The comparison between machine learning strategies and mathematical strategies for a trading bot on the cryptocurrency market

Reece Cripps, MSc

Submitted to The School of Computer Science, University of Nottingham  
in September 2022  
in partial fulfilment of the conditions for the award of the degree of   
Master of Science in Computer Science with Artificial Intelligence

# Text, letter Description automatically generated

**09/08/2022**

# Contents

[2](#_Toc110857239)

[Contents 3](#_Toc110857240)

[Abstract 4](#_Toc110857241)

[Acknowledgements 5](#_Toc110857242)

[Introduction 6](#_Toc110857243)

[1.1 Project Aims 6](#_Toc110857244)

[1.2 Requirements/Objectives 6](#_Toc110857245)

[1.3 Background into Cryptocurrency 8](#_Toc110857246)

[Literature Review 9](#_Toc110857247)

[2.1 Technologies and Libraries 9](#_Toc110857250)

[2.2 Research papers 10](#_Toc110857255)

[2.3 Existing Trading Bots 19](#_Toc110857265)

[Design and Implementation 21](#_Toc110857271)

[3.1 Software Design 21](#_Toc110857274)

[3.1.1 Graphical User Interface 21](#_Toc110857275)

[3.1.2 API utilisation 22](#_Toc110857276)

[3.2 Machine Learning Strategies 23](#_Toc110857277)

[3.2.1 Support Vector Machines 23](#_Toc110857278)

[3.2.2 Artificial Neural Networks 25](#_Toc110857279)

[3.3 Mathematical Strategies 27](#_Toc110857280)

[3.3.1 Moving Average 27](#_Toc110857281)

[3.3.2 Relative Strength Index 29](#_Toc110857282)

[Results and Discussion 31](#_Toc110857283)

[4.1 Machine Learning Results 31](#_Toc110857285)

[4.2 Mathematical Results 37](#_Toc110857289)

[4.3 Combined Evaluation 41](#_Toc110857290)

[4.4 Did I meet my Aims/Objectives? 44](#_Toc110857294)

[Critical Appraisal 45](#_Toc110857295)

[5.1 What went well 45](#_Toc110857297)

[5.2 What went badly 46](#_Toc110857298)

[5.3 What I would have done differently 46](#_Toc110857299)

[5.4 Future Development 47](#_Toc110857300)

[Conclusion 48](#_Toc110857301)

[References 49](#_Toc110857302)

[Appendix A: Supplementary Materials I 51](#_Toc110857303)

# Abstract

Cryptocurrency has taken off during the past couple of years, with popularity soaring throughout the globe. The amount of money flowing into cryptocurrency reached $3 Trillion making it one of the biggest industries in the world. The stock market has been around for a long time and there are a lot of tools out there that businesses use to maximise their returns. However, with cryptocurrency being new and more volatile, the tools available are scarce or not as accurate as they could be. In this project I have researched a variety of different methods to compare and evaluate the most accurate method at trading and predicting the price of the top crypto assets. With this project being a regression problem, the accuracy of the models can vary. Some machine learning methods have not been considered as they are not necessarily known for their effectiveness at solving regression problems. I have also taken some of the most popular mathematical indicators. These were used as a comparison to the machine learning methods and all the methods were compared to holding the asset for the duration of the run. I found that the machine learning techniques massively outperformed the mathematical ones regarding previous data. However, in the live demos the results varied with the machine learning techniques mostly performing better but the moving average trading bot was fairly close behind them in terms of profits made.

Keywords: Support Vector Machine (SVM); Artificial Neural Network (ANN); Moving Average (MA); Relative Strength Index (RSI); Machine Learning; Cryptocurrency; Trading Bots

# Acknowledgements

I would like to thank my supervisor Colin Johnson for being very supportive along the journey through completing this project. I would also like to thank my friends and family for helping me through the tough times and keeping me motivated throughout.

# Introduction

## Project Aims

The **Main Aim** of my project is to explore different strategies and apply them to the cryptocurrency market. This tool will be useful for researchers and academics because they can see a breakdown of a variety of different methods, both mathematical and machine learning. There is also a clear comparison between these methods to see which is the best overall.

The different trading bots will be tested on multiple scenarios:

* Past – 2018-2021
* Recent – 2021-2022
* Real-time – Current time of testing (July/August 2022).

For these scenarios they have been tested on multiple different coins as well. I have tested the top 2 coins (Bitcoin and Ethereum) as well as some less popular coins, LoopRing and Litecoin. These other 2 coins are still in the top fifty cryptocurrency assets because venturing further away from the popular coins induces more volatility which becomes extremely difficult to predict.

For each trading bot, users can see the breakdown of the data in graph form and also able to see the bots trading manoeuvres via the bottom of the screen. The bots trade at their own pace, buying and selling positions.

Data is then exported to an excel file where it can be analysed and compared. This will be a lot easier than looking at the graph and text output displayed by the program.

## Requirements/Objectives

Looking at the project aims, there are a number of different requirements and objectives that needs to be fulfilled.

**Simple and effective UI:**

For this project I aim for the UI to be simple but effective. This means that any users that wish to utilise my program will find it easy to use and not too confusing. For example, I do not want to bombard users with constant text printing out to the user but instead represent a graph format with a simple text view at the bottom of the screen. The main page of the program will also have a menu which contains all the bots so that users can click on the bot they wish to use.

**Mathematical trading strategies:**

There are a lot of mathematical strategies available to test and use. I wanted to take some of the most well-known strategies and compare them to machine learning strategies. So, for this, I used a number of different recommendations to pick out the best strategies to use [1]. The moving average strategy and Relative strength index strategy provide good indicators as to how the market is moving. A more in-depth explanation can be found in the implementation section (3.2 – Mathematical strategies).

**Machine Learning strategies:**

Machine learning is becoming more popular and the technology available makes creating machine learning programs more accessible to the average person. Creating machine learning strategies could be better than traditional mathematical strategies because of the computing power available. For this project I intend to use 2 Machine learning strategies: Support Vector Machines and Artificial Neural Networks. If time permits, I will also be creating a regression tree. How these strategies work will be explained in the implementation section (3.3 – Machine learning strategies).

**Multiple trading scenarios:**

In the current market environment, I think that it would be an unfair representation of the trading strategies. To grasp a better understanding of the methods, implementing multiple trading scenarios will help me understand how these strategies work and what scenarios they excel in.

For this project I have chosen to look at data from 2018-2021. This is because 2018 was when the popularity for cryptocurrency took off [Fig 1]. For the Machine learning strategies training data will be 2018 – 2020 and then test data will be 2021. For the mathematical strategies, they will just be tested on 2021 data.

The second trading scenario will be 2021-2022 where the machine learning bots will be trained on 2021 data and then tested on 2022 (Jan – July). The mathematical strategies will be tested on 2022 (Jan – July)

Finally, the trading bots will be tested in real-time to see how they perform. For this the machine learning strategies will be trained on 2022- Jan – July data (at 4-hour intervals) and then tested in real-time (1 day before current time – current time and left running for 2 days).



Figure 1 – Cryptocurrency Market Capacity

Source: <https://coinmarketcap.com/charts/>

**Variety of crypto coins:**

For this project I intend to test multiple different coins. This is because the prices of the different coins depend on multiple factors, so getting a larger range of coin data will allow for a more even spread and a fairer result of the strategies. However, there is a common pattern between Bitcoin and the rest of the coins. If bitcoin is on a downtrend then other coins will mostly follow (see 1.3 – Background into the crypto Market).

**Data analysis:**

To be able to see which trading bots perform well, exporting all the data to an excel file will allow for better readability. I can also create graphs and formulae to allow for a more in-depth analysis of the data and results.

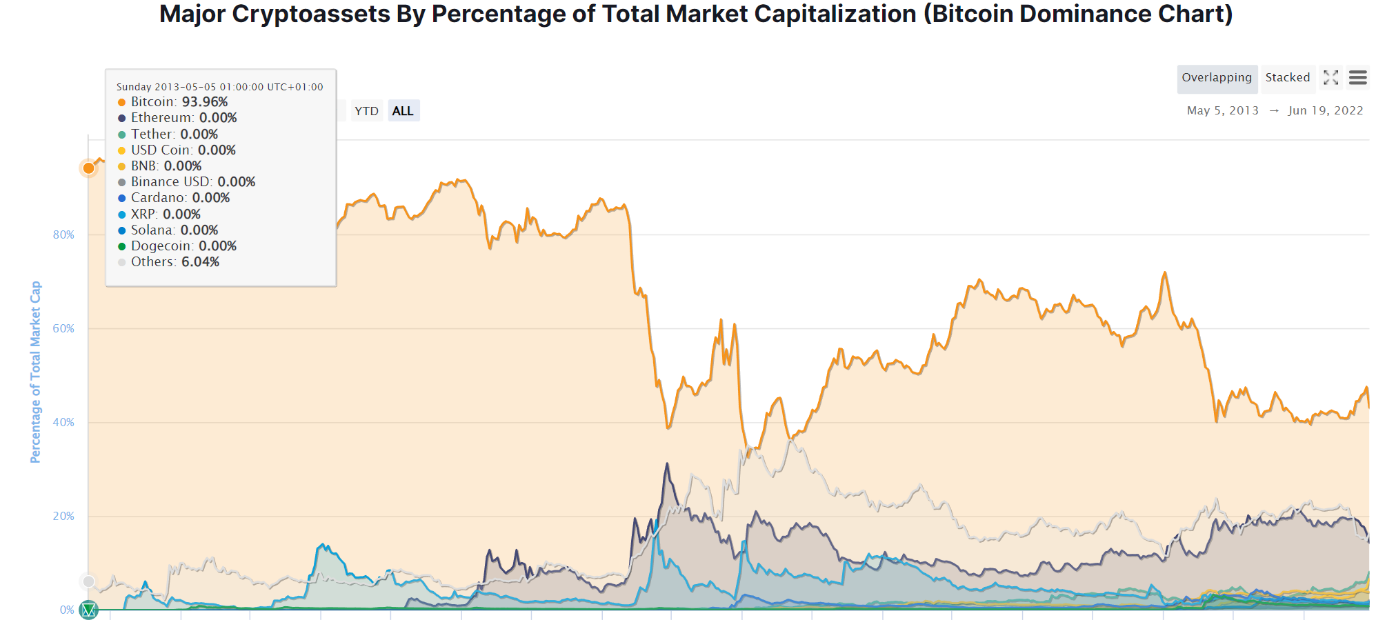
## Background into Cryptocurrency

Bitcoin started in 2008/2009 as the first blockchain network, created by Satoshi Nakamoto [3]. To collect Bitcoin, you had to mine it with a complicated algorithm. it first became tradable back in 2009/2010 being worth less than $0.14. The first physical transaction with Bitcoin was made in May 2010 where 2 pizzas were purchased using 10,000 Bitcoin. At Bitcoin’s all time high that would be worth roughly $680 Million.

In 2013/2014 the popularity of cryptocurrencies grew. More coins were being developed and the popularity of holding assets rose sharply. [Fig 2] shows the dominance of Bitcoin in the market. Back in 2013; Bitcoin held ~94% of the market. Whereas today, Bitcoin only accounts for 43%. Whilst this is still a huge portion of the market, things have changed drastically over the years.

Figure 2 – Bitcoin dominance

Source: <https://coinmarketcap.com/charts/>



This dominance means that Bitcoin tends to control the market prices. If Bitcoin is on a downward trend then the other coins will follow [4]. Another reason why Bitcoin impacts other coins’ price is due to the fact that other coins cannot normally be purchased or sold with fiat currency (like USD or GBP) instead, these coins have to first be traded to and from Bitcoin and then into their fiat currency meaning that Bitcoin acts as a middleman in these transactions making it more important.

Looking at [Fig 1] You can see how the global market compares today. At the back end of 2021, the total amount of money in crypto currency was around $3 Trillion which is vastly bigger than 2013’s $94 Million. Looking at the market today, it is in a downward trend and many coin prices are falling. This means that the strategies that I am using might not be very effective when testing the real-time trading. Therefore, I am using these strategies on a variety of scenarios to collect multiple data points and allow for a fairer analysis of the strategies.

# Literature Review



## Technologies and Libraries

In this section I will describe the background research done to find relevant libraries and technologies that I have used within my project.

### [Market](https://binance-docs.github.io/apidocs/spot/en/" \l "change-log) data API

The CoinAPI.io is an efficient and effective API that includes all of the endpoints that I will be needing for implementation [5]. Most other APIs available are either paid for or don’t offer enough depth in their historical data.

The endpoint that I will be using the most is:

GET v1/ohlcv/{symbol\_id}/history?period\_id={period\_id}&time\_start={time\_start}&time\_end={time\_end}&limit={limit}

This endpoint allows OHLCV (open, high, low, close and volume) data to be collected with a symbol\_id (crypto coin) and the period for the data points (1 day, 1 month, 1 year…) as well as the start and end time for the data points.

For example, 'https://rest.coinapi.io/v1/ohlcv/COINBASE \_SPOT\_BTC\_USD/history?period\_id=1DAY&time\_start=2018-01-01T00:00:00&time\_end=2020-12-31T00:00:00&limit=2000'

Will allow me to collect data on a daily basis between 2018 to 2020 for the value of bitcoin to USD on the Coinbase market.

### Pandas/NumPy/ Xlsxwriter

Pandas and NumPy are libraries useful for data processing in python. These two libraries work hand in hand for manipulation of data. I have used these libraries to compress the data collected from the API endpoints.

Xlsxwriter allows the use of writing data from python to an excel file. For my project I have taken all the data from each bot and written this to an excel file to allow better analysis of the data rather than looking at command line prints.

Xlsxwriter also works well with the Pandas library so coupling these together is beneficial for data analysis.

### TensorFlow/SciKit

TensorFlow allows a simple creation of a neural network. For my project the neural network will be used to predict prices of the crypto coins. An in-depth description of how this works can be found in 3.3 – Machine Learning strategies.

SciKit allows the creation of the support vector machine model. This will be used as my second machine learning strategy. More about how this works can be found in 3.3 – Machine Learning Strategies.

### Matplotlib

To display the market data Matplotlib will be used as a visually appealing way to display data to the users.

## Research papers

2.2.1 and 2.2.2 are my own work cited from [7].

### Crypto Price prediction using SVMs

Nor Azizah Hitam, Amelia Ritahani Ismail, and Faisal Saeed used an optimised support vector machine (SVM) based on particle swarm optimisation (PSO) to predict the price of cryptocurrencies [8].

Not only did they use Optimised SVM-PSO, but they also used a variety of other machine learning concepts as a comparison to their model. They used these models to test their predictive accuracy of 6 cryptos (Bitcoin, Ethereum, Litecoin, Nem, Ripple and Stellar).

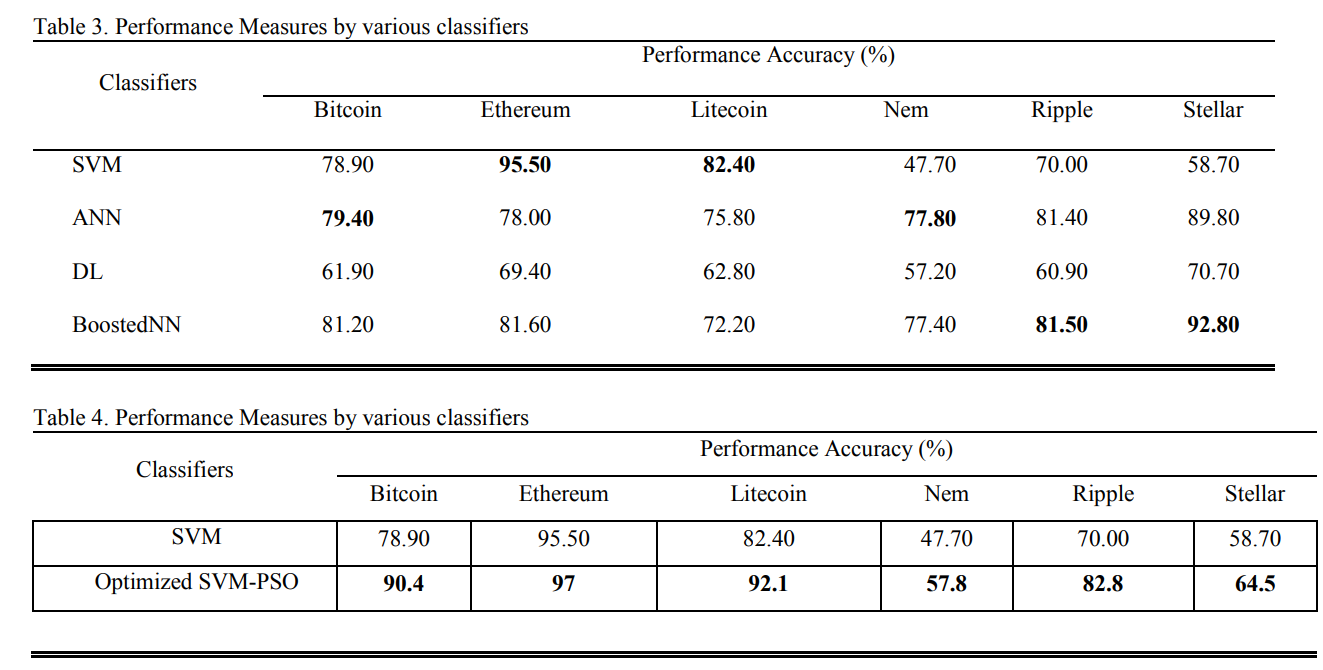
The researchers designed a new algorithm to test their method against all other existing methods. They used a collection of daily price charts for 5 years from 2013 to 2018. For each classifier they had a training set from 2013 to 2017 and then tested the models on the data from 2017 to 2018.

They used quantitative methods to obtain results that are relevant to the paper. In this scenario, quantitative methods are far more appropriate because they needed to test how good their new method is at predicting prices. Qualitative methods would not have been able to provide mathematical evidence to prove their theory.

To back up their work they tested the accuracy of their model at predicting 364 days of daily price charts using classification. They found that Optimised SVM-PSO was superior at predicting 4/6 of the cryptocurrencies, with an accuracy of 97% for Ethereum.

These results back up their conclusion “The outcomes show that SVM-PSO outperformed other classifiers with the accuracy of 97%.” as their table [Fig.**3**] provides significant evidence of the performance of each model tested. The researchers could have also compared the prediction speed of each model to show which model is efficient as well as accurate.

Fig 3 - Results table from [8]



### Deep Learning-based Cryptocurrency Price Prediction

In this paper, the researchers apply a hybrid Long short-term memory (LSTM) and Gated Recurrent Unit (GRU) model to 2 cryptocurrencies (Litecoin and Monero) to predict their price [9].

These researchers are creating a new model that has not been tested before. They give a breakdown of all the methods used and their corresponding accuracy as well as the year it was created.

They collected their daily data and then performed data pre-processing. The data they used was ranged from August 2016 – February 2020 for Litecoin (1279 data points) and January 2015 – February 2020 for Monero (1851 data points). After performing data pre-processing, the researchers split their data into testing and training. An image of their proposed model is shown in [Fig. **4**], they combine 2 LSTM models with a single GRU model to get a price prediction.

The researchers used quantitative methods to test their designed algorithm. They also compared their model to the standard LSTM model and provided a graph of this comparison. They tested the algorithms across different windows (1-day, 3-day and 7-day prediction) and created a table to breakdown their results. Quantitative methods are more appropriate for this paper because the researchers needed to know how accurate their method is in terms of numbers so they are easily able to compare this to other models that are available.

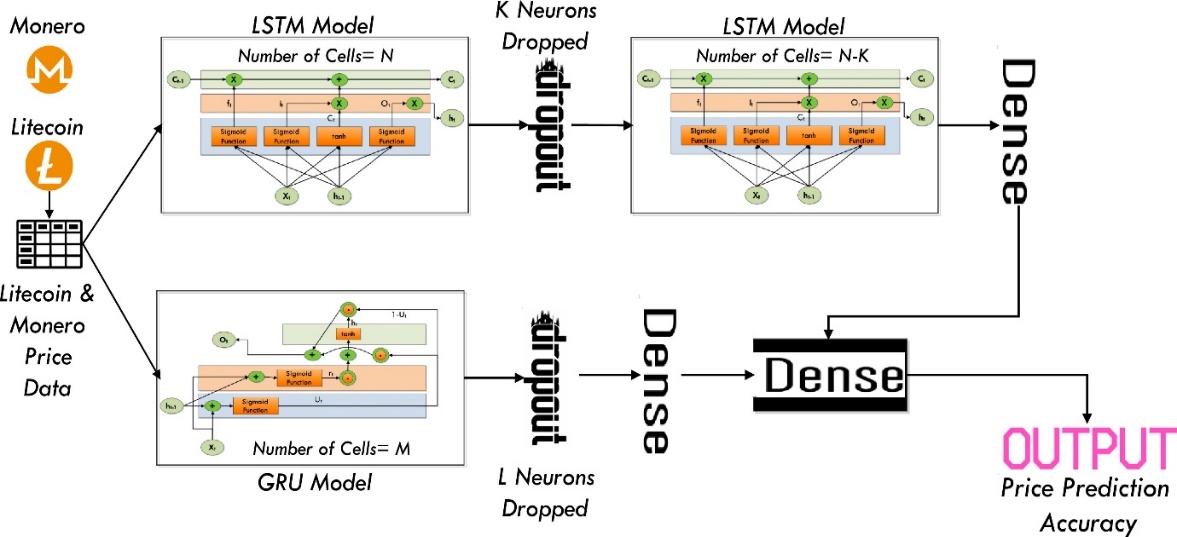


Fig 4 – Proposed model from [9]

The researchers compared the accuracy of their model to other models available. They collected a variety of accuracy variables, such as: Mean Squared Error, Root Mean Squared Error, Mean Absolute Error and Mean Absolute Percentage Error (MSE, RMSE, MAE and MAPE) to evaluate their algorithm. They then used graphs to plot these error values to compare the standard LSTM model with their hybrid model. Their error values proved to be a lot lower than the other model they were comparing against which supports their findings. They concluded with “Our proposed scheme has proved to be better than the LSTM network, which is evident from the errors of prediction.” But they also stated that there were a large number of challenges for future researchers to delve deeper into.

### Stochastic Neural Networks for Cryptocurrency Price Prediction

In 2020, researchers published a paper exploring price prediction for cryptocurrency using stochastic neural networks[10]. These researchers used a stochastic multi-layer perceptron and long short-term memory (LSTM) which is like 2.2.2 where they used an LSTM model when predicting Litecoin and Monero.

These researchers took a variety of different data when compiling their models. Including the daily low, high and trading volume. Along with social sentiment information such as google trends and tweet volumes. They then used 850 data points from 2017 to 2019 to help train their model. After collecting their data, they computed data pre-processing to normalise all their data. An image of their model can be found in [Fig.5].

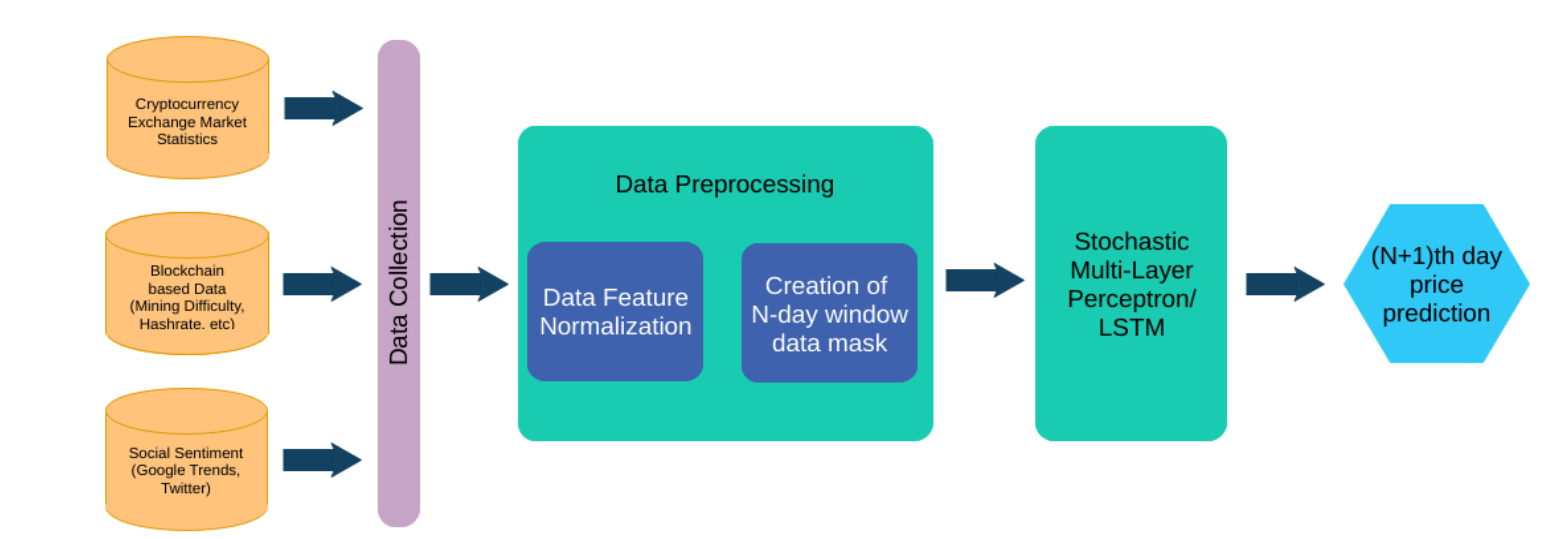


Fig 5 –model from [10]

After collecting their results, the researchers used a variety of evaluation metrics, like 2.2.2 which was an effective way to see if their model was accurate or not. The Mean Absolute Percentage Error (MAPE), Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and Mean Squared Error (MSE) were used. Their results showed that stochastic neural networks were slightly better than deterministic models.

This was a highly effective paper that provided a lot of insight as to how to go about comparing a variety of different models. There are a lot of strengths in this paper that I have been able to take on board when it comes to evaluating my own dataset.

### The Recurrent Reinforcement Learning Crypto Agent

Researchers from University College London used a recurrent reinforcement agent to trade bitcoin VS USD on a perpetual swap contract [11]. This was done over a period of 5 years and the agent returned a total of 350% profit.

The researchers used transfer learning, an echo state network and a target model using the Sharpe ratio to train their reinforcement agent to learn to trade digital assets using futures.

Futures trading is where you either short (predict that the price of the asset will go down) or long (predict that the price will increase over time).

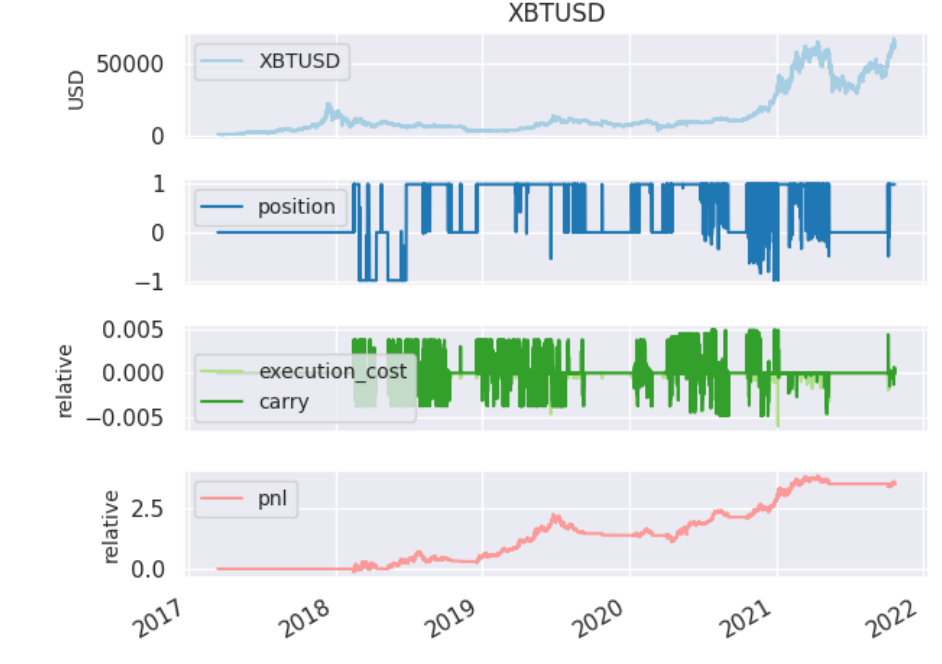


Fig 6 –results from [11]

Looking at [Fig 6] you can see how well the agent performed. Something that I really like about this paper is in contrast to the rest of the research I have seen previously. Highlighted by the blue square is the profit that the agent gained towards the middle of 2019. This agent performed exceptionally well during the bear market which is very impressive that it was able to predict bitcoin to stay low and gain massive amounts of profit. This contrasts with other research available as most have been unsuccessful in predicting the price to fall.

### Cryptocurrency Trading Using Machine Learning

Thomas E. Koker and Dimitrios Koutmos went about researching cryptocurrency trading using a reinforcement-based machine learning approach [12]. For this, they traded the top 5 crypto coins at the time (Bitcoin, Ethereum, Litecoin, Ripple and Monero) and compared their model with a buy and hold approach.

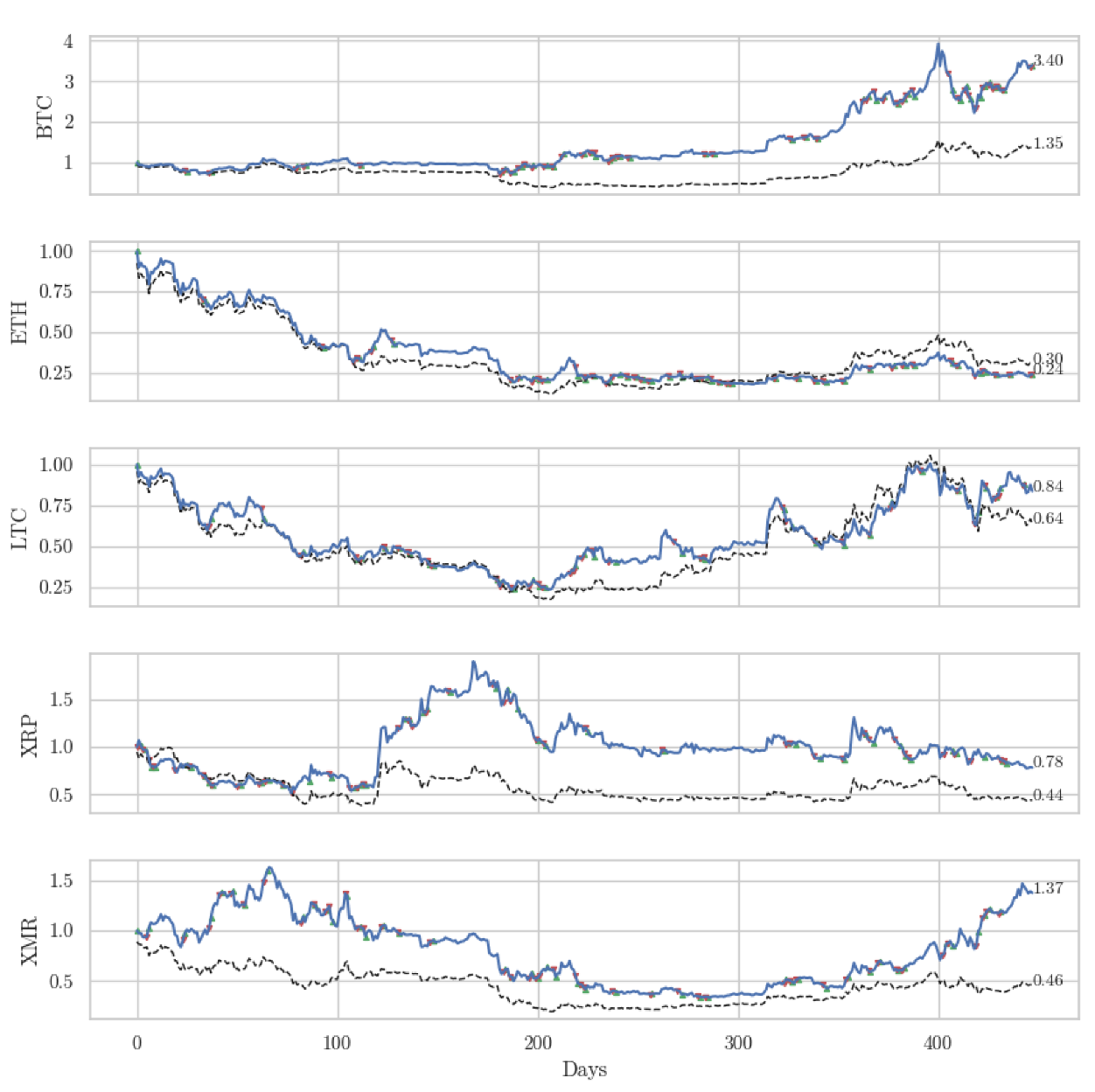
Thomas and Dimitrios trained their model for one thousand days and then simulated trading for a further one hundred days before letting their model trade for 447 days. Their model is based off of the buy-and-hold trading method but similar to 2.2.4 they used the Sharpe ratio to calculate the risk of returns.

Fig 7 –results from [12]

The results from this can be found in [Fig. 7] and they found that their model (blue line) significantly outperformed the buy-and-hold strategy (black line). Whilst this paper is an interesting read, I was hoping that the researchers would give some comparisons to other machine learning methods. The researcher’s paper isn’t very in depth, but it provides some good insights as to how their model compares with the buy-and-hold strategy. I would have been more insightful if they had better metrics for comparing their model.

### Using machine learning for cryptocurrency trading

Jifeng Sun, Yi Zhou and Jianwu Lin used random forest to predict a variety of cryptocurrencies’ prices [13]. These researchers compared a variety of different market data. Using 10 second intervals and 10-minute intervals for their data points.

They found that 10-minute intervals were more effective than the 10 second data.

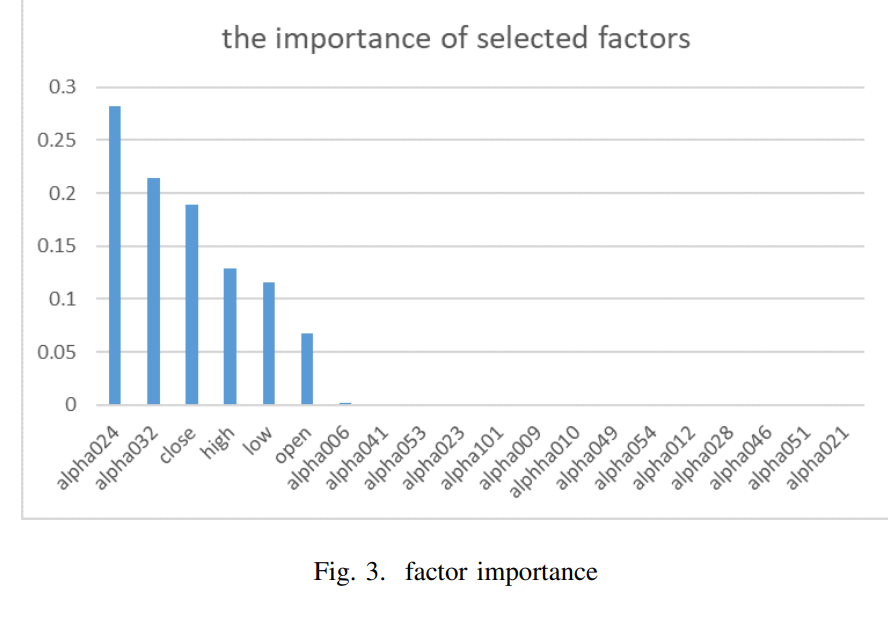


Fig 8 –Factor importance [13]

[Fig 8] shows the importance of the factors used in the researchers’ machine learning models. They found that the close, high, low and open data proved to be very important. This is useful for my research because my models will be using this data as an input into the machine learning methods that I will be testing.

This paper provided some interesting research into random forests and price prediction. The accuracy of their models reached 58% which is why I chose not to include decision trees as part of my machine learning models. However, this paper does a good job at showing which factors are important for the machine learning models.

### Is smarter better? : a comparison of adaptive and simple moving average trading strategies

In 2005, Craig A.EllisSimon and A.Parbery researched the moving average strategy and compared this to buying and holding 3 stocks in the stock market [14]. They compared the adaptive moving average (AMA), simple moving average (SMA) and the Buy and hold method.

Table

Description automatically generated with medium confidence

Fig 9 –Results [14]

[Fig 9] shows the results of the experiment. Throughout the research the moving average strategies outperformed the buy and hold strategy in terms of gross return. However, the moving average strategy is dependent on the market fluctuating because if the market was just to rise in price then the buy and hold strategy is more effective.

This paper shows how impactful the moving average strategy is. However, this paper was made in 2005 and the moving average strategy is under-researched, especially in more recent years. The researchers did a good job at displaying their results, but in the years that they carried out their tests the market was constantly on the rise which may invalidate some of their results.

### Prediction of cryptocurrency returns using machine learning

Graphical user interface, application, table, Excel

Description automatically generatedResearchers used a variety of machine learning techniques to analyse the top twelve most liquid cryptocurrencies. For this experiment, they used market data along with technical indicators as model features. The researchers used data from 2017-2018 to train their models. Similar to my project, the researchers have used the relative strength index and a variation of the moving average.

Fig 10 –Results [15]

[Fig 10] shows the average results from all 4 machine learning algorithms tested by the researchers. From the table we can see that the machine learning algorithms are more effective at the daily predictions rather than 15-minute intervals.

This paper did a really good job at explaining how all the machine learning methods were implemented. Furthermore, the researchers carried out a lot of different tests on a large range of cryptocurrencies especially in comparison to the rest of the papers that I have researched. The researchers also tested a wide range of machine learning models which gave me an insight as to which methods are better to use on this type of problem.

### Summary of what I’ve learnt from these papers

In summary, all the papers used quantitative methods to evaluate their algorithms. Most papers used accuracy as a measure of effectiveness for their algorithm. The papers show that the most effective way to evaluate a model is by using MSE, RMSE and MAE which is what I’ll be considering when evaluating my models. Looking at the papers, I can see that the better models were some variations of Support vector machines (SVM) and artificial neural networks (ANN).

From paper 2.2.6 I can see that decision trees are ineffective hence why I have not chosen to explore this model in my project.

|  |  |  |
| --- | --- | --- |
|  | Pros | Cons |
| ANNs | * Accurate at predicting regression. * Very flexible and can be applied to a large variety of problems. * If a few datapoints are missing the model can still provide accurate results. | * Many hyperparameters to set and test. * Very long training times. * Require many datapoints to create an accurate model. * Computationally expensive. * Might overfit on the training data used. |
| Regression Trees | * Easiest model to implement * Easy to visualise * O(depth) testing complexity | * Tends to overfit if pruning isn’t done correctly. * An exponential number of possible trees for each problem. |
| SVM | * Gaussian kernels allow for SVMs to excel in high dimensional scenarios. * Uses support vectors to create a margin which makes is computationally efficient for linear problems. * Good accuracy for regression problems | * Requires a good setting for the hyperparameters. * Struggles on larger data sets * Takes a longer time to train when computing regression problems. * SVMs are sensitive to noise. |

Fig 11 –Pros and cons table for machine learning methods

From these papers and other research, I can breakdown the machine learning methods with pros and cons [Fig 11]. This is useful for my project as I am now able to see which methods excel in certain areas and I will be able to apply this to my problem.

In terms of mathematical methods, this area seems to be under-researched as the only paper I was able to find was written in 2005. I will be taking their findings and applying them to the crypto market, and it will be interesting to see how the methods compare to machine learning methods.

## Existing Trading Bots

### Pionex

To see what the Pionex trading page [16] looks like please visit the appendix [Fig 1].

This trading bot has a very low fee of 0.05% per transaction which is low compared to the rest of the industry. As you can see from the figure, the page has a lot of potential bots that you can create to automatically trade a very large variety of cryptocurrency. Furthermore, there is a manual trading page that is useful for experienced traders who want to do this manually.

Whilst there is a large variety of bots available, there is only a small description as to how these work and their description is very vague. For example, “buy low and sell high” seems to just be setting values for the bot to buy and sell at, with very little intelligence at play.

So, for this bot page, there is very little intelligence being used and some of the bots seem quite complex to set up with little guidance.

### CoinRule

CoinRule is another trading bot that boasts a large variety of different trading techniques[17]. As you can see from appendix [fig 2] there are over 150 different trading strategies available for members. These methods are very in depth and explained exceptionally well. Some methods include the relative strength index (RSI) and the moving average (MA) which I will be taking advantage of in my project.

Their user interface is very friendly for beginners allowing If/then parameters to setup your bots. The premade templates are also useful for people who are unsure on how to start.

Whilst this bot is very well made, the barriers to entry are high. The free account only allows a few premade strategies, a small number of bots and a small amount of trading volume. Upgrading to the highest tier will set you back over £400 a month.

### TradeSanta

TradeSanta is another popular crypto trading platform used by lots of people [18]. Looking at appendix [Fig 3] you can see what the home page of the trading bot page looks like. Something good about TradeSanta is that they have a tutorial which is very useful for new users. Furthermore, they also have a virtual trading bot which lets you experiment with fake money first before staking your own cash.

Their technical indicators also utilise the RSI and MA strategy along with a few other strategies, but a lot less compared to the other trading bots available.

### Cryptohopper

Cryptohopper is the last trading bot platform that I will be looking at [19]. There are a lot more out there, but these are just the most popular bots that I have found amongst the community. Cryptohopper is the only trading bot that states they use AI to automate their trading.

Appendix [Fig4] shows the landing page of cryptohopper. When first logging in you are greeted with a tutorial that takes you through all of the different tabs and documentation that will guide you. If creating a free account, you are very limited in terms of bots you can setup. There are only a few strategies available.

They only allow you to use the AI-powered strategy if you sign up to their most expensive monthly subscription. So, I am unsure as to what AI method they are using to back test their bots.

### Summary of what I’ve learnt from these trading bots

From these trading bots I have learnt a lot of useful information with regards to different trading strategies and user interface available to the users of the platform. It seems like a lot of these platforms offer a wide range of strategies and ease of use when setting up the bots.

However, most of the platforms require payment to get the most out of their bots. The free tiers offer extraordinarily little for consumers with minimal strategies and volume limitations.

Taking these details into account when looking at my project, I will be trying to implement a user-friendly interface that is not confusing for beginners. Furthermore, I have seen a large variety of mathematical trading indicators that I had previously researched. This backs up my research and I will be using the RSI and MA methods when implementing my mathematical trading strategies.

These methods seem to be well known in the community and very popular. So, comparing these methods to machine learning strategies will be a good indication as to how they perform.

# Design and Implementation



## Software Design

In this section I will talk about the different software choices that I have made to make the program easier to follow and understand.

### Graphical User Interface

The program consists of a home page that lets the user switch between the time periods that I have programmed (2018-2021, 2021-2022, Live) Once they have chosen their desired time frame there are options for each specific bot that they are able to open. This will load a new page which will display some different details.

Looking at figure 12 you can see what the home page looks like. This shows clearly how the user should interact with the program, there is a clear dropdown box that allows the user to change the time period. Furthermore, I have colour coded the buttons so that purple indicates the machine learning techniques and red indicates the mathematical techniques.

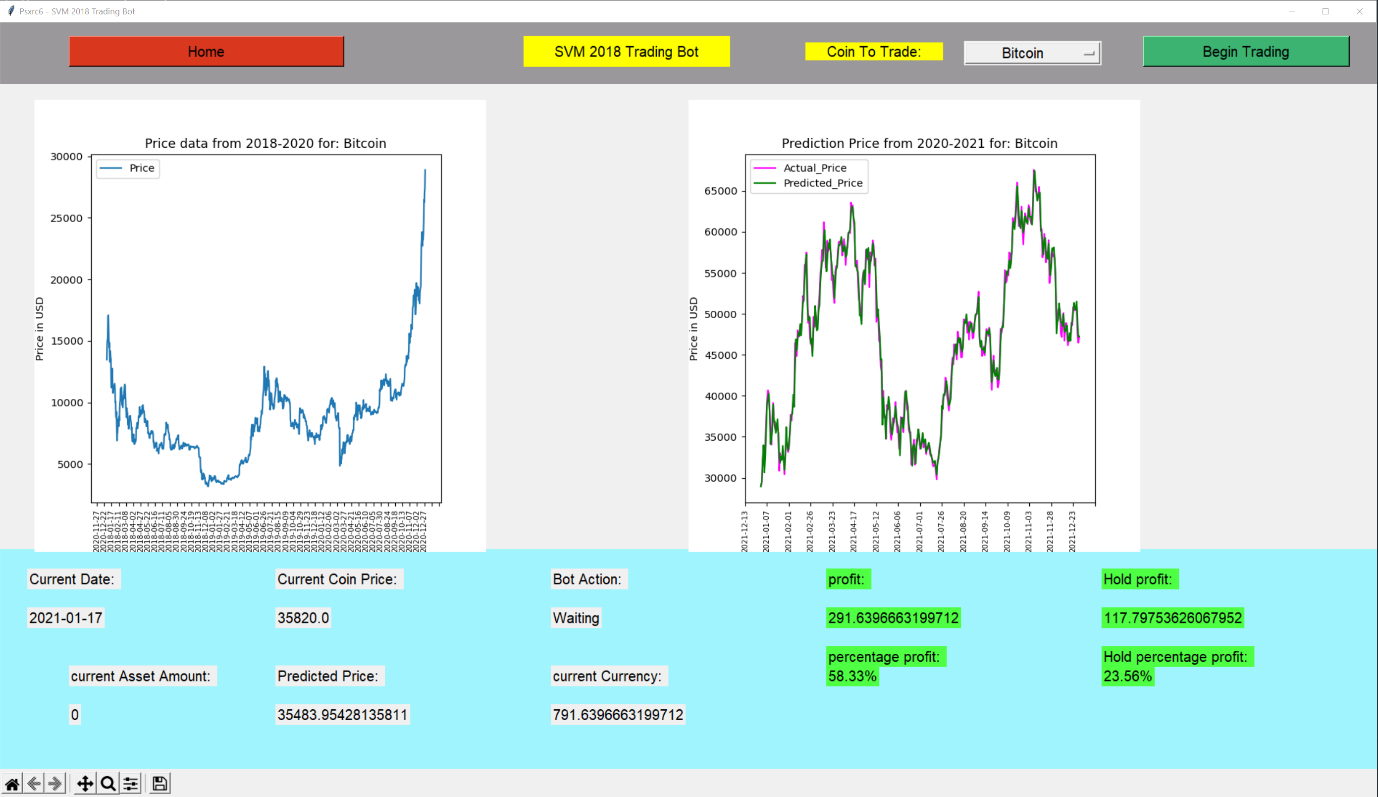
Fig 12 – Program home page



Figure 13 shows an example of the bot page. The top is a navigation bar that lets the user go back to the home page, change the coin they wish to test using a dropdown or begin trading with the green button in the tope right. The left-hand side is a graph of the training data, this lets the user see what data was used to train the trading bot. The right-hand side is the test data graph and then the bot’s predictions over the top of the test data. The bottom of the screen displays details of how the bot is trading.

This design is super simple and effective, it displays all the necessary details needed to understand what is happening and also lets the user clearly see the bot’s predictions.

Fig 13 – Bot trading page



### API utilisation

To collect data about each coin and its market history, an API was used. This allowed me to collect large amounts of data efficiently and effectively.

I was able to manipulate the data and store it into an excel file so that when the program was run it was not always calling the API for the same data. However, the live bots call an API request every time they are run as the dates will change. Also, for the live data, code was used to calculate the closest time to the next datapoint so that no errors were created when creating the API request [Fig14].

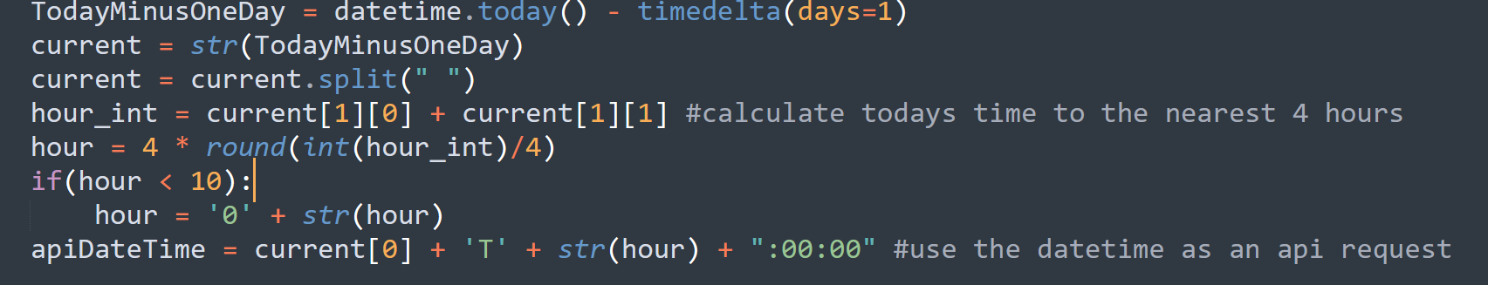


Fig 14 – Date code

Combine all the variables to get a string API request

Round the hour to the nearest 4 as this is what the data point intervals were

Appendix [Fig 5] shows the code used to retrieve the data. This uses a nested for loop as the API returned a dictionary of rows and then each row was an array that had to be looped through and entered in to an excel file. I also split the date entries because there were unnecessary timestamps added to the datapoints.

When handling the data, I had to normalise the numerical values using a standard scalar, this allowed the machine learning techniques to learn more effectively. I also had to strip the dates when using test data as this was not possible for the machine learning techniques to process.

## Machine Learning Strategies

In this section I will talk about the different machine learning strategies I have implemented in my project. I will cover the basics as to how they work and talk about how they work within my program.

### Support Vector Machines

Support Vector Machines (SVM) are a type of supervised machine learning technique effective for both classification and regression.

In classification SVMs use a decision boundary (hyperplane) and takes some points as support vectors to work out whether a new point would belong to class A or class B. Looking at [fig 15]you can see how this works. There is a decision boundary and a margin for either side of the classes. Any point that falls inside that margin will become a support vector to ensure that new points are not misclassified.

The classification in [figure 15] is an example of a linearly separable dataset.

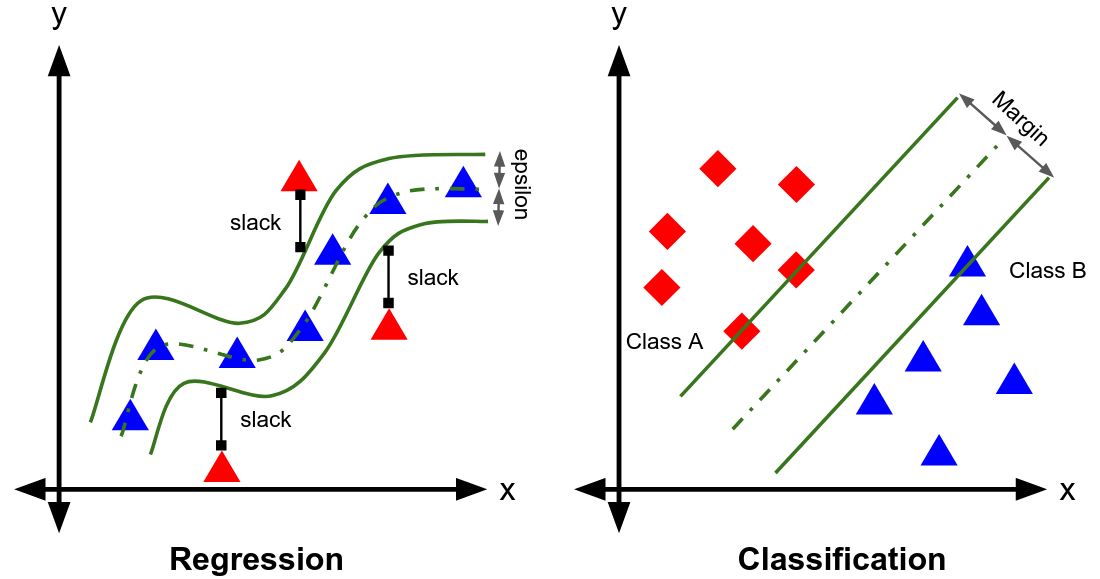
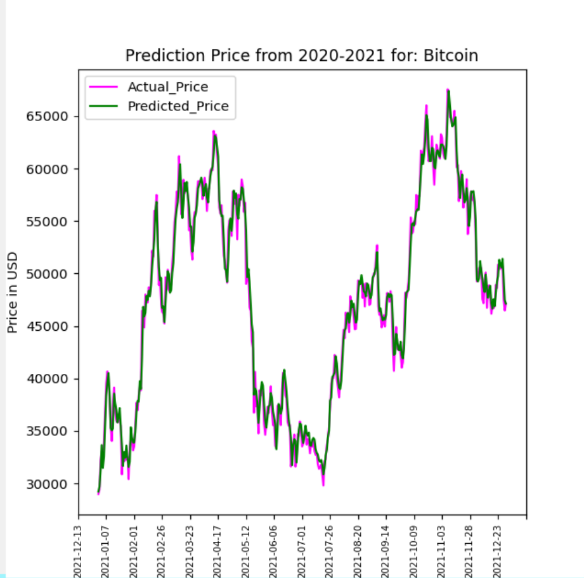
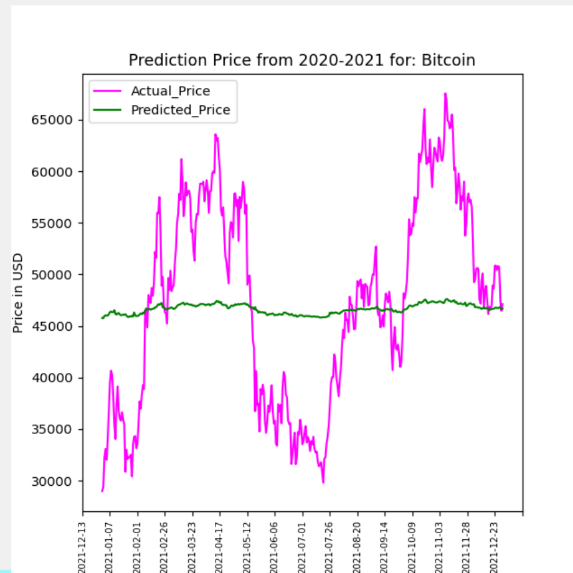


Fig 15-regression and classificationfrom[20]

In my project, the dataset consists of 5 features: 'price\_open', 'price\_high', 'price\_low', 'volume\_traded', 'trades\_count' representing this on a graph becomes very difficult. With support vector machines, a kernel trick can be used to transform the data into a higher dimension making the support vector machine perform better and allows us to map the data as linearly separable. In this project I have used the gaussian kernel (also known as the RBF kernel) to transform my feature set into a higher dimension.

When creating an SVM machine there are a variety of parameters that can be set. C is known as the box constraint, which is the slack variable that controls the decision boundary. A larger value of C will use a smaller hyperplane (or margin) which proves to be particularly good for the data that I am using. A larger value significantly outperforms the smaller value of C [fig 16].

Fig 16 – Left C of 1, Right C of 1e4



Epsilon (also known as gamma) signifies the width of the decision boundary, the larger the value of epsilon, the larger the decision boundary will be. A larger decision boundary will result in less support vectors being used by the SVM. However, this also causes a lower accuracy for the model, the smaller the value of Epsilon the better the model will perform, but it will also increase computation time. This can be seen in [figure 17], the lower value of epsilon massively outperforms the higher value of epsilon.

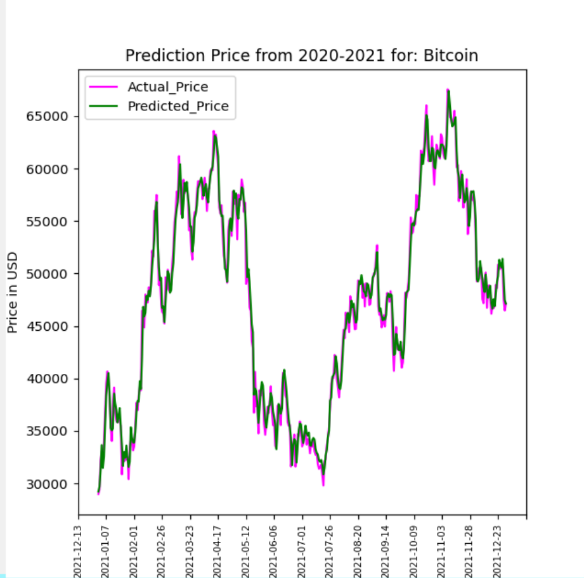
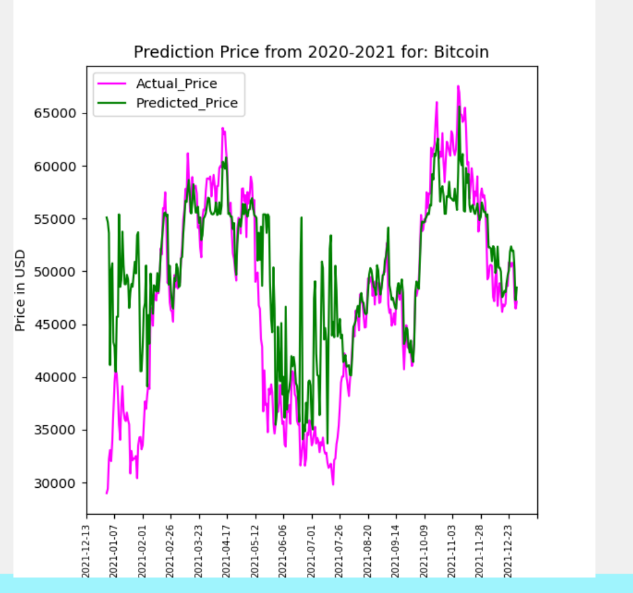


Fig 17 – Left Epsilon of 2, Right Epsilon of 0.0001

If I had more time I would have performed K-fold cross validation to tune the hyper parameters. Nested cross validation is needed to optimize the accuracy of the model. The outer loop of nested cross validation is used to estimate the generalization error (or overall accuracy of the model). The inner loop in cross validation is used in selecting and optimizing the hyper parameters. However, the default values used for my SVM returned an extremely high accuracy and therefore I didn’t feel the need to tune the hyperparameters.

### Artificial Neural Networks

Artificial Neural Networks (ANN) are a type of deep learning algorithm. The way they work is modelled after a human brain, with neurons sending signals to each other.

A neural network is composed of an input layer, some series of hidden layers and then an output layer [fig 18].

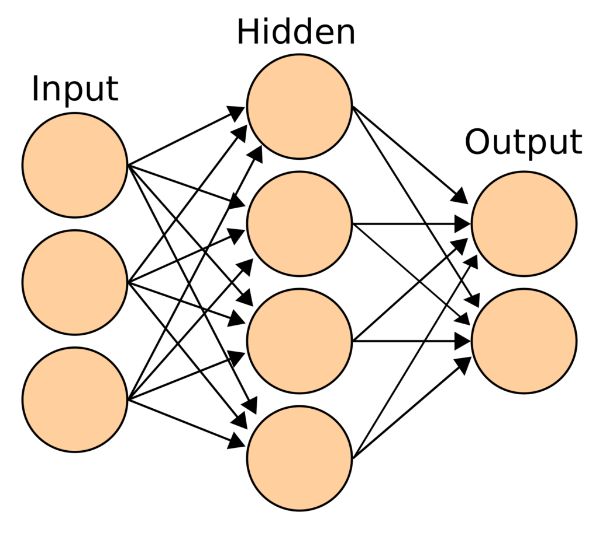


Fig 18 – representation of a neural network

This is a simple example of how a neural network can be modelled. The input layer is a 1:1 ratio with the number of features in the problem with an additional node acting as a bias. The input layer passes data to the hidden layer in a fully connected way. The data that is passed can be represented by an equation:

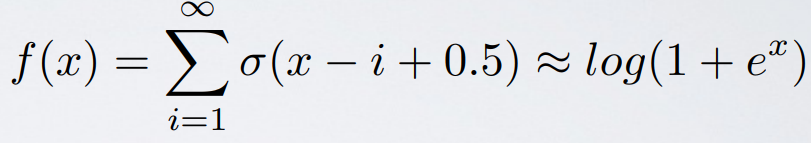
aj = Σ wji xi + wj0

i=1

D

Where aj is the is the activation at new node j. D is the number of nodes in the layer, ji is the path between node i and node j. W is the weight being passed and Wj0 is the bias weight added onto the activation.

The hidden layer performs a type of activation function (set in the code) that either links to another hidden layer or links to the output nodes. In my program I have chosen to use a rectified linear unit (RELU) activation which uses the equation:



This equation is fairly new and solves the vanishing gradient problem. This is when multiple layers are added, and the gradient becomes so small that it is very hard to train the model and get an accurate prediction.

The output node is just a single node for my regression problem that in my case will be an estimate of a coin price on a given date time, this is also referred to as Ŷ.

The goal of a neural network is to minimise the cost function (also known as loss), the network goes through using backpropagation or feedforward techniques to adjust the weights and biases of the network’s layers to reach a point of convergence where the cost is at its lowest point.

Text

Description automatically generatedAn optimiser is used to adjust the weights, some are more aggressive than others with weight adjustment. Being too aggressive can lead to overshooting the local minimum, but not adjusting the weights enough can lead to a very long process of getting to the local minimum.

Fig 19 – my program’s neural network

When creating my program I tested a lot of different options regarding optimisers, loss functions and layer sizes. As I was limited for time, I was not able to thoroughly test the options available but [Fig 19] shows how I have modelled my neural network.

When doing my research, I was able to find that the LSTM (long short-term memory) was the most effective layer to add. One of many sources stated [21] “Long Short-Term Memory Network or LSTM, is a variation of a recurrent neural network (RNN) that is quite effective in predicting the long sequences of data like sentences and stock prices over a period of time.”

The dense layer acts as the output layer, so my neural network only consists of one set of 128 nodes as a hidden layer. I found that this gave the best performance, because when connecting multiple layers in succession I found that the accuracy of the model decreased, and training time doubled [Fig 20].

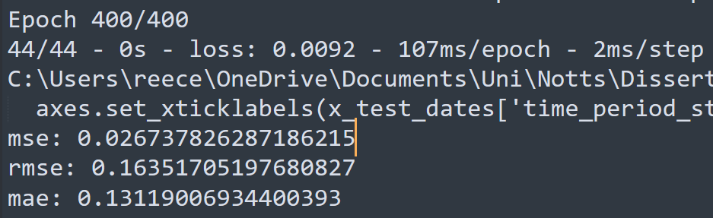
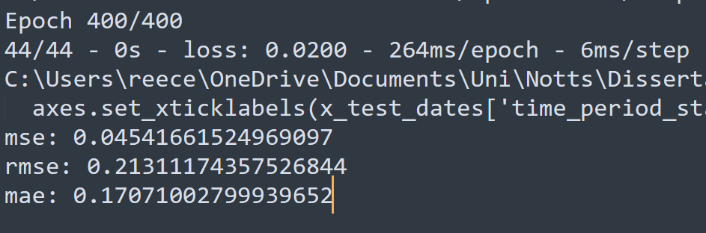


Fig 20 – left multiple hidden layers, right one hidden layer

The loss chosen was mean squared error, as this gave the best result across all the loss functions when optimising. When choosing an optimiser, the Adamax was chosen, this is a similar optimisation function to the Adam optimiser however, the Adamax optimiser is more robust to noise and as crypto price data is quite noisy Adamax is the better choice and yields better results.

## Mathematical Strategies

In this section I will be discussing the mathematical techniques implemented in my program. This section will describe how they work in general and then the application of the strategy within my program.

### Moving Average

The moving average strategy is a popular mathematical technique used amongst traders. There are many different types of moving averages that are calculated in different ways. [Fig 21] displays some of the different types of moving average techniques used by traders.

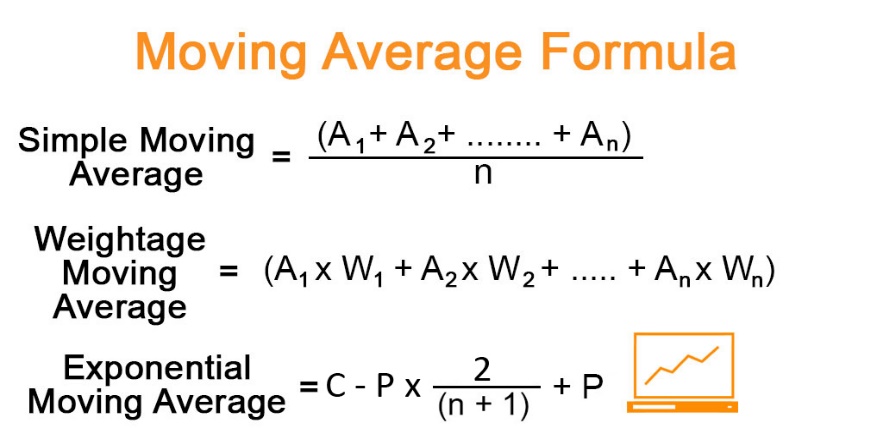


Fig 21 – Formula for calculating moving average from [22]

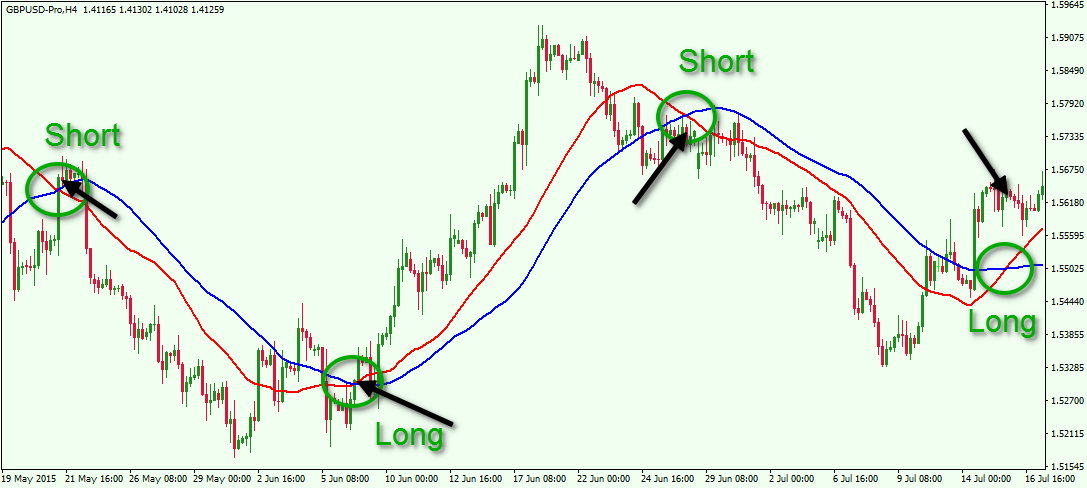


Fig 22 – Moving average on a graph [23]

The way this looks on a graph is shown in [Fig 22]. The blue line represents the longer moving average (50-day period) and the red line represents the shorter moving average (30-day period). Also represented on the graph is when to short (sell) and long (buy) the asset.

The trading technique works by using the crossover points of the moving average lines. When the shorter moving average line (red) becomes greater than the longer moving average line then this will signal a buy command and when the shorter moving average line is below the longer moving average then this will trigger a sell command.

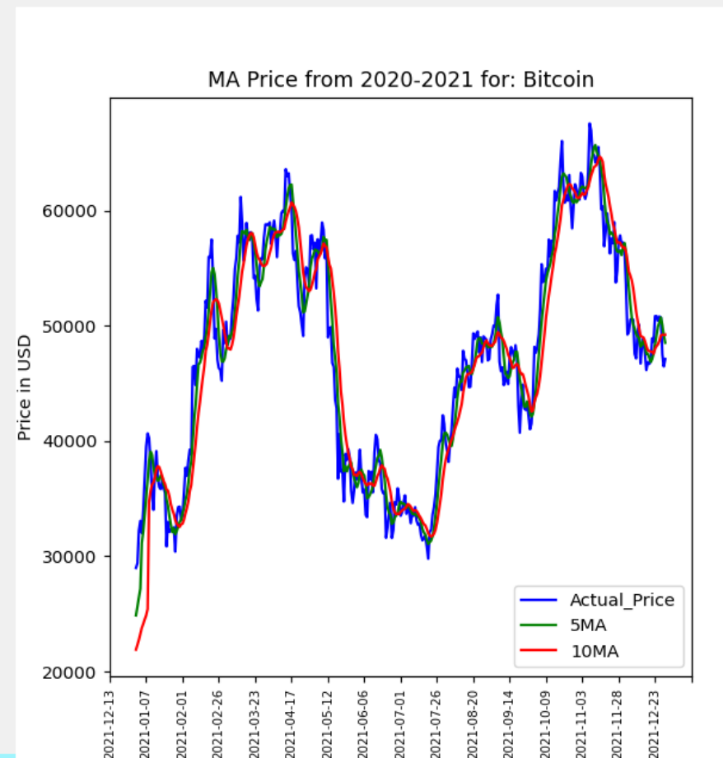
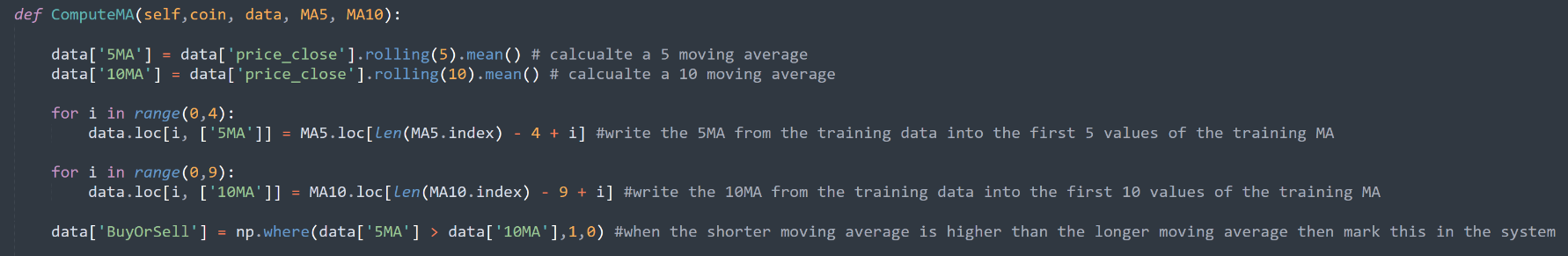


Fig 23 – Moving average from my program – bitcoin 2020-2021

[Fig 23] shows how the moving average works within my program. I have chosen a 5-day moving average as the shorter period and a 10-day moving average as the longer period I have chosen these periods to make my program more reactive when it comes to live runs. This also works well when the market crashes (please see 4.2.2).

Fig 24 – Moving average code



To Compute the moving average the rolling mean function is used from the panda’s library which will go through the dataset and add up x number of points and averages them. I then save this into a new column in the data frame [Fig 24]. However, as you need five data points to get the first moving average this would be NaN for the first four points, to fix this error I compute the moving average on the training data and then append the last 4 datapoints to the first 4 data points in the testing data. This is then also done for the 10-day moving average just with nine points.

Finally, I compute whether there should be a buy or sell signal (using the technique discussed above). If the 5 moving average is higher than the 10 moving average the row gets a 1 added to it to indicate a buy signal. Otherwise, a zero is written which indicates a sell signal.

For Live data, I have tested a 5-point moving average and a 10-point moving average. Also, a 25-point moving average and a 45-point moving average. The code for this can be found in the appendix [Appendix Fig 6]. I have chosen to do a point average instead of a daily average because otherwise the daily average would only be updating twice for the whole run which is not a good indication of reactionary decisions.

### Relative Strength Index

The relative strength index is another common mathematical indicator used by traders. This works by taking a relative strength (RS) which is calculated by the assets average gain divided by average losses. This relative strength is then standardized from a value between 1 and 100 resulting in the relative strength index. The formula for this can be seen [Fig 25].

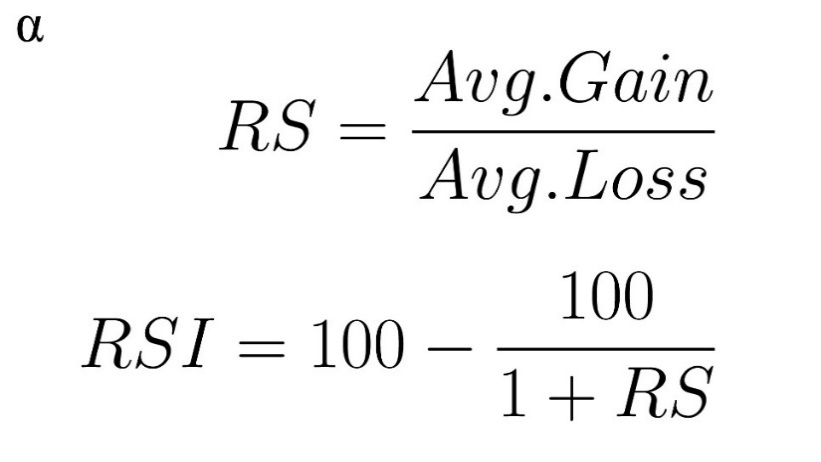


Fig 25 – Relative Strength Index Formula

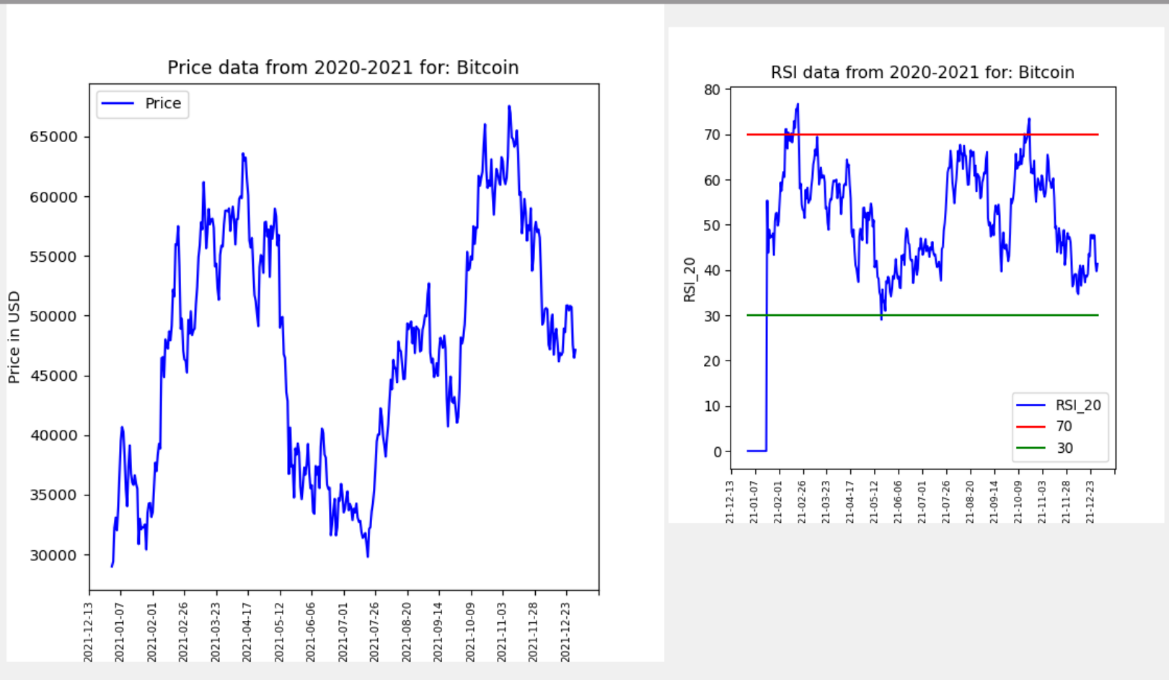


Fig 26 – Relative Strength Index from my program Bitcoin 2020- 2021

[Fig 26] illustrates how the relative strength index is displayed in my program. The left-hand side is the normal price chart of the asset (in this case bitcoin), and the right-hand graph is the relative strength index. As you can see, the peaks show a correlation to the tip of Bitcoin’s price.

This trading technique works by using the buy and sell lines (green and red respectively). If the RSI falls above the red line then the trading algorithm will sell the asset. As you can see from the price graph this will be the perfect time to sell as this would be just before the price starts to plummet. If the price falls below the green line, the algorithm will buy the asset. This is a great time to buy as the price is normally at its lowest. The problem with this technique is that the buy and sell lines are set by a human. This technique is normally a very safe technique meaning that profit will not be as high as other techniques, but this will also minimise losses.

Fig 27 – Relative Strength Index code for Live price trading



[Fig 27] Shows how the relative strength index is calculated within the live functionality of my program. The program checks if there are enough RSI values to calculate a new RSI from the live data is has already collected. Otherwise, the program will take close prices from the end of the training data to help compute a new RSI value for the new data point. Once the new RSI is calculated, the program will then determine if this lies inside the boundaries or if a buy or sell signal needs to be sent to the trading algorithm.

For an explanation as to how the RSI trading algorithm works see the commented code located in the appendix - [Appendix Fig 7].

# Results and Discussion



## Machine Learning Results

In this section I will be looking at the results obtained by the machine learning strategies and performing an analysis on their performance across all the coins. Each section represents the year that they were tested in, starting first with the support vector machine. All techniques started with $500 of currency that they used to trade for a period of time.

### SVM 2018-2021

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Error Values** | Bitcoin | Ethereum | Litecoin | LoopRing |
| Mean Squared Error | 0.009 | **0.006** | 0.017 | 0.016 |
| Root Mean Squared Error | 0.093 | **0.080** | 0.130 | 0.125 |
| Mean Absolute Error | 0.073 | 0.067 | 0.089 | **0.062** |
| Confidence | 0.991 | **0.994** | 0.983 | 0.984 |

Figure 28 – SVM 2018 - 2021 Error Value table

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Profits** | Bitcoin | Ethereum | Litecoin | LoopRing | Total |
| Bot Profit Made | $46004.43 | **$273674.47** | $156550.39 | $208302.06 | $684531.35 |
| Bot Percentage Profit | 9200.89% | **54734.89%** | 31310.08% | 41660.41% | 136906.27% |
| Hold profit | $312.73 | $2014.69 | $94.57 | **$5378.90** | $7800.89 |
| Hold percentage profit | 62.55% | 402.94% | 18.91%  Figure 29 – SVM 2018 - 2021 Profits table | **1075.78%** | 1560.18% |

The support vector machine performed extremely well; the tables above show the results from running the 2018-2021 SVM machine on the four cryptocurrency assets [Fig 28 – Fig 29]. The first table compares the error values from the predictions and actual price of the coins. The SVM machine performed extremely well across the board.

The built-in confidence function from sklearn gives a score of how well it thinks the trained SVM machine would do based on new data. So, the confidence is above 98% for all coins which is very accurate. As you can see from the first table, the best performing asset was Ethereum with the best values for 3/4 of the metrics.

This correlates to the second table where Ethereum obtained the highest amount of profit, turning $500 into $274,174.47 which is outstanding.

The hold profit is used as a comparison, this works by buying $500 worth of asset at the start of the trading period and selling it at the end of the period. As 2020-2021 was a good year for crypto all of these are profitable but none of them come even close to the profits made by the SVM bot.

### SVM 2021-2022

Figure 30 – Bitcoin price 2021-2022

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Error Values** | Bitcoin | Ethereum | Litecoin | LoopRing |
| Mean Squared Error | **0.002** | 0.004 | 0.003 | 0.005 |
| Root Mean Squared Error | **0.048** | 0.059 | 0.050 | 0.073 |
| Mean Absolute Error | **0.034** | 0.047 | 0.036 | 0.059 |
| Confidence | **0.998** | 0.996 | 0.997 | 0.995 |

Figure 31 – SVM 2021 - 2022 Error Value table

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Profits** | Bitcoin | Ethereum | Litecoin | LoopRing | Total |
| Bot Profit Made | $4555.51 | $11704.51 | $15462.09 | **$37195.20** | $68917.31 |
| Bot Percentage Profit | 911.1% | 2340.9% | 3092.42% | **7439.04%** | 13783.46% |
| Hold profit | $ -290.63 | **$ -258.69** | $ -284.93 | $ -379.85 | $-1214.10 |
| Hold percentage profit | -58.13% | **-51.74%** | -56.99% | -75.97% | -242.83% |

Figure 32 – SVM 2021 - 2022 Profits table

During this testing period, the prices of all coins dropped dramatically. This is a good test for all techniques to see if they could still make profit in a bear market. The SVM bot performed very well again, despite massive crashes on all coins. Bitcoin lost over half of its value within this period [Fig 30] so I did not expect the SVM bot to make much profit at all.

The tables above [Fig 31 – Fig 32] show that the bot managed to make $68,917.31 profit in a bear market which means that it performed far better than just holding the assets.

Despite getting the lowest error values and highest confidence bitcoin was the worst performing asset I think this is due to the way bitcoin’s price fell. Bitcoin’s price kept on fluctuating before eventually plummeting. The way the bot trades means that fluctuation in price highly constrains the performance. This can be seen with how the bot trades in a live scenario (4.1.3).

### SVM Live

Sensitive to rapid changes in price

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Profits** | Bitcoin | Ethereum | Litecoin | LoopRing | Total |
| Bot Profit Made | $41.82 | $20.94 |  |  |  |
| Bot Percentage Profit | 8.36% | 4.19% |  |  |  |
| Hold profit | $13.71 | $22.45 |  |  |  |
| Hold percentage profit | 2.74% | 4.49% |  |  |  |

Chart, line chart

Description automatically generated

### ANN 2018-2021

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Error Values** | Bitcoin | Ethereum | Litecoin | LoopRing |
| Mean Squared Error | **0.032** | 0.044 | 0.045 | 0.056 |
| Root Mean Squared Error | **0.178** | 0.210 | 0.213 | 0.237 |
| Mean Absolute Error | 0.145 | 0.168 | 0.134 | **0.089** |

Figure 36 – ANN 2018 - 2021 Error Value table

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Profits** | Bitcoin | Ethereum | Litecoin | LoopRing | Total |
| Bot Profit Made | $20088.12 | $25558.53 | $37422.62 | **$391359.36** | $474428.63 |
| Bot Percentage Profit | 4017.62% | 5111.71% | 7484.52% | **78271.87%** | 94885.72% |
| Hold profit | $312.73 | $2014.69 | $94.57 | **$5378.90** | $7800.89 |
| Hold percentage profit | 62.55% | 402.94% | 18.91% | **1075.78%** | 1560.18% |

Figure 37 – ANN 2018 – 2021 Profits table

The Artificial Neural Network also performed well. As you can see from the tables above [Fig 36– Fig 37] the ANN bot managed to total 94,885.72% which is $474,428.63 profit from $2,000 to begin with. This is an extraordinary amount of profit and from this we can see that this method is very profitable for this time period. LoopRing performed the best achieving 82.49% of the total profits made by the ANN bot.

Looking at the first table, you can see that Bitcoin performed better than all the other coins on two thirds of the evaluation metrics. However, this did not correlate to profits, in fact, Bitcoin had the lowest profits from all the assets in the ANN bot’s trading history. This is because of the general price trend of bitcoin during this time period. LoopRing’s price went from below $0.5 at its lowest and $3.5 at its peak. Whilst Bitcoin’s price prediction was more accurate, its price increase was not as high, and it also fell a lot quicker than LoopRing’s price [Fig 38].

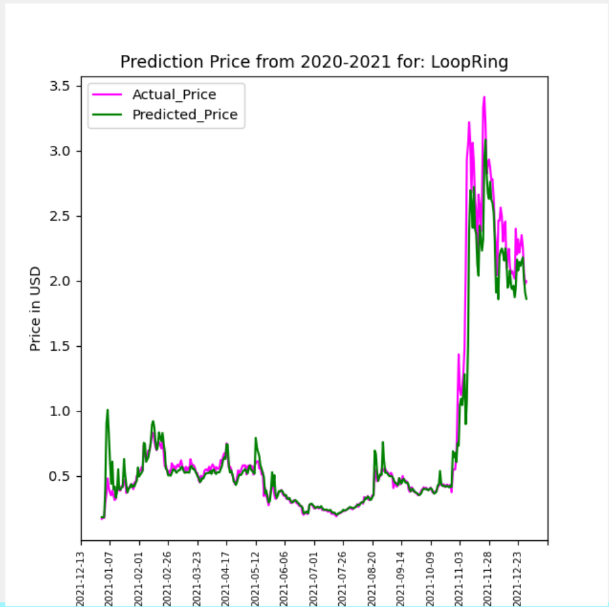
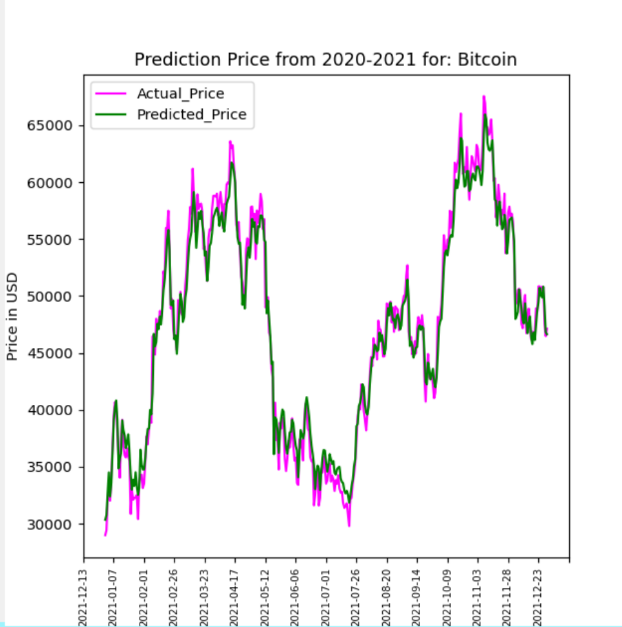


Figure 38 – Bitcoin price vs LoopRing

### ANN 2021-2022

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Error Values** | Bitcoin | Ethereum | Litecoin | LoopRing |
| Mean Squared Error | **0.013** | 0.018 | 0.014 | 0.040 |
| Root Mean Squared Error | **0.115** | 0.134 | 0.119 | 0.200 |
| Mean Absolute Error | **0.089** | 0.113 | 0.094 | 0.153 |

Figure 39– ANN 2021- 2022 Error Value table

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Profits** | Bitcoin | Ethereum | Litecoin | LoopRing | Total |
| Bot Profit Made | $4585.61 | **$19125.47** | $13747.26 | $1830.19 | $39287.53 |
| Bot Percentage Profit | 917.12% | **3825.09%** | 2749.45% | 366.04% | 7857.7% |
| Hold profit | $ -290.63 | **$ -258.69** | $ -284.93 | $ -379.85 | $-1214.10 |
| Hold percentage profit | -58.13% | **-51.74%** | -56.99% | -75.97% | -242.83% |

Figure 40 – ANN 2021- 2022 Profits table

Looking at the year 2021-2022, the ANN bot was still very profitable even during the downtrend of all assets. This makes the strategy a viable option when trading in a bear market. Yet again, Bitcoin’s price prediction was most accurate but was second least profitable [Fig 39 – Fig 40]. As mentioned in the previous section (4.1.2) this is due to the fluctuations in Bitcoin’s price. As seen from the live run, when fluctuations in price occur the algorithm does not perform very effectively.

However, despite this, the bot still managed to total $39,287.53 which is a $40,501.63 difference if you were just to hold $500 of each asset from the start of the trading period. Ethereum outperformed all other coins in this scenario, this is because the downtrend for Ethereum was a bit smoother and it also recovered a lot more towards the end of the time frame [Fig 41].

Chart, histogram

Description automatically generated

Figure 41 – Ethereum Price Trend

### ANN Live

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Profits** | Bitcoin | Ethereum | Litecoin | LoopRing | Total |
| Bot Profit Made | $39.69 | $20.67 |  |  |  |
| Bot Percentage Profit | 7.94% | 4.13% |  |  |  |
| Hold profit | $13.71 | $22.10 |  |  |  |
| Hold percentage profit | 2.74% | 4.42% |  |  |  |

## Mathematical Results

In this section I will be looking at the results obtained by the machine learning strategies and performing an analysis on their performance across all the coins. Each section represents the year that they were tested in, starting first with the Moving average technique. All methods started with $500 of currency that they used to trade for a period of time. There are not any performance metrics for the mathematical techniques due to them not making predictions. So, the tables are just profit values.

### Moving Average 2018-2021

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Profits** | Bitcoin | Ethereum | Litecoin | LoopRing | Total |
| Bot Profit Made | $1693.57 | $4276.69 | $3762.84 | **$56240.21** | $65973.31 |
| Bot Percentage Profit | 338.72% | 855.34% | 752.57% | **11248.04%** | 13194.67% |
| Hold profit | $312.73 | $2014.69 | $94.57 | **$5378.90** | $7800.89 |
| Hold percentage profit | 62.55% | 402.94% | 18.91% | **1075.78%**  Figure 44 - Moving Average 2018 - 2021 Profits table | 1560.18% |

The Moving average bot performed far better than I anticipated. As the prices were going up in this duration I expected it to make around the same amount as the holding profit however, this bot managed to achieve 8.5 times more profit than just holding the assets[Fig 44]. This is because this strategy can also be smart around price decreases due to it selling when the 5-point average line falls below the 10-point average line. This means that this strategy will find an optimal point to enter the market during a downtrend.

### Moving Average 2021-2022

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Profits** | Bitcoin | Ethereum | Litecoin | LoopRing | Total |
| Bot Profit Made | $486.35 | $1287.79 | $1228.65 | **$2946.58** | $5949.37 |
| Bot Percentage Profit | 97.27% | 257.56% | 245.73% | **589.32%** | 1189.88% |
| Hold profit | $ -290.63 | **$ -258.69** | $ -284.93 | $ -379.85 | $-1214.10 |
| Hold percentage profit | -58.13% | **-51.74%** | -56.99% | -75.97% | -242.83% |

Figure 45 – Moving Average 2021- 2022 Profits table

During 2021-2022 the moving average bot performed well [Fig 45]. Due to its smart selling technique this strategy was again able to make profit but this time in a difficult market scenario where prices fell by over 50% for all assets. Whilst holding the assets lost $1,214.10, the moving average strategy was able to profit $5,949.37 which is particularly good in this market.

### Moving Average Live

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Profits** | Bitcoin | Ethereum | Litecoin | LoopRing | Total |
| Bot Profit Made | $24.28 | $34.49 |  |  |  |
| Bot Percentage Profit | 4.86% | 6.9% |  |  |  |
| Hold profit | $13.89 | 22.45 |  |  |  |
| Hold percentage profit | 2.78% | 4.49% |  |  |  |

### Relative Strength Index 2018-2021

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Profits** | Bitcoin | Ethereum | Litecoin | LoopRing | Total |
| Bot Profit Made | $845.55 | $433.26 | $410.78 | **$1665** | $3354.59 |
| Bot Percentage Profit | 169.11% | 86.65% | 82.16% | **333%** | 670.92% |
| Hold profit | $312.73 | $2014.69 | $94.57 | **$5378.90** | $7800.89 |
| Hold percentage profit | 62.55% | 402.94% | 18.91% | **1075.78%** | 1560.18% |

Figure 49 – RSI 2018-2021 Profits table

The Relative Strength Index technique is a safer technique that attempts to minimise loss instead of making huge amounts of profit. The technique managed to make profit. However, it made less than half of what the hold profit made which I was not expecting. This means that during a bull run where prices are constantly increasing this technique is not optimal. This is because the RSI will be above 70 and therefore, the technique will think that the asset is overbought and will sell it. This causes less profit to be made when prices a continuously rising.

LoopRing still managed to make most of the profits with an impressive 333% return on investment. But this was still way below the hold profit for this asset. LoopRing’s price increased dramatically around the 3rd of November which is when the RSI rose to well above 80 so the technique would have missed out on most of the gains [Fig 50].

Graphical user interface, chart, application, histogram

Description automatically generated

Figure 50 – LoopRing price 2020- 2021

### Relative Strength Index 2021 - 2022

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Profits** | Bitcoin | Ethereum | Litecoin | LoopRing | Total |
| Bot Profit Made | $9.12 | $-48.72 | $-36.21 | **$12.10** | $-63.71 |
| Bot Percentage Profit | 1.82% | -9.74% | -7.24% | **2.42%** | -12.74% |
| Hold profit | $ -290.63 | **$ -258.69** | $ -284.93 | $ -379.85 | $-1214.10 |
| Hold percentage profit | -58.13% | **-51.74%** | -56.99% | -75.97% | -242.83% |

Figure 51 – RSI 2021-2022 Profits table

During 2021 – 2022 the Relative Strength Index method did not perform well. However, it did minimise loss during a huge crash in the market which is exactly what the method is supposed to do. In comparison with the holding profit, it managed to reduce the risk from -242.83% to only -12% [Fig 51]. It also managed to make profit on 2 assets during this crash which is surprising.

### Relative Strength Index Live

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Profits** | Bitcoin | Ethereum | Litecoin | LoopRing | Total |
| Bot Profit Made | $1.45 | $0.34 |  |  |  |
| Bot Percentage Profit | 0.29% | 0.007% |  |  |  |
| Hold profit | $13.89 | $22.45 |  |  |  |
| Hold percentage profit | 2.78% | 4.49% |  |  |  |

## Combined Evaluation

### 2018 – 2021

From section 4.2 the tables shown illustrate how well each technique did in the year 2018-2021. As expected, the machine learning strategies outperformed the mathematical ones. The bar chart below shows how each of these strategies compared in terms on total profit [Fig 54]. As shown by the graph, RSI made a miniscule amount of profit in comparison to SVM. Where the SVM bot made 204 times more profit than RSI.

Comparing the machine learning techniques, both SVM and ANN made significant profit. However, SVM still performed better. This is due to the SVM bot being more accurate in its predictions. [Fig 55] Shows the comparison in error metrics between the two methods. This is calculated on average between all coin assets.

Looking at the mathematical strategies the MA bot outperformed the RSI bot by over 19-fold. This is because the RSI method is a safer technique that’s used to make small margins of profit and reduce the risk of losing in a market crash. The moving average technique performs well in both scenarios however, it may not be as responsive as the RSI method due to the average only updating daily.

Looking at the crypto coins, it was interesting to see that LoopRing was the most profitable asset for 3/4 of the trading techniques. The hold profit suggests that this asset was the most volatile as it made over double of any other asset available. This coin has a far smaller market capacity compared to bitcoin and so we can see that these trading technologies will benefit from a faster growing coin.

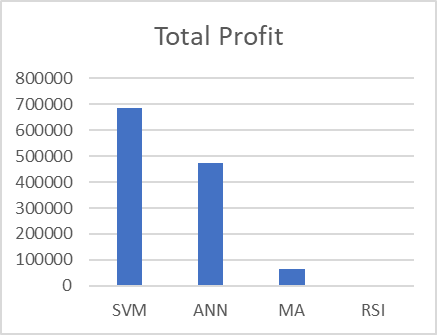


Figure 54 – Total Profit 2018- 2021

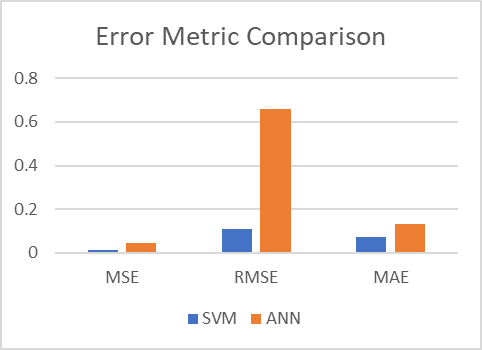


Figure 55 – Average error comparison 2018- 2021

### 2021- 2022

In this era of cryptocurrency, the market crashed dramatically with over 50% losses on all assets tested. I thought this would be a good test to see how each technique faired with a market crash. Looking at [Fig 56] we can see that 3 of the techniques managed to make a substantial amount of profit considering the price crash for all assets.

SVM again coming out on top with $29,629.78 more profit than ANN in second. Again, looking at the error metrics [Fig 57] we can see that this time all error values were far lower than 2018-2021 so the trading bots were more accurate with their predictions. However, as the market was crashing they were unable to make profit, instead they were able to avoid losing money in the downfall whilst knowing the optimal entry point and best point to sell again before another crash.

MA managed to outperform RSI again. This is because the moving average will buy once the price starts to recover (5-point line goes above the 10-point line) and sell when the reverse occurs. Meaning that it is able to avoid the big crashes, but it is not quite as reactive as the machine learning techniques. RSI on the other hand is a 20-point average so will take a long time to adjust to knowing that the asset is overbought meaning that it will keep the asset for a while whilst it is crashing.

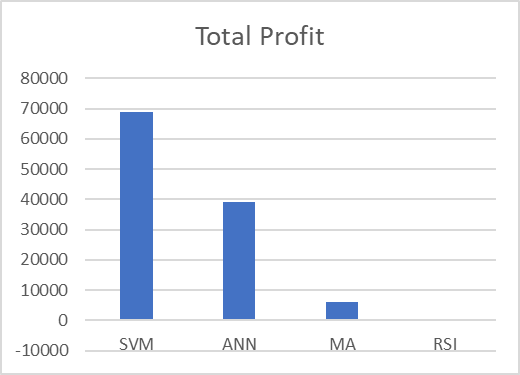


Figure 56 – Total Profit 2021- 2022

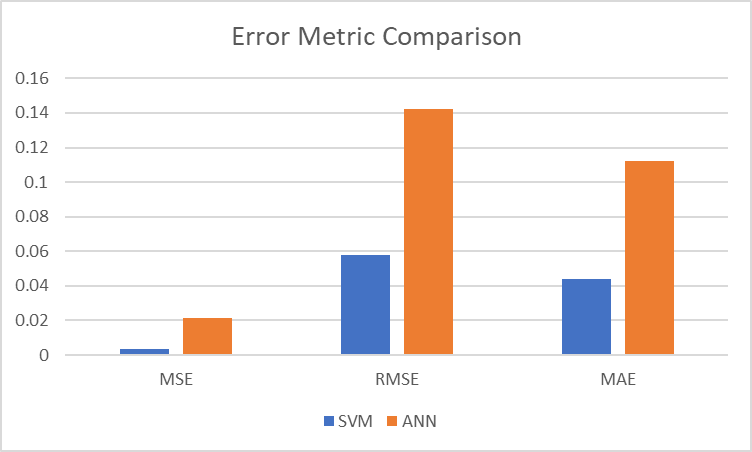


Figure 57 – Average error comparison 2020- 2022

### Live

Bitcoin – 09/08 – 12/08

Ethereum – 11/08 - 14/08

Litecoin – 22/08 – 25/08

## Did I meet my Aims/Objectives?

In this section I will talk about the objectives I set out in section 1.2 and whether I met these objectives. For the most part all the objectives were met, but for a more in-depth analysis of what did not go so well, see section 5.2.

### Simple and effective UI (Fully met)

This objective was to make the user interface easy for non-computer literate people to understand and use. The UI that I made is simple yet effective, [Fig 12 - 13] users are easily able to switch between the 3 trading scenarios by clicking on the drop-down menu and changing the scenario.

I also believe that the trading bot screen is well laid out enabling users to see all the important details without information overload. Furthermore, the graphs provide an insight into how the bots are trading in real time so users can see how each trading method works.

### Machine Learning Strategies (Fully Met)

I have fully met this objective because both strategies work with all trading scenarios. The support vector machine trading bot is the most effective trading method and bot machine learning strategies are very accurate for past data.

The artificial neural network bot is still effective, but it takes a long time to train and then yields slightly worse results (Section 4.1). However, both methods return a lot more than I was expecting in regard to the past data. Both methods performed adequately with the live results it would be interesting to see how they perform when run for longer.

### Mathematical Trading techniques (Fully Met)

The mathematical trading techniques I set out to implement in section 1.2 have been integrated into the program. In terms of performance, they did not perform as well as the machine learning techniques, but they served as a good comparison. However, these mathematical techniques are a lot easier to compute and take less computational power, which is something to take into consideration.

### Multiple Trading Scenarios (Fully Met)

I initially wanted to implement 3 different scenarios to test the strategies and their performance in these given situations. The first scenario (2018-2021) was to test how well they performed in a bull run (Where the market is constantly on the rise) to see how much profit each strategy to make. The second scenario (2021-2022) was during a bear market (where prices fall) to see if each technique could minimise the loss and potentially make profit. Finally, the live demo was to see how each technique would react to an unseen problem. This objective was fully met, as seen from section 4.1-4.3 where the results are split into these 3 scenarios.

### Variety Of Crypto Coins (Fully Met)

During this research I wanted to test a wide range of crypto assets to see how each strategy would perform in relation to the volatility of the cryptocurrency. This was interesting to see and discover how the different coins performed. As a general rule, the smaller coins will always be more volatile meaning that more profit is to be made during a market rise but also more money can be lost when the market is falling.

### Data Analysis (Partially Met)

Whilst I performed some data analysis in section 4.1-4.3, I believe that this objective was not fully met. I collected a lot of data and exported this to an excel file, but I could have done more analysis on the data that I collected. For example, it would have been useful to see how the different trading bots reacted to the live data, such as, the number of sell and buy orders that the bot executed and comparing these with each of the bots to see when a decision was made.

Furthermore, I believe that the comparison between different assets would have been beneficial instead of a table format, a graph displaying each individual asset for each method would have made analysing this data easier. This would have enabled me to see how the volatility of each asset worked with each given method. For instance, one method might have been more efficient when trading a more volatile asset, whilst another would have been better trading a slow-moving asset.

# Critical Appraisal

In this section I will talk about how I think my project has gone including what went well, badly, and what I would have done in hindsight knowing what I know now.



## What went well

During the project I had weekly meetings with my supervisor and in each meeting I showed significant progress towards my end goal with clear objectives set for the following week.

When coding the project, I made sure that the code followed a readable format ensuring that the code was commented thoroughly with both method level comments explaining what a function did and line comments to explain more complicated lines of code.

For the predictions, the machine learning methods were far more accurate than I anticipated. Even for live runs the accuracy follows the same general pattern as the real price. However, the past data is where they really excel, achieving 90%+ accuracy.

From the large amount of data I collected, I was able to see which method was the best. At the start I predicted that the ANN would be the most profitable trading bot. But I found that the SVM bot was far more profitable than all the other bots.

## What went badly

During my project I encountered some problems that were either difficult to solve or I was unable to solve them. The first problem I encountered was for the RSI bot where I wanted to show two graphs on the same graph with the RSI being integrated within the price graph. However, I was unable to solve this problem and ended up putting two graphs side-by-side so the user could still see both graphs.

Secondly, when running the live data, I was limited to 100 API calls per day. This meant that initially when I tried to do 15-minute calls I would run out of requests before the day was over making the program crash. To fix this I had to get additional API keys and also change to 30-minute calls instead.

Furthermore, I was unable to run all the trading bots and coins at once when running the live data. This was due to the lack of API calls. So, I ran all the bots on one coin at a time as I thought this would be better to compare rather than comparing the results of the different coins.

Finally, the live data failed to write to the excel file after running for two days. This was because it was writing the data from the day before the live run and then due to xlsxwriter making you unable to edit excel files the live data wasn’t written to the file. To fix this I added all the data to an array at once and then wrote it all to the same file.

## What I would have done differently

There are a few things that I would have done differently if I were to start the project again knowing what I know now. Firstly, I would have tested different types of moving averages as there are exponential averages and not just sum averages. Also, I would have experimented with different N (number of days) in the average formula.

Additionally, for the RSI bot I would have tested multiple different buy and sell lines using an algorithm for the past data. I would run the bot multiple times with buy and sell lines ranging from 20 and 70 then collect the most profitable values instead of just setting them at 25 and 70.

Moreover, I would have tried a variety of different trading algorithms instead of just buying and selling the whole amount. For example, with the machine learning algorithms I would introduce a prediction accuracy into the trading bot so if the bot was more accurate for a given day then the buy or sell would be weighted higher than another day that it is not as accurate.

Leading on from this, I would also take into consideration the trading fees that are incurred when selling an asset. Most marketplaces charge a fee every time an asset is bought and sold. These are referred to as taker and maker fees. The average marketplace takes around 0.4% of the buy trade and 0.6% of the sell trade. Implementing this into the trading algorithm would mean that the bots would have to be more conserved when it comes to buying and selling assets to reduce the total fees that it would have to pay.

## Future Development

There are a variety of different recommendations I would like other researchers to consider when undertaking this type of project. For the future, I suggest that a variety of different techniques are combined with each other to attempt to make a more profitable trading bot. For example, combining the moving average with the support vector machine could improve the results. Whereby, the machine learning bot contributes for a larger percentage of the decision, but this is then weighted with the moving average bot to help improve the SVM bot when prices are fluctuating.

In addition, I think that adding the ability to short assets would be beneficial to the trading bots. This is known as arbitrage trading where you can invest money into predicting if the price of an asset will rise (long) or fall (short) you would then make money if you shorted the asset and the price fell. In my project, if the bots thought that the price would fall then the just sell the asset. If arbitrage trading was introduced then the bots would be able to make money off of price falls instead of waiting for the price to recover.

Furthermore, there are many more other techniques that I was not able to experiment with due to time constraints. I would recommend other researchers to test other machine learning and mathematical techniques. For example, the regression tree and Bollinger Bands mathematical strategy. Referencing this paper would let other researchers be able to compare all the strategies to get the best insight as to which technique is the most effective.

Finally, I believe that testing more assets would be more beneficial for the research. I took assets within the top 10 because there are not a lot of assets with data that dates back to 2018. However, if other researchers were just looking at live or 2022 data then they would be able to test more volatile assets which would be interesting to see how the trading bots react in these scenarios.

# Conclusion

To conclude, my project was able to successfully create four different types of trading bots of two different sub methods (machine learning and mathematical). Almost all of my initial aims were met but there were still improvements that I could have made to fully meet all of my objectives. The main aim of my project was to be able to compare the 4 types of technical strategies in the cryptocurrency market. I was able to achieve this and found that the machine learning approaches were far superior to the mathematical ones in all three trading scenarios. All trading bots performed extremely well during the uptrend of 2018-2021 (as expected) but then the machine learning bots also performed far better than expected in the market crash of 2021-2022. In terms of the live trading, I wasn’t expecting a large amount of profit as this was only being run for a couple of days. However, to achieve higher than the holding profit on all coins was exceptional for the ANN and SVM trading bot. Also, the moving average achieving higher in most of the coins is also impressive.

I feel like this has advanced the technology in this area because…

Future developments…

# References

[1] Lamport, L. (1994). LATEX : A document preparation system (Second ed.). Addison-Wesley.

[2] PrimeXBT Editorial Team “Cryptocurrency Trading Strategies: Learn to Profit from Bitcoin, Ethereum, and More”. April 13, 2021. Last visited 09/06/2022. Found at: <https://primexbt.com/for-traders/cryptocurrency-trading-strategies/>

[3] Evan Jones – CryptoVantage “A Brief History of Cryptocurrency” June 17, 2022, Last Visited: 13/06/2022. Found at: <https://www.cryptovantage.com/guides/a-brief-history-of-cryptocurrency/>

[4] Scott Chipolina “Why do Altcoins Usually Follow Bitcoin?” Decrypt. Sep 2020. Last Visited 17/06/2022. Found at: <https://decrypt.co/42834/why-do-altcoins-usually-follow-bitcoin>

[6] CoinAPI.io “Market Data - Starter Guide” 2022. Last visited: 28/07/2022 Found at: https://docs.coinapi.io/?python#historical-data-get

[7] Reece Cripps “COMP4037 Research Methods Coursework 3 – Computer Science in Cryptocurrency” April 2022.

[8] Hitam, N.A., Ismail, A.R. and Saeed, F., 2019. “An optimized support vector machine (SVM) based on particle swarm optimization (PSO) for cryptocurrency forecasting.” Procedia Computer Science, 163, pp.427-433.

[9] Patel, M.M., Tanwar, S., Gupta, R. and Kumar, N., 2020. “A deep learning-based cryptocurrency price prediction scheme for financial institutions.” Journal of information security and applications, 55, p.102583.

[10] P. Jay, V. Kalariya, P. Parmar, S. Tanwar, N. Kumar and M. Alazab, "Stochastic Neural Networks for Cryptocurrency Price Prediction," in IEEE Access, vol. 8, pp. 82804-82818, 2020

[11] Borrageiro, G., Firoozye, N. and Barucca, P., 2022. “The Recurrent Reinforcement Learning Crypto Agent” (pp1- 19).

[12] Koker, T.E.; Koutmos, D. Cryptocurrency Trading Using Machine Learning. J. Risk Financial Manag. 2020, 13, 178.

[13] J. Sun, Y. Zhou and J. Lin, "Using machine learning for cryptocurrency trading," 2019 IEEE International Conference on Industrial Cyber Physical Systems (ICPS), 2019, pp. 647-652.

[14] Ellis, C.A. and Parbery, S.A., 2005. Is smarter better? A comparison of adaptive, and simple moving average trading strategies. Research in International Business and Finance, 19(3), pp.399-411.

[15] Akyildirim, E., Goncu, A. and Sensoy, A., 2021. Prediction of cryptocurrency returns using machine learning. Annals of Operations Research, 297(1), pp.3-36.

[16] Pionex trading bots Last visited: 01/07/2022 Found at: <https://www.pionex.com/en-US/>

[17] CoinRule trading bot Last visited: 01/07/2022 Found at: <https://coinrule.com/>

[18] TradeSanta trading bot Last visited: 02/07/2022 Found at: <https://tradesanta.com/en>

[19] cryptohopper bot Last visited: 02/07/2022 Found at: <https://www.cryptohopper.com/>

[20] Support Vector Regression – Beny Maulana (2019) - Last visited: 15/07/2022 Found at: <https://medium.com/it-paragon/support-vector-machine-regression-cf65348b6345>

[21] Keras LSTM Layer Explained for Beginners with Example – Palash Sharma – Feb 2021 – Last visited: 18/07/2022 - Found at: <https://machinelearningknowledge.ai/keras-lstm-layer-explained-for-beginners>

[22] Moving Average Formula– Madhuri Thakur - Educba - Last visited: 01/08/2022 Found at: <https://www.educba.com/moving-average-formula/>

[23] Anatomy of Popular Moving Averages in Forex - Last visited: 01/08/2022 Found at: <https://forextraininggroup.com/anatomy-of-popular-moving-averages-in-forex/>

# Appendix A: Supplementary Materials I

Appendix Figure 1 - Pionex Trading bot page

Chart

Description automatically generated

Figure 2 – CoinRule Available strategies

A picture containing chart

Description automatically generated

A screenshot of a computer

Description automatically generatedFigure 3 – TradeSanta Bot Home page

Figure 4 – CryptoHopper Bot home page

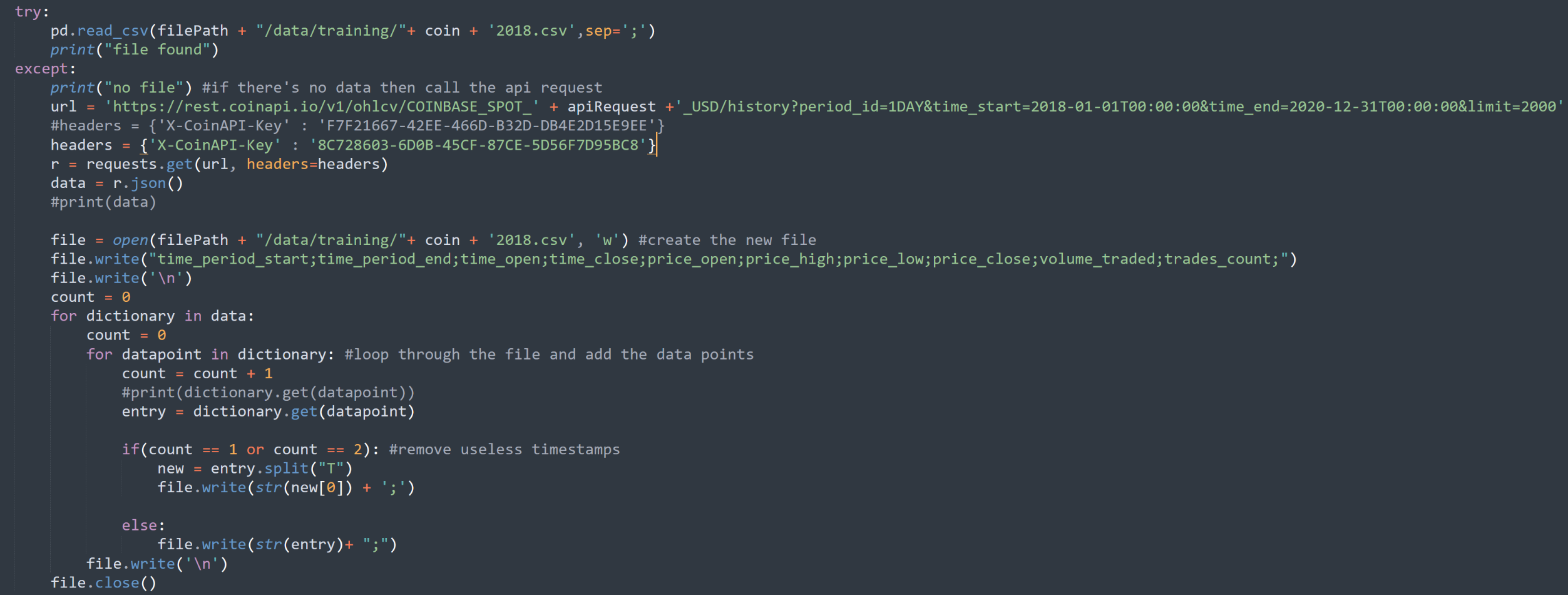
A screenshot of a computer

Description automatically generated

Figure 5 – 2018 data reading (Code)

See if a file already exists

create the API request with daily intervals



Write the column names to the file

For every row in the returned data from the API (the rows are dictionaries)

Loop through the dictionaries in each row

Split the time to just add the date

Add a semicolon to the end of the line

Text

Description automatically generated Figure 6 – Moving Average Live computation (Code)

Update the plot on the screen

If there’s enough points for 25 then use the close price from the live data already collected

If the live data has enough points in it then use the close price to calculate an average

If the live data has less than 45 points in get points from previous testing data

If the live data has less than 25 points in get points from previous testing data

Initialise the averages

Text

Description automatically generated

Figure 7 – RSI Live Trading Algorithm

When Buying coins calculate how much x amount of currency yields an amount of asset using the current price

When selling coins calculate how much x amount of asset would sell for using the current price

If there are no coins to sell then the algorithm is waiting for an optimal entry point into the market

Check if the RSI went above 70 and if so sell any coins if there are any.

Buy the coins and calculate the new profit

In the data frame column check if there’s a buy signal. If there is this means that the RSI went below 20.