architecture of the LSTM is elaborated below. A hypothesis for the performance of the LSTM is that during training, the layers will reveal some sequential patterns of events happening before price fluctuations occur despite the poor correlations between the feature data and the target.

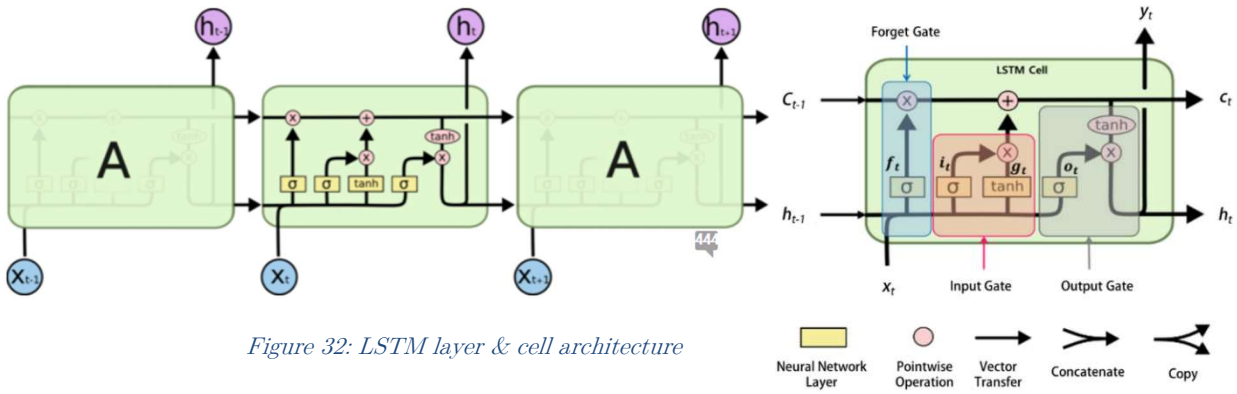


Figure 32: LSTM layer & cell architecture

Source of image: https://colah.github.io/

LSTM networks use a matrix as input data of dimension (n,t), where n represents the number of features and t represents the window size. The number of LSTM cells is proportionate to the window size t of the input data. The input vector x_t of length n for a given LSTM cell, is concatenated with the previous hidden state h_{t-1}, a vector which length corresponds to the number of units defined for the layer. The concatenated vector $h_{t-1} \frown x_t$ will be referred to as the "combined input".

The combined input is passed through 4 fully connected networks (FCN), each with a defined objective.

The combined input is passed through the "Forget Gate", regulated by an FCN with sigmoid activation functions. The output of the activation functions returns a vector of decimal values, which are multiplied by the values of the prior cell state C_{t-1}. The effect of this is a regulation of the "long term" memory passed through each cell.

The current input is used as input for two other FCN's, one with a sigmoid activation function and the other with the hyperbolic tangent activation function. The output vectors of the FCN's are multiplied, allowing for both positive and negative values added to the regulated cell state vector. This is due to $-1 < tanh(x_{i,t}) < 1$. The events of the two prior sentences are visually displayed as the "Input Gate".

Lastly, the vector defining the current cell state C_t is copied. One copy is passed the previous cell state to the next cell, whereas the other copy passes through an FCN with the hyperbolic tangent activation function. The output of said FCN is multiplied by the output of the fourth FNC, using the current input, which uses the sigmoid activation function. The last actions mentioned are referred to as the "Output Gate", which defines the current hidden state h_t, passed to the next LSTM cell while also being passed as the output y_t.

For stacked LSTM architectures, the sequential output of LSTM layers is used, given that the following layer is an LSTM layer. The last LSTM layer only passes the last (rightmost) computed vector of y.

6.2.2. Project implementation of LSTM

The general configuration style of the LSTM's used in this research project is as seen in figure 33 and 34 below. The first layer of the network is an LSTM with the number of units defined as a hyperparameter. The following layers consist of hidden LSTM layers, where a layer of batch normalization is added prior to the hidden layer. The intention of this will be further elaborated in section 6.4, discussing batch normalization.

The last LSTM, whether it is the input layer or a hidden layer, returns a single vector, as seen in the output shape in figure 34, where the output vector of said layer, has a shape equal to the number of hidden layer units. The last layers of the models potentially have dense layers (defined as a hyperparameter) before the last dense output layer is reached, consisting of a single node, regressing the predicted value for the input data.

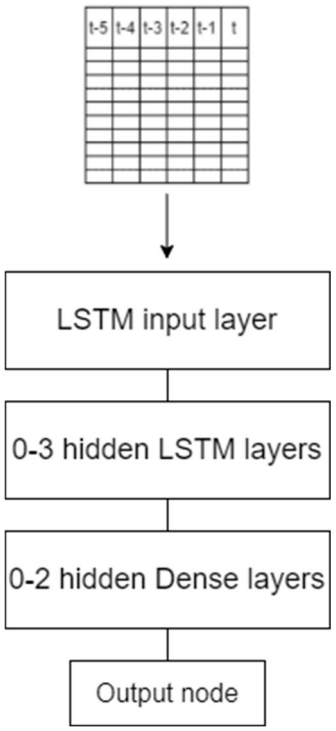


Figure 33: LSTM model structure for project

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_layer_LSTM_id1 (LSTM)  (None, Window size, Input units parameter)
_________________________________________________________________
1_h_layer_BatchNorm_id1 (Bat (None, Window size, Input units parameter)
_________________________________________________________________
1_h_layer_LSTM_id1 (LSTM)    (None, Hidden layer units parameter)
_________________________________________________________________
1_dense_layer_id1 (Dense)    (None, 50)
_________________________________________________________________
Output_layer_id1 (Dense)     (None, 1)
=================================================================


_________________________________________________________________
```

Figure 34: Example of LSTM model description

6.3 CNN

6.3.1. Motivation behind CNN's

Convolutional neural networks have shown promising results in the domain of computer vision, with applications in object detection, classification, and regression, primarily applying image-based data. Various configurations of convolutional neural network architectures have shown a state of the art performance on the ImageNet dataset. The ImageNet data consists of more than 14 million images with 20000 different labels, specified for object detection tasks.

(2020, Mehtab et al.) presented a research paper on "Robust Analysis of Stock Price Time Series Using CNN and LSTM-Based Deep Learning Models", portraying a similar motivation of using CNN's and LSTM's as I have. The motivation is derived from CNN structure's capability of pattern recognition in images, represented as tensors. The hypothesis is that the 2D matrix with dimension represented by each feature in the dataset and a defined window of time, as described in section 5.9, presumably containing patterns which might be correlated with future price fluctuations.

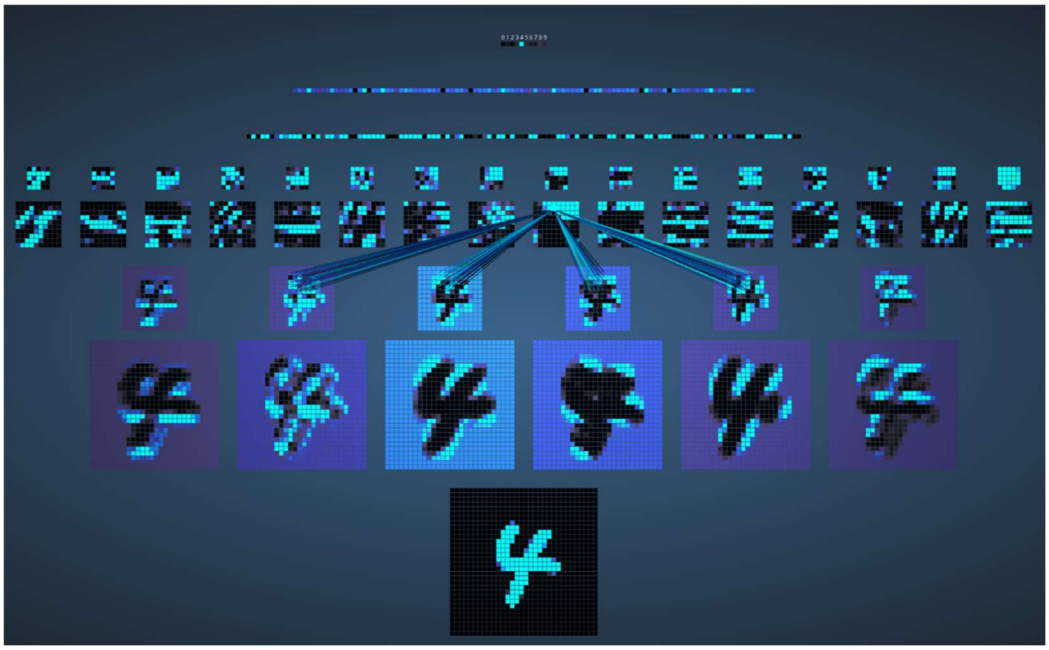


Figure 35: Example of CNN for classification of hand written digits by Adam Harley

Figure 35 illustrates how patterns in handwritten digits are found by using convolutional layers and downsampling layers before passing on informative patterns to an FCN. Lastly, the FCN passes on information to a SoftMax layer, classifying the image as a digit between 0 and 9.

Each convolutional layer has a parameter denoting the number of filters used for that layer. The filters are matrices with randomly initialized values, having a width and length defined by the kernel size. The output of a filter having performed a convolution on some input is called a feature map.

The number of feature maps created is proportionate to the number of filters of that convolutional layer.

A single value of a feature map is created by a section of the input data, proportionate to the kernel size. Each value of this section is multiplied by the values in the filter with a corresponding placement. The multiplied values are summed and passed through a defined activation function, which defines the output of a single value in the feature map.
To calculate the next value of the feature map, the stride parameter determines the shift of the filter, where the following actions of multiplying the selected input values by the values with the corresponding placing in the filter is performed again. This continues after the filter map is constructed.

The combined feature maps are used as the input data for the following layer, creating a 3D tensor, with a width and height determined by the shift, kernel size and the padding of input data. The depth of the tensor is determined by the number of filters applied in the convolutional layer.

In figure 35, the input dimension of the input image is 32x32x1, representing a greyscale image of ahand written digit.
The first convolutional layer creates data from feature maps of 28x28x6, meaning that the convolutional layer had 6 filters, of kernel size 5, with a stride of 1.

6.3.2. Project implementation of CNN

The general structure configuration of the CNN's used in this project consisted is described in detail below. Many parameters were slightly static due to the nature of the data. The window size of the input data was defined as a fixed value rather than a hyperparameter for CNN models. As the features for a given time are derived by OHLC values prior to said time, it was decided that a window size of past feature data should not be too large. This window size denoted as w in figure 36 should not be too small either, as the shape of the input data becomes smaller and smaller throughout the network. Decreasing the data shape can be avoided using upsampling or padding; however, neither of those choices seemed particularly attractive given the task at hand.

Figure 36 above shows the general configuration of the CNN models used. The first convolutional layer was always followed by an average pooling layer, whereas additional hidden convolutional layers were always following by a max-pooling layer. Before each hidden convolutional layer, a batch normalization layer was included. Furthermore, the stride of convolutional layers was set to 1, while set to 2 for pooling layers. The first convolutional layer is defined with a kernel size of 7, while hidden convolutional layers were defined with a kernel size of 4. Pooling layers had a kernel size of 2.

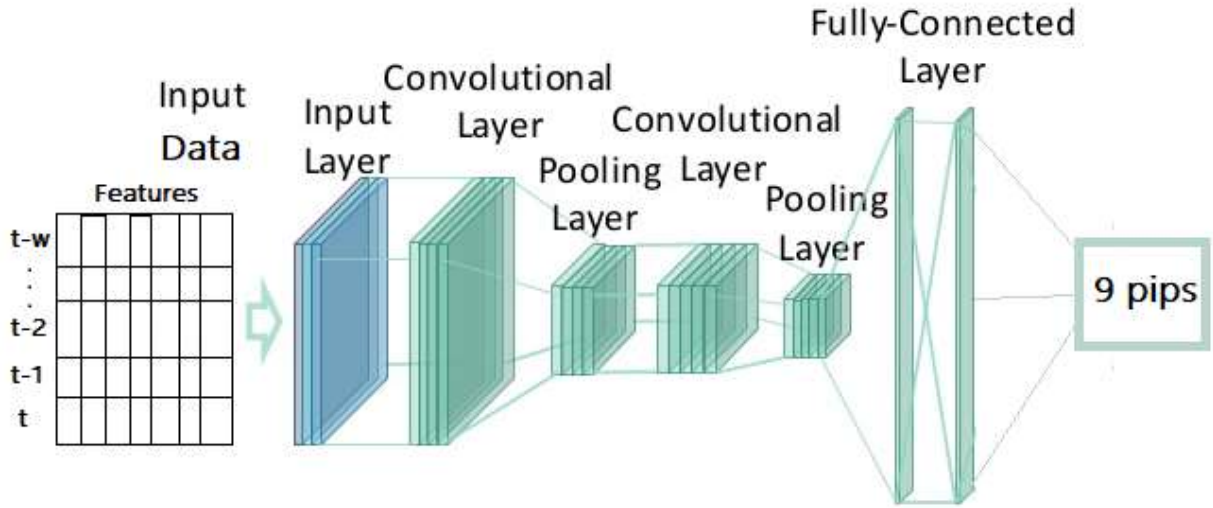


Figure 36: CNN general implemented configuration for this project.

6.4 Batch normalization

Batch normalization is implemented in deep neural network structures, attempting to avoid exploding/vanishing gradient problems by normalizing a layer's output. In implementations for this project, batch normalization is added before a hidden layer in the network configuration. The normalized data is relative to the distribution of the batch, making the mean and standard deviation of the data for that batch 0 and 1, respectively.

The benefit of this measure is ensuring that the following layer receives data that does not have a very large or small standard deviation. Problems using batch normalization may apply when using a smaller value for the batch size (such as 2-10), as the distribution of the initial input values for each batch may differ more than it otherwise would with batch sizes of 128. A large batch size may also cause problems, as the changes of internal weights in the network during backpropagation are changed to minimize the loss functions MSE and MAE.

The mentioned issue of large batch sizes is not due to batch normalization but rather a problem when optimizing weights regarding the loss function. Important instances of input data defining large future price fluctuations may become less dominant during backpropagation for precisely updating internal weights of the network.

The loss functions defined below state that the loss is calculated by the mean of the differences between target values and predict values for a batch of size n. The suspension mentioned above of the reduced predictive capabilities of models using large batch sizes, is due to the distribution of target values reported in section 5.5.

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|Y_i - \hat{Y}_i| \qquad \text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

(2017, Cooijmans) demonstrated how bach normalization increased the performance of an LSTM model in several areas, including validation accuracy during training and improved generalization. The research proposed batch normalization between LSTM cells for hidden states and cell states.

The implemented models for the project used batch normalization before each hidden layer, regardless of the layer type being dense, LSTM or Convolutional.

7 Experiment Setup

7.1 Hardware & Software configuration

All experiments were trained on a local machine, using the high-level TensorFlow API wrapper called Keras. The specifications of the local machine, python version, CUDA installation, and Keras version are as followed.

Hardware Configuration:
CPU: i9 9900k
GPU: GTX 1060 6GB
RAM: 4x8GB (check ram speed)
Software Configuration:
Python 3.7.4 (conda 4.7.12)
CUDA: v10.1
Keras version: 2.4.3

7.2 Experiment structure

Part of this research paper is focused on hyperparameter optimization for the neural networks trained. Some ranges and configurations for hyperparameters were assumed to yield favourable results while also being within computational feasibility. The reason for this is that the "best" hyperparameters for a model are often unknown and can increase performance considerably. With that said, the need for hyperparameter tuning was needed in order to answer the research question fully. The best models constructed are additionally compared to a baseline model, which in this case will be a multivariate linear regression, as described in section 6.1.

The action of using a baseline model has several benefits. Firstly, the baseline model is able to show the features predictive capability by fitting the most optimal hyperplane for the high-dimensional space created by the features and the target. Secondly, comparing more complex models with a baseline model will reveal the effectiveness of creating time series formatted data for predictive regression.

7.3 Hyperparameters

Both models chosen for this experiment have many options of hyperparameters to choose from, however, some configurations must stay static, for the same reason as the random search approach was taken. The hyperparameters used in this experiment are as followed:

1. Future Candles: The number of rows shifted when calculated the difference in the future close price to the current close price. Future candles can hold values 3, 6, 12, 18.
2. Batch size: Exposing the neural networks to different batch sizes may yield different training results when backpropagating.
3. Optimizers: Adaptive momentum estimation (Adam), Root mean square propagation (RMSprop), and lastly, a variant of Adam (AdaMax) are used.
4. Fully connected dense layer: A parameter adding the possibility of ending the network with one or two dense layers before the final regression node. Adding a single layer results in a layer with 100 nodes, while two layers consist of 100 nodes followed by 50 nodes.
5. Dense layer activation functions: Elu, Relu, Hyperbolic Tangent
6. Learning rates, varying depending on results.
7. Loss functions: Mean Absolute Error (MAE) and Mean Squared Error (MSE)

7.3.1. Hyperparameters for LSTM

Apart from the general hyperparameters defined above, the model architecture of the LSTM models had some hyperparameters defined explicitly for it. This included:

1. Input layer nodes: An integer being either {128,256,512}, defining the number of input nodes/units used for the input LSTM layer.
2. Number of hidden layers: An integer with a minimum value of 0, and a maximum value of 3, defining the amount of hidden LSTM layers added to the model.
3. Number of hidden nodes: An integer being either {64,128,256}, defining the number of input nodes/units used for the hidden LSTM layers. The number of hidden nodes are equal for each hidden LSTM layer.
4. Dropout rate: A hyperparameter defining the dropout rate for each hidden LSTM layer. Once defined for a model, the dropout rate is consistent for all hidden layers if they were added. The dropout rates were either 0, 0.1 or 0.2.
5. Window size: The hyperparameter defining the shape of the input for the LSTM, having values {6,12,18,24}

7.3.2. Hyperparameters for CNN

CNN's are often used for image processing, where the shape of the input data limits the flexibility of some of the most significant hyperparameters such as kernel size, strides, and pooling size. Additionally, the input window size is usually a fixed parameter, defined by the width/height of an image. For the experiments, the window size was fixed at 28, minimizing the need for padding and reshaping throughout the layers. The kernel size for the input convolutional layer was set at 7x7, and the hidden layers kernel size was set at 4x4, both with a stride of 1.

The hyperparameters used were:

1. Number of Filters for input layer: Minimum value of 8, and a maximum value of 16, with steps of 4.
2. Number of hidden layers: As the model structure included pooling after a convolutional layer, the number of hidden layers were limited to have a value of 0,1 or 2.
3. Number of filters for hidden layers: Minimum value of 8, and a maximum value of 24, with steps of 8.
4. Activation function: A selection of activation functions for the convolutions will be added as a hyperparameter: ELu, ReLu, Hyperbolic Tangent.

7.3.3. Training models

Hyperparameter tuning was done using a random search approach, as proposed by (2012, Bergstra et al.). A random search approach was primarily chosen due to limited computational power, but also in order to efficiently gather information of favourable configurations with fewer trained models.

After training a model, its regressed labels for the EURUSD 2020 run through a trading simulation of various thresholds, triggering simulated trading positions. The results of the trading simulations will dictate the hyperparameters to be removed, changed, or kept for the following iterations of model training. Using the results of the trading simulation instead of the loss score on the test dataset was because of the model's ability to predict fluctuations being the main point of interest, not the overall accuracy, although presumably closely related.

7.4 Simulating trading

As the purpose of this research is to determine whether technical indicators are descriptive for machine learning models, a very simple, naive, and potentially penalizing approach for the trading simulation will be implemented.

Trading strategies are implemented as a measure of minimizing risk and maximizing profit. If done correctly, the effects should positively influence the outcome of one's cumulative pips gained. In a non-academic setting, this is most definitely desired; however, in this analysis, a straightforward and simplistic approach is favourable as the results are kept transparent.

The outcome of the simulations are presumed to be penalizing as a significant part of FOREX trading is dependent on strategies. The said strategies influence actions while holding a given position, closing a given position, or creating a position. Consider a trader has a trade placed which continuously declines for a certain amount of time. It is suspected that most individuals would opt out of the given position, cutting their losses. Likewise, if a given position keeps increasing in value, the trader is expected to keep the trade longer than expected initially.

The implemented simulation was programmed with so-called "diamond hands" for the pre-defined period. The implemented "diamond hands" mean that the position of a trade placed is held for a pre-duration corresponding to the future candle value used for that model. A sudden drop or increase in price does therefore not influence the trade duration.

7.4.1. The implemented trading simulator

The method for evaluating trained models will be done by simulating a trading year. Trading simulations are based on the predicted labels created from a given model evaluating a test set, the EURUSD 2020 dataset. The following rules determine the actions performed during the simulation:

1. For any given point in time, only one active trade may be open.

2. A trade is active for the time corresponding to the test set parameter; future candles, e.g., future candles with a value of 12 will result in 120-minute positions, while future candles with a value of 3, will result in 30-minute positions.

3. The performance of the model's predictions is measured by two variables. Firstly, the cumulative pips gained obtained by trades performed during the simulation. Secondly, pips ratio; the number of pips gained relative to the number of pips traded.

4. A trade is placed when a pre-defined threshold is met. For each threshold, both a buy and sell threshold exists, defined by $\mu \pm \sigma$ of the predicted labels.

5. The predefined **spread is set to 4 pips** , meaning that any trade placed will have an unfavourable starting position shifted by 2 pips, with the intent to emulate the spread and cost of trades of an online broker.

For a given set of predicted labels, the mean and standard deviation of all predicted labels are used to determine thresholds for when trades should be placed during the simulation. A single simulation trades with one of four different thresholds, using multipliers {1,2,3,4} on the standard deviation. A hypothesis being that the thresholds defined by larger value, e.g. a value of 4 will place more winning trades at the cost of fewer cumulative pips gained. The lower thresholds will presumably place more trades, subsequently resulting in more cumulative pips gained, given that the pip ratio is above 0.

These thresholds are used as the point of interest is a model's interpretation of a single instance of input data relative to other instances. In essence, the regression problem becomes a classification problem as the regressed labels determine 3 different actions, namely: Buy, Sell, or No action taken.

7.4.2. Evaluating simulations

Models will be evaluated by their performance of a trading simulation using the test set. The performance metrics are cumulative pips gained and pip ratio. A third metric was defined to maximize a hypothetical profit while still considering the risk. The proposed metric is defined in the formula below: the square of cumulative pips gained by the pip ratio. The metric is to be maximized and is used to rank a model's performance during hyperparameter tuning. As an appropriate name for the invented metric was lacking, it will be referenced as mRMP, an abbreviation for "minimize Risk, Maximize Profit".

$$mRMP = (cumulative\ pips\ gained)^2 \cdot pip\ ratio$$

8 Best model configurations

Three iterations of hyperparameter tuning were performed, creating a combined total of 269 different LSTM models and 182 different CNN models. The best performing CNN and LSTM models were both found during the first iteration of hyperparameter tuning. The parameters of said models are displayed below.

Best models hyperparameter configuration

Hyper Parameter name	LSTM Paramter value	CNN Parameter value
Learning Rate	0.001	0.0005
Loss function	MSE	MAE
Optimizer	AdaMax	RMSprop
Number of hidden layers	1	0
Input Units/Filters	128	12
Hidden Units/Filters	128	N/A
Convultional layer activation function	N/A	Hyperbolic Tangent
Dropout Rate	0.2	0
Batch size	256	64
Window size	12	28
Future candles	12	3
Dense Layer 1 units	100	100
Dense Layer 2 units	N/A	N/A
Dense Layer activation function	Elu	Elu

9 Trading simulation results

The results of the models reflected in trading simulations yielded some interesting insights. Despite the feature data correlating poorly with future price fluctuations, some models created did perform decently. Model's performances were penalized during simulations, as the spread was set at 4, meaning that the resulting pips gained of each trade placed was subtracted by 2.

Section 9.2 displays a table of the baseline model results for the 4 different thresholds used to place trades. The LSTM and CNN performance sections show the models which achieved the highest pips gained during trading simulations, along with their respective threshold for executing trades. The other threshold results for the model with the intention of transparent reporting.

Lastly, the best performing LSTM and CNN model will be tested on EURGBP and GBPUSD for 2015 - 2020, as a supplementary test set, to see the models' performance on different currency pairs. The results of the simulations based on different currency pairs, are not considered the primary results to analyze for two main reasons:

1. The models are trained on the EURUSD, which is a different currency pair. It is unknown whether this influences the simulation results, which is part of the motivation for showing the mentioned results.
2. The years 2015 to 2019 were incorporated with in the training set. As both the currency pairs share at least one currency with the EURUSD, it is expected that the results will be affected by this.

The other currency pairs are included to gain insights into longer-term performance and the capabilities of a model trained on a different currency pair.

9.1 Random guessing

The hypothesis of this project states that the results of the simulations are expected to perform better than random guessing. As a suitable measure of expectations, an expression for the pips gained and pip ratio by random guessing is stated below.

Let Z denote a random variable of a standard normal distribution, defining a random the decision to either place a short or long position. Let the value of n denote the number of trades taken, x_i the value of pips gained from trade i, and s, the spread included in the trading simulation.

The numerical values for the mean and standard deviation are stated below and are taken from the target data statistics from section 5.5.

$$\mu = -0.02$$
$$\sigma = 5.2987$$
$$s = 4$$
$$x_i \sim N(\mu, \sigma); \quad i = 1, \ldots, n$$

$$Z \sim N(0, 1)$$

$$Pips\ Gained = \left(\sum_{i=1}^{n} \begin{cases} -x_i, & Z \geq 0 \\ x_i, & Z < 0 \end{cases} - \frac{s}{2} \right)$$

$$Pips\ Gained = n(\bar{x} - \frac{s}{2})$$

$$Pip\ Ratio = \frac{n(\bar{x} - \frac{s}{2})}{\sum_{i=1}^{n} |x_i|}$$

$$lim_{n \to \infty} \quad , \quad \bar{x} \sim \mu$$

$$Pip\ Ratio < 0$$
$$Pips\ Gained = -\infty$$

9.2 Baseline performance EURUSD 2020

The following results are of the baseline linear regression model, along with a visualization of the cumulative pips gained for the best threshold value, namely σ_3. As mentioned in section 7.4, the simulation involves no trading strategies. The value for future candles is set to 6 in this simulation, meaning that the trades were placed and held for 60 minutes before closing the position.

	Pips gained	Pip ratio	Wins	Losses	Total trades	mRMP
Base, σ_1	-6256.5	-0.234	1286	1982	3268	-9159647
Base, σ_2	-543.2	-0.0756	301	370	671	-22307
Base, σ_3	-43.3	-0.0204	62	80	142	-38.25
Base, σ_4	-91.2	-0.1097	16	26	42	-912.42

Figure 37: Linear Regression model, σ_3

9.3 LSTM performance EURUSD 2020

Best LSTM models	Pips gained	Pip ratio	Wins	Losses	Total trades	mRMP
Model 96_1, σ_1	-1156.5	-0.0558	714	834	1556	-74568
Model 96_1, σ_2	-297.8	-0.0317	270	301	571	-2812
Model 96_1, σ_3	513.8	0.1349	95	106	201	35613
Model 96_1, σ_4	486.8	0.5633	27	19	46	133486

Model 96 of the first iteration of hyperparameter tuning runs scored the highest pips gained and mRMP of all LSTM models trained, gaining 513.8 pips on EURUSD during 2020.

The model's future candle value was 12, meaning it used the indicators to predict the change in price 120 minutes ahead of the current price.

Figure 38: CNN model 96_1 on EURUSD, 2020

9.4 CNN Performance EURUSD 2020

Best CNN models	Pips gained	Pip ratio	Wins	Losses	Total trades	mRMP
Model 87_1, σ_1	-6448.7	-0.2384	1490	2132	3644	-9916830
Model 87_1, σ_2	-1567.1	-0.1589	444	531	981	-390293
Model 87_1, σ_3	132.9	0.0368	161	150	311	650.79
Model 87_1, σ_4	454.8	0.2995	67	43	110	61947

Model 87 of the first iteration of hyperparameter tuning runs scored the highest pips gained and mRMP of all CNN models trained, gaining 453.8 pips on EURUSD during 2020.

The model's future candle value was 3, meaning it used the indicators to predict the change in price 3 minutes ahead of the current price.

Figure 39: CNN model 87_1 on EURUSD, 2020

9.1 Results on other currency pairs

The following subsections will show the results of running the same trading simulation as performed on the EURUSD 2020, but on the EURGBP and GBPUSD during 2015-2020 instead. With the intention of proposing valid results, the thresholds previously defined by the prediction set's mean and scaled standard deviation of EURUSD 2020, will be used on the following two currency pairs.

The threshold which performed best for the LSTM and the CNN on the EURUSD 2020:

Model	Long Threshold	Short Theshold
LSTM	28.4182	-29.0056
CNN	2.38001	-2.56032

These thresholds differ a lot from each other, despite the LSTM placing more than twice as many trades throughout the EURUSD 2020 simulation. A visualization of the predicted labels compared to the test set will be displayed in the discussion, which will explain why these thresholds differ as much as they do.

9.1.1. LSTM EURGBP 2015-2020

Best LSTM models	Pips gained	Pip ratio	Wins	Losses	Total trades	mRMP
Model 96_1, σ_3	-1687.2 pips	-0.0846	627	763	1390	-240826

Figure 40: LSTM, model 96_1 on EURGBP, 2015 - 2020

9.1.2. CNN EURGBP 2015-2020

Best CNN models	Pips gained	Pip ratio	Wins	Losses	Total trades	mRMP
Model 87_1, σ_4	5.9	0.0007	391	405	797	0.0244

Figure 41: CNN, model 87_1 on EURGBP, 2015 - 2020

9.1.3. LSTM GBPUSD 2015-2020

Best LSTM models	Pips gained	Pip ratio	Wins	Losses	Total trades	mRMP
Model 96_1, σ_3	2156.8	0.06911	628	647	1275	321487

Figure 42: LSTM, model 96_1 on GBPUSD, 2015 - 2020

9.1.4. CNN GBPUSD 2015-2020

Best CNN models	Pips gained	Pip ratio	Wins	Losses	Total trades	mRMP
Model 87_1, σ_4	-2150.3	-0.1699	317	331	648	-785406

Figure 43: LSTM, model 87_1 on GBPUSD, 2015 - 2020

10 Discussion

The analysis and machine learning experiments provided insights into the structure and nature of FOREX data. The data created from the technical indicators were transformed into features with two intentions. Firstly, the indicator data was transformed, attempting to encode similar information perceived by human traders. Secondly, scaling the data with the intention of creating numerically informative data suitable for neural networks.

Conscious decisions were taken to avoid common neural network training problems, such as the exploding/vanishing gradient problem, inadequate input data distributions, and rationalizing the hyperparameters chosen for training.

Several sections of the paper will be reviewed, along with interpretations of the results and findings will be discussed in the following sections.

10.1 Technical indicators

Section 5 of this paper presented the origin of the applied data, its structure, popular technical indicators, and the considerations taken when transforming the indicators to features. In section 5.10, it was discovered that the correlation between the features and the future change in pips was not apparent.

While using the entire dataset, the most significant positive correlation of the features derived from the technical indicators was 0.014, while the most significant negative correlation was -0.021. Moreover, when the subset of the data was applied, defined by instances with high future price fluctuations, the most significant positive correlation of the features derived from the technical indicators was 0.14, while the most significant negative correlation was -0.21.

A high correlation could have been achieved if the target data was defined as the future price rather than the change in future price, and the technical indicators representing a variation of the exchange rate were kept as defined. Although the feature data might have had near-perfect correlations to the unchanged future price, it is suspected that implementing the data in such a way would have proven unsuccessful for trading simulations due to the low variance of input data for said features.

The low correlation between the features and future price fluctuations shaped the foundation of expectations for the feature's potential capability to be informative for predicting future price fluctuations.

10.2 Training-validation split

In section 5.8, the splits of training, validation, and test data were defined. A clear incentive for defining the test set a separated sequential period was taken, as creating a test set of randomly selected instances of the entire dataset would create instances of data similarly found in the training set, due to the 2D input data including a window of past feature data.

Unfortunately, this decision was unintentionally overlooked when splitting the training set into training and validation sets for model training. As a standard practice for 1-dimensional data is to randomly split 20% of the training data and use it as validation data, this practice was used.

As the models were trained using patience, it is suspected that some of the validation data shared similar values as the training data while presumably also having a similar target. The resulting effect is that the machine learning models trained were potentially overfitted, which is a known feat leading to worse model generalization. As this error was discovered late in the project, re-running the experiments was not a feasible option.

10.3 Baseline model

The plot shows a line chart of the placed labels from the baseline model and targets for future candles 6, on EURUSD 2020. The linear function was calculated using least squares regression. The resulting lack of variance in the predicted labels indicates that the feature data and change in the future price do not share a linear relationship.

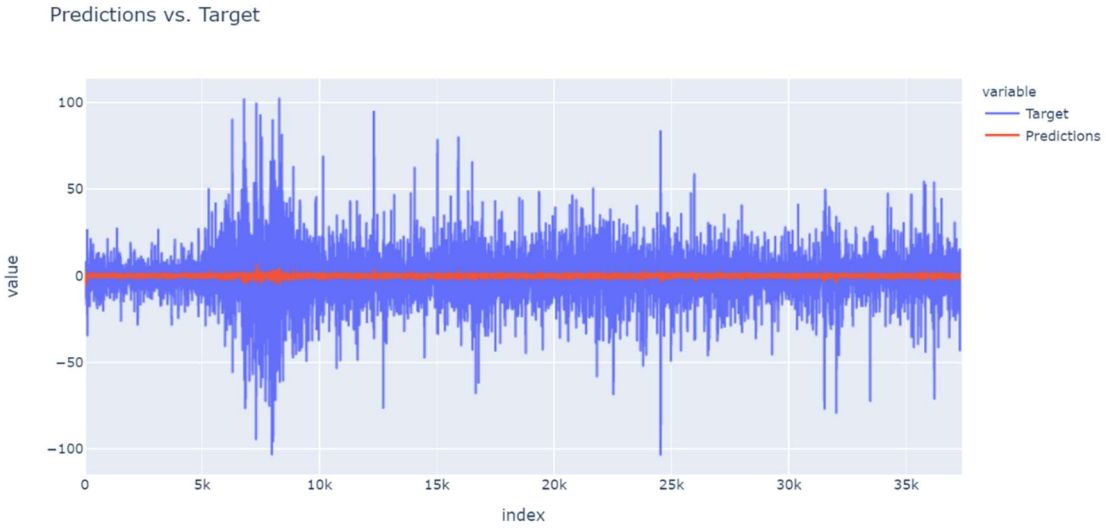


Figure 44: Baseline model predictions vs. Target EURUSD 2020

10.4 LSTM predicted labels vs target

The plot shows a line chart of the placed labels from the best LSTM model and targets for future candles 12, on EURUSD 2020. The best LSTM configuration used MSE as a loss function, which as hypothesized, resulted in larger predicted values than using the MAE, which is seen when using the CNN.

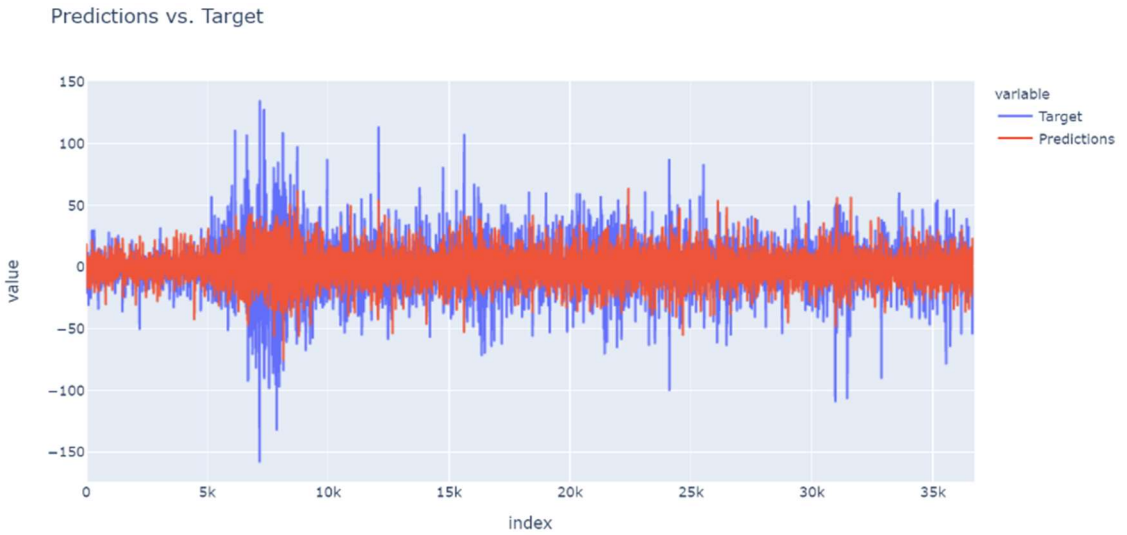


Figure 45: LSTM model_96_1 predictions vs. Target EURUSD 2020

10.5 CNN predicted labels vs target

A line chart of the placed labels from the best CNN model and targets for future candles 3, on EURUSD 2020. The best CNN model configuration used MAE as a loss function, resulting in a visually similar regression function as seen in the baseline model, with the predicted labels have a small variance, relative to the target variance.

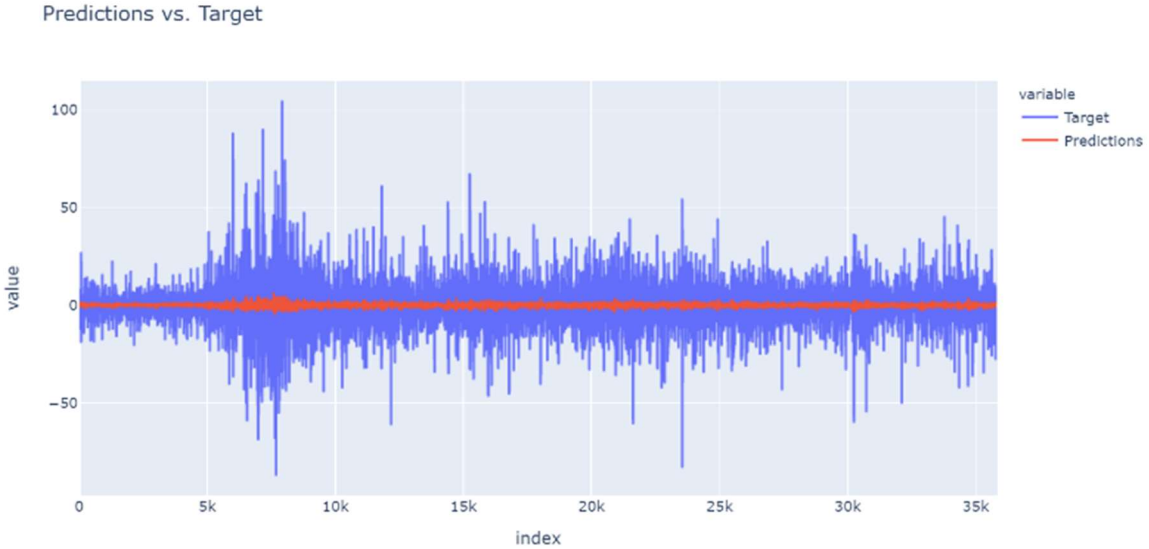


Figure 46: CNN_model_87_1 predictions vs. Target EURUSD 2020

10.6 Interpretation of trading simulation results for EURUSD

The results found from running the trading simulations with the best performing LSTM and CNN models on the EURUSD during 2020, showed that the models were able to perform better than randomly guessing, given that a conservative threshold was used. As presumed, higher pip ratios resulted in much fewer trades, which in total resulted in less pips accumulated over the trading year.

As the lower bound for performance is defined as randomly guessing, an upper bound for performance is defined as perfectly placing trades, at times where the difference in future price is significant. Let the threshold for a price fluctuation be defined as a change in price of 10 pips. Applying the same rules as the trading simulator, the maximum amount of pips for gain for the EURUSD during 2020, for each value of future candles is stated below:

```
Possible pips to gain for EURUSD 2020
Future candles: 3     trades placed: 2603   pips to gain : 40027.4

Possible pips to gain for EURUSD 2020
Future candles: 6     trades placed: 2412   pips to gain : 39403.7

Possible pips to gain for EURUSD 2020
Future candles: 12    trades placed: 1863   pips to gain : 32793.2

Possible pips to gain for EURUSD 2020
Future candles: 18    trades placed: 1491   pips to gain : 28714.4
```

Matching results of the defined upper bound is of course an unattainable objective, although, the values above do enable some sort of comparison.

The best trading simulation results obtained with the LSTM using future candles 12, gained 1.57% of the value defined above.

The best trading simulation results obtained with the CNN using future candles 3, gained 1.14% of the value defined above.

Although not terrible, it must be emphasized that these results were the best found, using 4 different thresholds.

10.7 The magnitude of predicted values as a measure of confidence

A noticeable aspect present in both model results was that the higher thresholds for determining a trade in the simulations significantly increased the pip ratio obtained, naturally at the cost of fewer trades. A curious sub-hypothesis emerged; the regression value may be interpreted as a measure of certainty of predicting the direction of the change in future price. As a method of exploring the mentioned suspicion, the predicted labels and target labels of the EURUSD 2020 will undergo an analysis.

Using the prediction's dataset and the target dataset, a sub dataset is created by removing all entries, where the prediction's absolute value is less than a defined threshold. The remaining predictions and respective targets are used to calculate the pip ratio of the said dataset, with the formulas stated below.

$$Pips\ Gained = \sum_{i=1}^{n} \begin{cases} |y_i|, & sgn(y_i) = sgn(\hat{y}_i) \\ -|y_i|, & sgn(y_i) \neq sgn(\hat{y}_i) \end{cases}$$

$$Pip\ ratio = \frac{Pips\ Gained}{\sum_{i=1}^{n} |y_i|}$$

The measures do not consider the spread or gaps of times between trades. The calculations are purely based on the remaining predicted labels and target labels.

The absolute value of the target is summed if the sign of the predicted value and the sign of the target value are equal. The sign of the predicted value determines the direction of the trade placed.

The tables below display the results of running the above expression for different predicted value thresholds.

LSTM predicted value threshold	LSTM pip ratio	Instances in data frame
4	0.05	20971
8	0.075	11327
12	0.095	6083
14	0.14	3386
18	0.209	1908
22	0.25	1087
26	0.326	596
32	0.489	298
36	0.528	159
40	0.75	89

CNN predicted value threshold	CNN pip ratio	Instances in dataframe
0.3	0.017	19250
0.6	0.031	9244
0.9	0.081	4281
1.2	0.132	2200
1.5	0.103	1142
1.8	0.134	580
2.1	0.211	315
2.4	0.296	174
2.7	0.336	103
3	0.364	57

The table above suggests that only considering higher predicted values correspond to greater pip ratios. The LSTM show significantly better results for higher thresholds than the CNN did. The discovery above supports the sub-hypothesis of a predicted regression value, corresponding to a confidence metric for the direction of future price changes.

10.8 Performance on other currency pairs

The performance results of the LSTM model_96_1 trading on the EURGBP, compared to the results from trading on the GBPUSD are considerably different. In this section, model_96_1, will simply be referred to as the LSTM model. One of results from the simulations stood out, which was that of the LSTM trading on the GBPUSD from 2015 to 2020. During June 2016, the model predicted some labels, which resulted in near-perfect trading actions in the simulation, accumulating 1831 pips within less than 24 hours on 6, 2-hour trades, which accounted for nearly 85% of all pips gained during the 5-year trading simulation. The figures 47 and 48, show the cumulative pips gained and the trading log for the years of the simulation.

Figure 37: Cumilative pips gained of LSTM, GBPUSD

```
BUY taken af idx: 53950  date: 2016-06-22T03:50  pips gained: 21.6
BUY taken af idx: 54005  date: 2016-06-22T13:00  pips gained: 38.6
BUY taken af idx: 54046  date: 2016-06-22T19:50  pips gained: 22.4
SELL taken af idx: 54116  date: 2016-06-23T07:30  pips gained: 33.3
BUY taken af idx: 54132  date: 2016-06-23T10:10  pips gained: 2.8
SELL taken af idx: 54196  date: 2016-06-23T20:50  pips gained: 942.4
BUY taken af idx: 54216  date: 2016-06-24T00:10  pips gained: 332.1
SELL taken af idx: 54242  date: 2016-06-24T04:30  pips gained: 221.5
BUY taken af idx: 54263  date: 2016-06-24T08:00  pips gained: 106.8
BUY taken af idx: 54280  date: 2016-06-24T10:50  pips gained: 137.7
SELL taken af idx: 54294  date: 2016-06-24T13:10  pips gained: 90.3
BUY taken af idx: 54325  date: 2016-06-26T20:20  pips gained: -19.9
BUY taken af idx: 54393  date: 2016-06-27T07:40  pips gained: 46.5
SELL taken af idx: 54417  date: 2016-06-27T11:40  pips gained: 40.1
BUY taken af idx: 54430  date: 2016-06-27T13:50  pips gained: 45.2
```

Figure 48: Trading Log of LSTM, GBPUSD

With the intent of validating the trades placed, a quick look at the GBPUSD during 23rd June - 24th June 2016, on Saxo Trader pro was taken.

As seen in the trading view in figure 49, a measurement of the difference in pips displays near-identical values for the accumulated pips. The close price of the time mentioned in the trading log is used as the starting point, where the close price 12 candles in the future define the end of the trade.

When viewing these particular trades, scepticism immediately arose, as the remaining trades shown for the GBPUSD performed poorly in comparison.

Figure 49: GBPUSD, model_96_1's sequence of perfect trades measured with data from a broker

Being aware that the models were trained on the EURUSD, from 2012 to 2019, a key point of interest was to check whether a similar pattern was found in the training set. This was suspected as the value of the USD is reflected in both the EURUSD, and the GBPUSD. Figure 50 below shows the training data (EURUSD) from 23rd June to 24th June 2016.

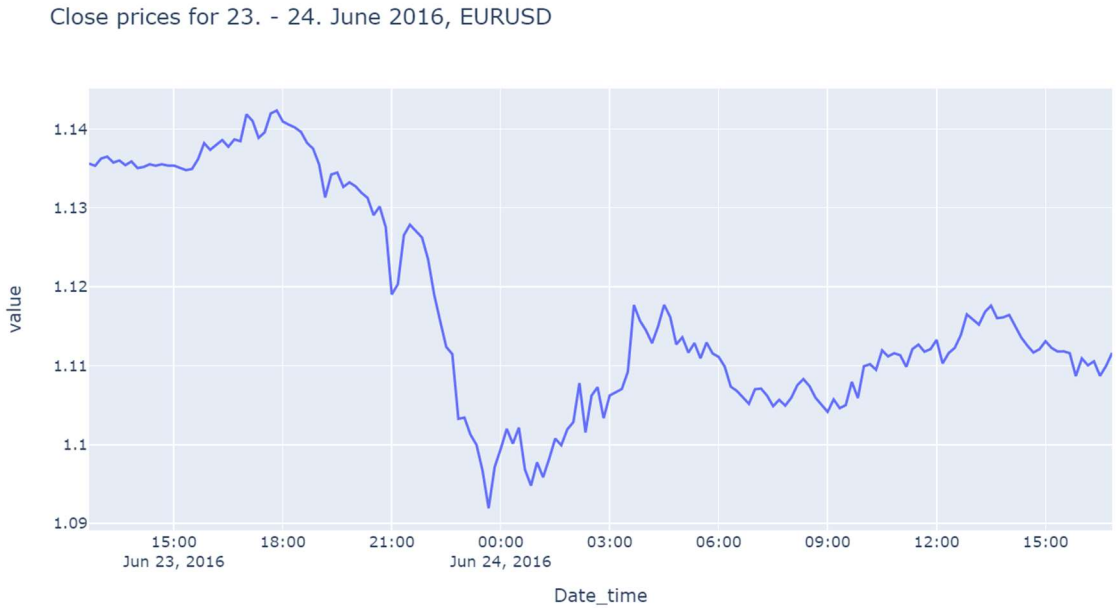


Figure 50: EURUSD, June 23rd - June 24th , 2016

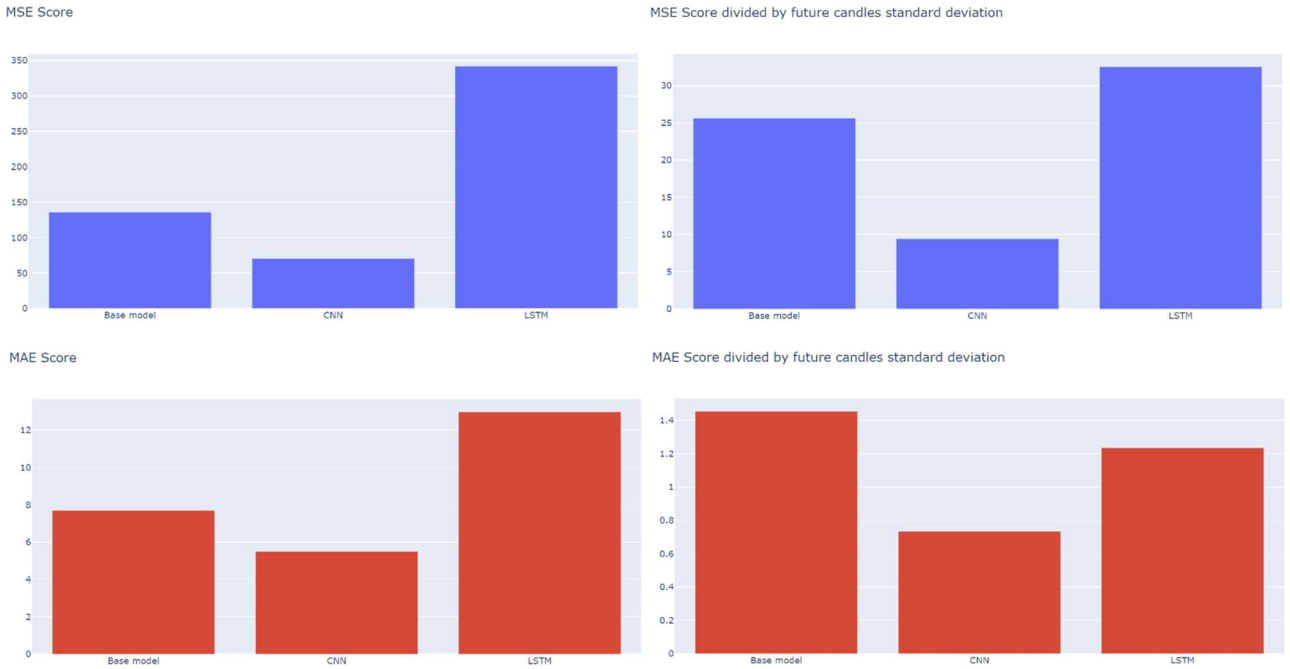
As suspected, a very similar pattern is seen in the training data during the same time period, which is most likely the root cause for the LSTM placing the near-perfect trades.

10.9 Loss functions and metrics

Machine learning metrics such as MSE and MAE were intentionally not included in the results, as one of the hyper parameters during training was the future candles value. Coincidently, the three models used for this paper were each trained and tested, using different future candle values for labels. As shown in section 5.5, the distribution of target values is different for each of the future candle values.

Although the metrics are objectively incomparable, the plots below compare apples of oranges, as the loss function metrics are compared.

The original loss function metrics are shown to the left, while an attempt to normalize the loss functions metrics are found to the right. The "normalized" metrics are divided by the standard deviation of the respective future candles, although, other factors such as the difference in time, presumably induce a high degree of uncertainty, which unfortunately is undefinable.



It was hypothesized that using MSE as the loss function would train models which would perform better in the trading simulations, as it would consider outliers during training (being high price fluctuations) to a greater extent.

$$MSE = \frac{1}{n} \cdot \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

$$MAE = \frac{1}{n} \cdot \sum_{i=1}^{n} |y_i - \hat{y}_i|$$

11 Conclusion

The research project's research showed a poor correlation with the indicators and future price fluctuations as targets. However, the mentioned correlation increased when isolating instances with high price fluctuations, suggesting that some features correlate slightly with future price fluctuations.

The simulated trading results of the neural network models for the EURUSD during 2020, showed that some price fluctuations could be predicted, achieving a simulated trading year with positive results, despite being penalized with an emulation of a spread, subtracting the pips gained for each trade placed. The LSTM gained 513.8 pips, with a pip ratio of 0.1349, placing 201 trades, while the CNN gained 454.8 pips, with a pip ratio of 0.2995, placing 110 trades.

With an ambitious upper bound for defining absolute success, the LSTM and CNN configurations were able to gain 1.57% and 1.14% of the possible pips to gain for that trading year with their respective test sets.

Further analyzing the potential of using the magnitude of a predicted label as a matric for the confidence of the direction of the change in future price showed results worth noting. The LSTM model showed a considerable increase in pip ratio, for predictions of high values, with predictions of values greater than 22 resulting in 1087 predicted labels (2.9% of the data), creating a pip ratio of 0.25, while a label of 32 resulted in 298 predicted labels (0.8%) creating a pip ratio of 0.489.

With endless possibilities of optimizing the said task by creating different time aggregations for OHLC data, applying different machine learning models and hyperparameters, using different methods for feature engineering, or defining the problem to be solved differently, it does seem possible with the results found in this research paper that technical indicators can reflect some degree of information usable for neural networks. However, the results found in this paper do not carry results significant enough, enabling the statement of technical indicators directly informative data for future price fluctuations to be factual.

This paper concludes: technical indicators were found mildly informative for neural networks, with the intent of predicting forex market volatility.

Appropriately stated, the research, models and results found in this paper are far from a free lunch.

12 Future work

- Applying U-nets, Inception nets, or state other of the art convulation nerural network architechtures for price fluctuation forecasting.

- Exposing models to training data with high absolute values of pip targets, with the expectation that predicted labels on the test data will be less accurate in terms of loss, yet would possibly perform better in trading simulations due to more trades taken. This objective is assuming that maximizing mRMP is desired, a combination of pip ratio and pips gained.

- A custom loss-function for the objective of maximizing mRMP.

- Gaining inspiration from anomaly detection problems.

- Aggregations of time candles, perhaps 30 minutes or 1 hour aggreagations.

- A further correlation analysis creating a heat map with one axis consisting of technical indicators and the other axis consisting of combinations of different aggregated timeframes for OHLC combined along with future candles.

- Using a defined continuous time period for validation data, similarly done with the test set.

- Exploring options of maximizing performance, using trading strategies in the trading simulation.

- Research within predicting the direction of fluctuation for a candle 1 step ahead, with the intent of exploring the counter-trend strategy

13 Bibliography:

Hayes, Adam. "Technical Analysis Definition." *Investopedia*, Investopedia, 7 May 2021, www.investopedia.com/terms/t/technicalanalysis.asp.

(Triennial Central Bank Survey, Foreign exchange turnover, Monetary and Economic Department ,April 2019) https://www.bis.org/statistics/rpfx19_fx.pdf

The Ai Index Report – Artificial Intelligence Index
https://aiindex.stanford.edu/report/: https://aiindex.stanford.edu/report/

(2018, Barclay Hedge AI/Machine Learning in Investment Strategies) - https://www.barclayhedge.com/insider/majority-of-hedge-fund-pros-use-ai-machine-learning-in-investment-strategies

(Global financial markets liquidity study, August 2015): https://www.pwc.com/gx/en/financial-services/publications/assets/global-financial-market-liquidity-study.pdf

(Statista research department, 2016) Avergae Trade Length Forex Retail Traders in the Uk 2015-2016 Raynor Best - https://www.statista.com/statistics/655999/average-trade-length-in-forex-retail-trading-united-kingdom/

(counter trend strategy): https://www.quantstart.com/articles/what-are-the-different-types-of-quant-funds

Hhistorical Forex data source: https://www.histdata.com/download-free-forex-data/

(data/technical indicators/ MACD): Appel, Gerald (2005). *Technical Analysis Power Tools for Active Investors*. Financial Times Prentice Hall. p. 166. ISBN 0-13-147902-4.

(1997 , Sepp Hochreiter et Al.): https://www.bioinf.jku.at/publications/older/2604.pdf

(2020, Mehtab et al.): https://arxiv.org/ftp/arxiv/papers/2011/2011.08011.pdf

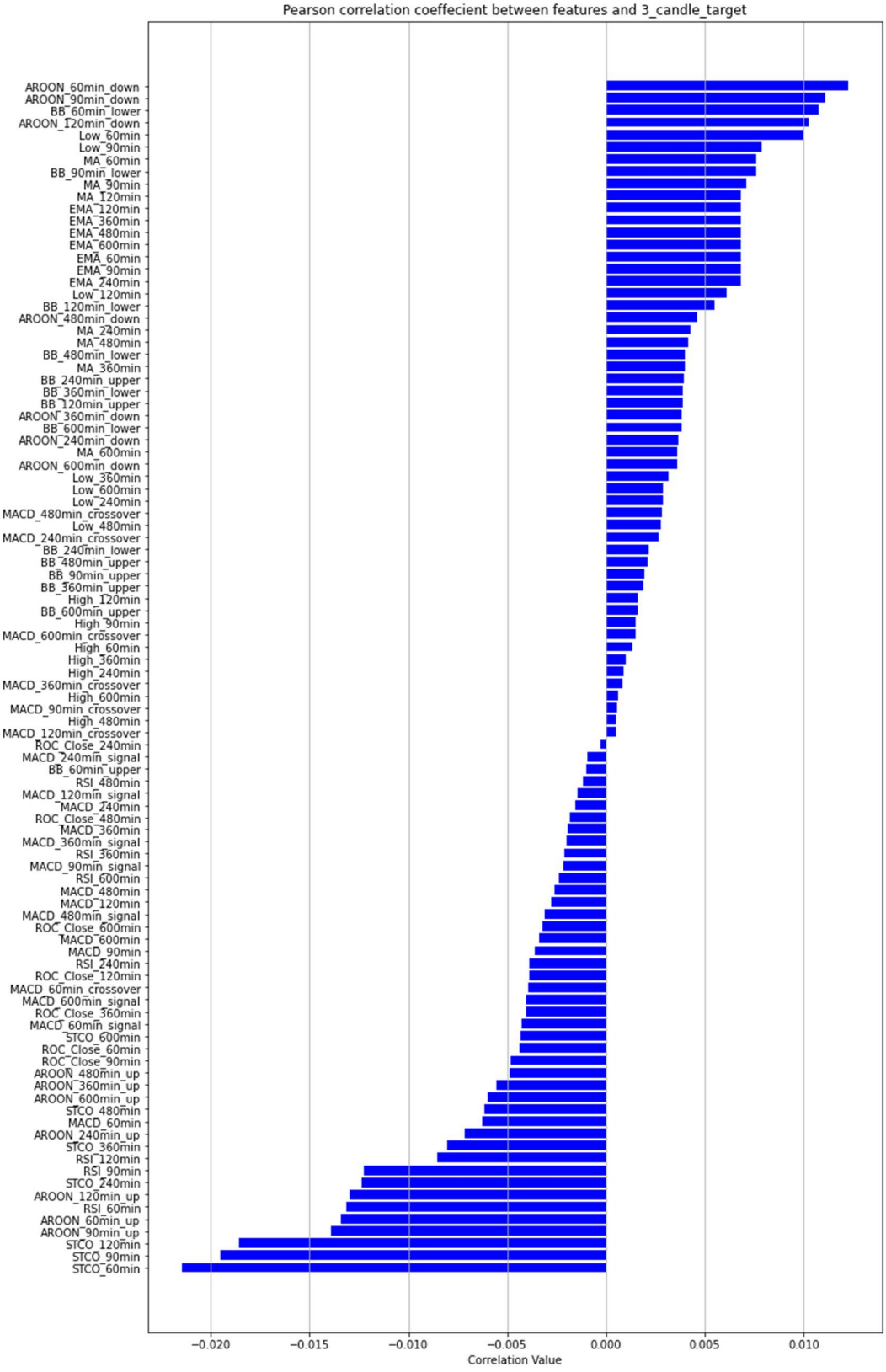
(2017, Cooijmans) Recurrent Batch Normalization: https://arxiv.org/abs/1603.09025

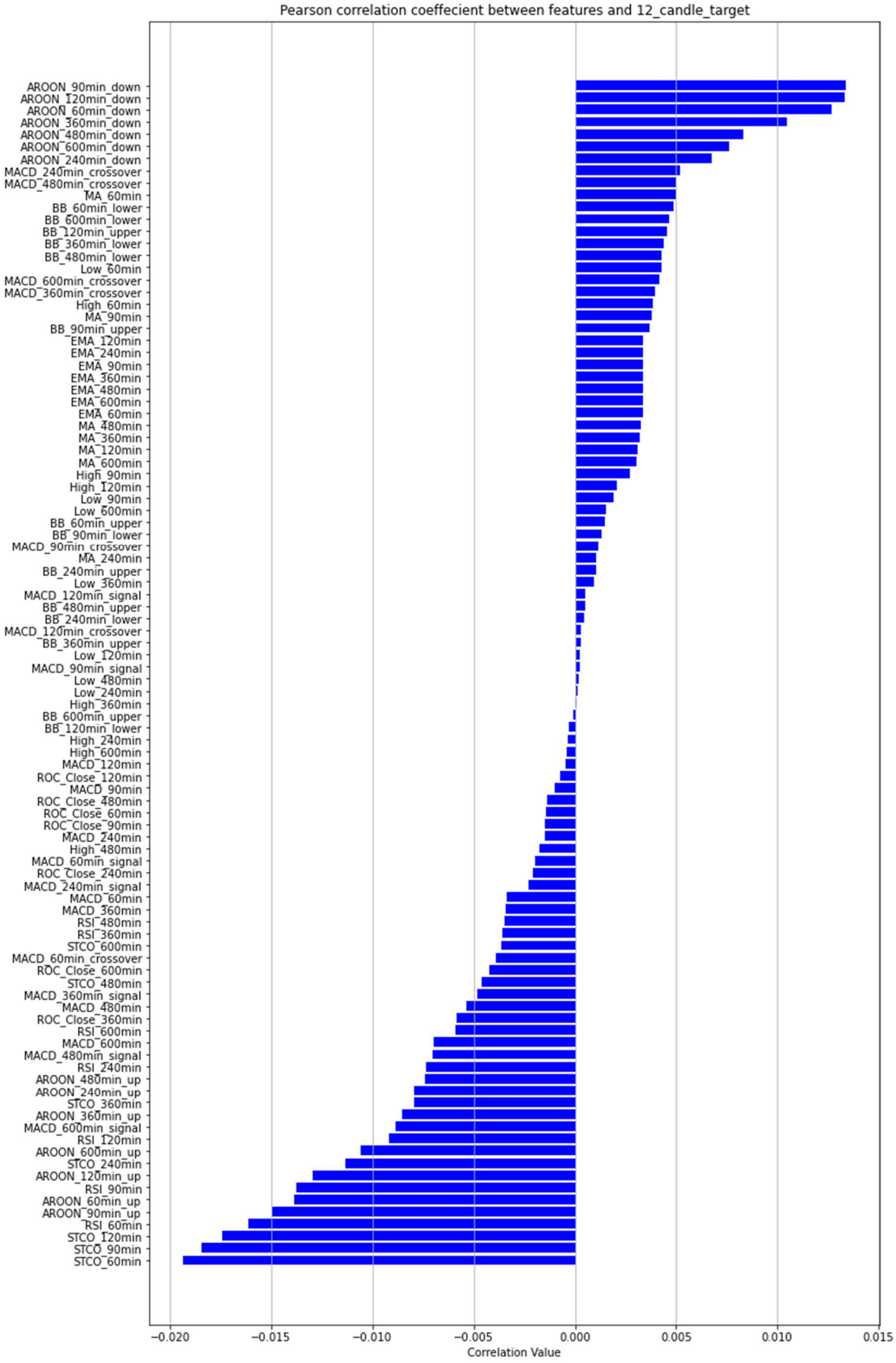
(2012, Bergstra et al.) : https://jaywhang.com/assets/batchnorm_rnn.pdf

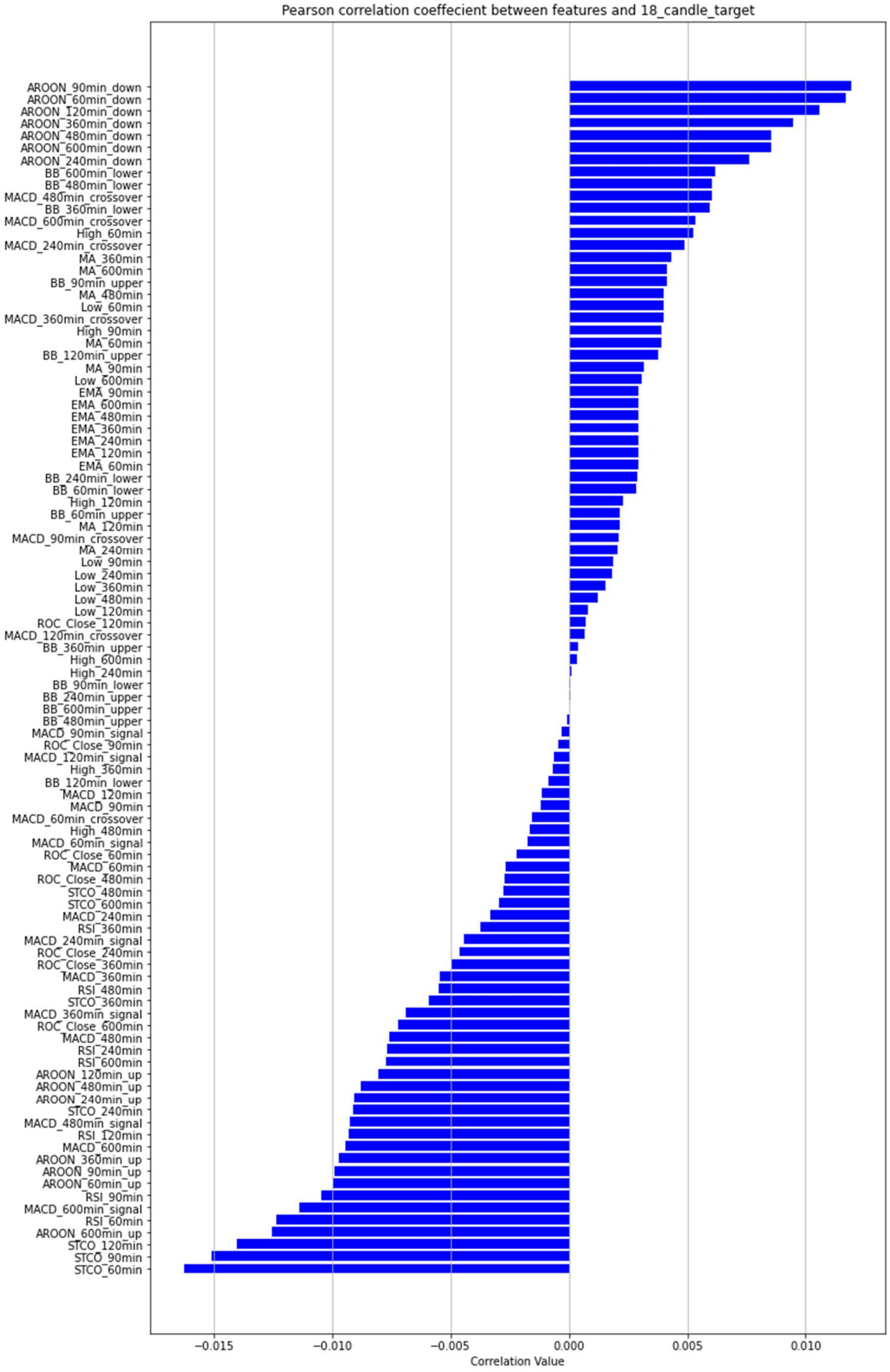
: https://www.jmlr.org/papers/volume13/bergstra12a/bergstra12a.pdf

14 Appendix

14.1 Correlation of technical indictors for other future candles




Pearson correlation coeffecient between features and 12_candle_target
AROON_90min_down
AROON_120min_down
AROON_60min_down
AROON_360min_down
AROON_480min_down
AROON_600min_down
AROON_240min_down
MACD_240min_crossover
MACD_480min_crossover
MA_60min
BB_60min_lower
BB_600min_lower
BB_120min_upper
BB_360min_lower
BB_480min_lower
Low_60min
MACD_600min_crossover
MACD_360min_crossover
High_60min
MA_90min
BB_90min_upper
EMA_120min
EMA_240min
EMA_90min
EMA_360min
EMA_480min
EMA_600min
EMA_60min
MA_480min
MA_360min
MA_120min
MA_600min
High_90min
High_120min
Low_90min
Low_600min
BB_60min_upper
BB_90min_lower
MACD_90min_crossover
MA_240min
BB_240min_upper
Low_360min
MACD_120min_signal
BB_480min_upper
BB_240min_lower
MACD_120min_crossover
BB_360min_upper
Low_120min
MACD_90min_signal
Low_480min
Low_240min
High_360min
BB_600min_upper
BB_120min_lower
High_240min
High_600min
MACD_120min
ROC_Close_120min
MACD_90min
ROC_Close_480min
ROC_Close_60min
ROC_Close_90min
MACD_240min
High_480min
MACD_60min_signal
ROC_Close_240min
MACD_240min_signal
MACD_60min
MACD_360min
RSI_480min
RSI_360min
STCO_600min
MACD_60min_crossover
ROC_Close_600min
STCO_480min
MACD_360min_signal
MACD_480min
ROC_Close_360min
RSI_600min
MACD_600min
MACD_480min_signal
RSI_240min
AROON_480min_up
AROON_240min_up
STCO_360min
AROON_360min_up
MACD_600min_signal
RSI_120min
AROON_600min_up
STCO_240min
AROON_120min_up
RSI_90min
AROON_60min_up
AROON_90min_up
RSI_60min
STCO_120min
STCO_90min
STCO_60min
−0.020
−0.015
−0.010
−0.005
0.000
0.005
0.010
0.015
Correlation Value


Pearson correlation coeffecient between features and 18_candle_target
AROON_90min_down
AROON_60min_down
AROON_120min_down
AROON_360min_down
AROON_480min_down
AROON_600min_down
AROON_240min_down
BB_600min_lower
BB_480min_lower
MACD_480min_crossover
BB_360min_lower
MACD_600min_crossover
High_60min
MACD_240min_crossover
MA_360min
MA_600min
BB_90min_upper
MA_480min
Low_60min
MACD_360min_crossover
High_90min
MA_60min
BB_120min_upper
MA_90min
Low_600min
EMA_90min
EMA_600min
EMA_480min
EMA_360min
EMA_240min
EMA_120min
EMA_60min
BB_240min_lower
BB_60min_lower
High_120min
BB_60min_upper
MA_120min
MACD_90min_crossover
MA_240min
Low_90min
Low_240min
Low_360min
Low_480min
Low_120min
ROC_Close_120min
MACD_120min_crossover
BB_360min_upper
High_600min
High_240min
BB_90min_lower
BB_240min_upper
BB_600min_upper
BB_480min_upper
MACD_90min_signal
ROC_Close_90min
MACD_120min_signal
High_360min
BB_120min_lower
MACD_120min
MACD_90min
MACD_60min_crossover
High_480min
MACD_60min_signal
ROC_Close_60min
MACD_60min
ROC_Close_480min
STCO_480min
STCO_600min
MACD_240min
RSI_360min
MACD_240min_signal
ROC_Close_240min
ROC_Close_360min
MACD_360min
RSI_480min
STCO_360min
MACD_360min_signal
ROC_Close_600min
MACD_480min
RSI_240min
RSI_600min
AROON_120min_up
AROON_480min_up
AROON_240min_up
STCO_240min
MACD_480min_signal
RSI_120min
MACD_600min
AROON_360min_up
AROON_90min_up
AROON_60min_up
RSI_90min
MACD_600min_signal
RSI_60min
AROON_600min_up
STCO_120min
STCO_90min
STCO_60min
−0.015
−0.010
−0.005
0.000
0.005
0.010
Correlation Value