The University of Hong Kong

Department of Computer Science

FITE4801 Final Year Project (2023-24)

FYP23003: Discovering Winning Strategies in Algorithmic Trading Through a Comprehensive Approach

Interim Report

Lee Sze Choi (3035779571)

Li Wang Hei (3035782279)

Tsang Yan Chak (3035780790)

Under the supervision of Prof S. M. Yiu

Date of Submission: 21 January 2024

Table of Contents

1. Introduction

1.1 Project Background

Algorithmic trading first came into existence in the 1970s when algorithms are used for trade execution. As the Internet develops in the 1990s, financial markets began to widely adopt the use of electronic communication networks (ECNs) for digital execution. In the past decades, the latency of trade executions through ECNs has dropped to hundreds of microseconds (Menkveld & Zoican, 2017; Cboe, 2018). With lower latency and higher data availability, algorithmic trading developed quickly. Since then, artificial intelligence and machine learning had gained massive attention, which promoted the development in the algorithmic trading industry.

With the growth of algorithmic trading, many research studies have been conducted. Nuti et al. (2011) identified 3 main types of algorithmic trading systems: execution trading, market-making, and proprietary trading systems. Nuti et al. defined that execution trading systems aim to minimize market impact and time costs in executing positions. The profits gained from these execution trading systems primarily originate from commissions and fees. They also defined that market-making systems provide market liquidity by quoting bid and ask prices simultaneously. Market-making systems gain profits based on spreads, which is the difference between the bid and ask prices. Proprietary trading systems apply various strategies, ranging from technical analysis to machine learning (Nuti et al., 2011) which gain profits from price movements in the underlying security. This research study has served as the foundation of classifications between various algo trading systems.

1.2 Motivation

In a typical algorithm, complex mathematical models are used to compute and generate trading signals. Despite the fact that profitable strategies were developed in the past, previously lucrative strategies had been exploited and returns have diminished given the industry’s fierce competition.

Research studies had been done on individual algorithms. Chu et al.'s (2020) research focused on high-frequency momentum trading that combines two exponential moving averages to indicate trading signals for cryptocurrencies. Lv et al. (2019) investigated machine learning and deep learning systems using 44 technical indicators as input. This motivated us to research and analyse known strategies, in the aim of combining their strengths and reducing their weaknesses. The project focuses on a wider range of trading strategies, as opposed to solely focusing on deep learning models which were investigated in previous final year projects. This project aims to create an outperforming strategy that can generate consistent returns by analysing and combining existing strategies.

Apart from creating a profitable strategy, this project also aims to bridge the research gap concerning the omission of transactional costs and the distinction between institutional investors and retail traders. It is found that there is a dearth of research studies that focus on algorithmic trading for retail investors and compare performance across various markets, while also considering transactional costs.

The existing studies on algorithmic trading tend to concentrate on two types of algorithmic trading systems: execution trading and market-making. The studies on execution trading primarily aim to reduce execution costs and risks (Feng et al., 2012; Gatheral & Schied, 2013; Hendershott & Riordan, 2012), which is provided as a service for institutional investors. The studies on market-making involve high-frequency trading using complex algorithms (Alexandru, 2016). As the majority of retail investors could not perform high-frequency trading and reduce execution costs, studies on both execution trading and market-making systems are not feasible for them.

There are fewer research studies focusing on proprietary trading systems. These studies also tend to cover only one specific strategy and focus on a single market, without taking transactional costs into consideration. For instance, Chu et al. (2020) conducted research on moving average strategy on cryptocurrencies; however, the study did not account for any transactional costs. Lv et al. (2019) explored various machine learning and deep learning models for predicting stock prices, while considering transactional costs; however, this study is limited only to the stock market.

The research gaps concerning the omission of transactional costs and the absence of back-testing in multiple markets are evident. Additionally, the majority of research studies concentrate on institutional investors. Therefore, there is a need for a comprehensive study on algorithmic trading strategies that specifically cater to retail investors, while taking transactional costs into consideration.

1.3 Project Objectives

This project has two main objectives. The first objective is to evaluate a wide range of trading strategies and arbitrage opportunities across the United States (U.S.) stock market and cryptocurrency market that are available for retail traders. The profitability of existing strategies and their ability to predict future prices will be examined. This project aims to provide a comprehensive overview of the return profile of existing strategies after taking transaction costs and brokerage fees into account.

The second objective is to create a profitable strategy that can outperform the market. Existing known strategies in the market will first be analyzed, followed by parameter optimization through back-testing. This project will also analyze and compare the efficiency of various strategies in both the U.S. stock market and the cryptocurrency market. After performing back-testing on both existing and new trading strategies, a final algorithmic trading system will be developed by combining the best-performing algorithms identified. Ultimately, the overarching objective is to produce excess returns for retail investors across different market conditions.

1.4 Project Deliverables

This project has two major deliverables. The first deliverable is a comprehensive research report evaluating the existing trading strategies. These results provide an overview of the most common algorithms used in trading and evaluates their profitability and consistency.

The second deliverable is the self-developed trading algorithms with forward testing results. It is anticipated that the project will develop 5 to 10 trading strategies. The logic of these strategies, along with the evaluation results, reports and codes, will be provided.

The algorithm developed by the project aims to generate stable and positive returns, and forward testing results will be used to assess the feasibility of the algorithm in a live trading environment.

1.5 Outline of the Report

The remainder of the report will cover the literature review *(Section 2)*, methodology and evaluation metrics *(Section 3)*, the trading algorithms with the corresponding back-testing results and evaluation *(Section 4)*, and the project schedule *(Section 5)*. This report ends with a conclusion *(Section 6)*.

2. Literature Review

Based on Nuti et al. (2011)'s classification on different types of algorithmic trading systems, this project has reviewed different researches on proprietary algorithmic trading systems.

Kilgallen (2012) has implemented a moving average strategy across the stock market, commodities market and currencies. The moving average is calculated by the average price of a certain security in the last n months. The research generates trading signals when the security price crosses the moving average line. The research demonstrated that the strategy has a better performance in terms of its return, volatility and maximum drawdown, compared to a simple buy and hold strategy that most retail investors use (Kilgallen, 2012). However, the study lacks a thorough investigation into the strategy by empowering a long-only strategy, meaning no short-selling is done when the security price falls below the moving average line. By not considering short-selling in the strategy, it is assumed that the security or the market will exhibit an upward trend in the long run. Moreover, transactional costs are not considered in this research. With the moving average strategy being prone to generating many unprofitable signals when security prices move closely to the moving average line, the strategy may incur a lot of transactional costs. This hinders the practicability of the strategy to be applied and executed by a retail investor.

Chu et al. (2020) have researched the momentum trading strategy by applying exponential moving average on cryptocurrencies. It focuses on a high-frequency trading strategy using moving average lines to generate directional signals. The study shows the applicability of moving average in cryptocurrency trading. Signals generated by the moving average is positively correlated with the corresponding movement of the cryptocurrency prices. However, the high-frequency nature of the study limits the feasibility to be used by a retail investor. The study also did not simulate the performance of the algorithm under the consideration of transactional costs.

Shen et al. (2012) have researched different machine learning algorithms to forecast the stock market. The research focused on trend prediction and the prediction accuracy of various machine learning models. However, their proposed trading model is relatively simple, where the trading decisions are purely based on the model predictions. Moreover, tax and transaction fees are not considered. Subsequent updates in the model predictions, stop loss and exit strategies are neglected.

Lv et al. (2019) have researched different machine learning and deep learning algorithms, using 44 technical indicators as the input to predict stock prices. The research used different metrics including the annualized rate of return, Sharpe ratio, win rate and maximum drawdown, setting a reference for using similar metrics in this project. The research found that machine learning algorithms produce higher returns and Sharpe ratios compared to the index. Although the research has also taken transactional costs into account (Lv et al., 2019), it purely focused on the U.S. stock market and the Chinese A-shares market.

Madan et al. (2015) have researched the feasibility to trade Bitcoin automatically using machine learning algorithms. Specifically, the study used Bitcoin price data and considered 26 cryptocurrency metadata, including block size, hash rate and number of transactions per day as

the data source. The research found out that two machine learning models achieved an accuracy exceeding 0.94. However, the future work of the research focused on clustering the data patterns inside different subsets within the data source, instead of building a trading algorithm and evaluate its profitability.

Huang and Wang (2016) researched the profitability of mean reversion trading strategy, by modelling the residuals of the stock return as a Ornstein-Uhlenbeck process and estimate the model parameters using maximum likelihood estimation. The research found out that some pairs of stock could obtain an extraordinarily high Sharpe ratio. However, the trading signals generated in the research is based on an arbitrary threshold given the output of a standardized score. This may limit the trading performance because the true distribution of the process can change over time, in which the optimal threshold needs to be dynamically adjusted.

3. Methodology

The methodology of the project is outlined in this section. In *Section 3.1*, the scope of research of the project is introduced, followed by the back-testing platform chosen in *Section 3.2*. Throughout the research process, a stringent procedure outlined in *Section 3.3* is conducted. Finally, in *Section 3.4*, the evaluation metrics are introduced, which are used to quantitatively evaluate the performance of various trading strategies researched by this project.

3.1 Scope of Research

In this project, trading algorithms for the U.S. stock market and the cryptocurrency market are considered. Since the U.S. stock market is a mature market where institutions are actively participating in, it can serve as a baseline for evaluating the performance of various algorithms against professional traders. Cryptocurrency is a relatively new market without institutional investors, where we can investigate the performance of trading strategies against retail traders. In both the U.S. stock market and the cryptocurrency market, the data sources are readily available. Thus, these markets are chosen for further research and investigation.

Difference financial securities are chosen as the back-testing security in the U.S. stock market and the cryptocurrency market. Within the U.S. stock market, the SPDR S&P 500 ETF Trust (SPY) was used to back-test and optimise the strategies. In addition, the top 10 constituents of SPY by market capitalization (AAPL, MSFT, AMZN, NVDA, GOOGL, TSLA, META, BRK.B, XOM, UNH) were used for evaluation. Within the cryptocurrency market, Bitcoin (BTC) was used for back-testing and optimising strategies. In addition, the remaining top 10 cryptocurrencies (other than BTC) by market capitalization (ETH, BNB, XRP, ADA, DOGE, SOL, MATIC, LTC, TRX, AVAX) were used for evaluation.

The back-testing period in the U.S. stock market and the cryptocurrency market are 10 years and 3.25 years respectively. For U.S. stocks, the back-testing period spans from January 1, 2013 to December 31, 2022. For cryptocurrencies, the back-testing period will span from September 1, 2019 to December 31, 2022. The discrepancy in the duration of back-testing periods in the U.S. stock market and the cryptocurrency market is due to the availability of cryptocurrency data. Despite the discrepancy in the duration, the number of back-testing trading hours are balanced in both the U.S. stock market and cryptocurrency market. This is due to the fact that U.S. stocks trade 6.5 hours from market open (9:30 a.m.) to market close (4:00 p.m.) per trading day, while cryptocurrencies trade 24 hours a day. Moreover, for each trading year, the U.S. stock market has an average of 113 calendar days as trading holidays while the cryptocurrency market has no trading holidays. Thus, the trading hours that are back-tested in both markets are roughly equivalent.

For both asset classes, the forward testing period will be January 1, 2023 onwards. The separation of back-testing and forward testing in both markets ensure that the trading strategies developed by the project would not overfit to the back-testing period. This is to ensure the robustness of the algorithms and to enhance the generalizability of the strategies under different market conditions.

3.2 Back-testing Platform

QuantConnect is chosen as the back-testing platform after comparing with three other popular platforms available to retail investors, which are Zipline, BackTrader and QuantRocket. When compared to Zipline and BackTrader, QuantConnect offers built-in historical datasets on market data, eliminating the need for external data purchases. In terms of functionality, both QuantConnect and QuantRocket provide similar features; however, QuantConnect stands out as it offers cryptocurrency data and tick-level data for U.S. stocks, which QuantRocket lacks. Therefore, QuantConnect was chosen as the primary back-testing platform for this project.

3.3 Research Procedure

A standardized research procedure consisting of three steps has been established to ensure equal comparisons among different algorithms and markets.

Firstly, the algorithms will be developed by referencing the original research study and adjusted according to the data available on the QuantConnect platform. QuantConnect is one of the best cloud-based trading platforms that provides both stock and cryptocurrency data. This back-testing environment allows us to develop and back-test algorithms in a collaborative environment, where members of the project are allowed to perform tasks simultaneously.

Next, the algorithms will be back-tested on the QuantConnect platform with SPY and BTC. The parameters of the algorithms will be optimized and fine-tuned iteratively during the back-testing process according to the evaluation metrics outlined in *Section 3.4*.

Finally, the algorithms will be back-tested on the U.S. stocks and cryptocurrencies listed in *Section 3.1*, using the same back-testing period for evaluation. The results will be recorded and compared to the benchmarks, which are the cryptocurrency large-cap index and the S&P 500 index respectively.

3.4 Evaluation Metrics

Quantifiable metrics are used to assess the risk-adjusted returns and evaluate the capacity to withstand different market situations of the algorithms developed by the project. There are five major metrics for performance evaluation, including (1) annualized return, (2) Sharpe ratio, (3) Alpha, (4) maximum drawdown and (5) win rate.

3.4.1 Annualized Rate of Return

The annualized rate of return (annualized return) measures the percentage gain or loss of an algorithm. The profitability of the algorithm can be evaluated based on this metric.

$$R_{annualized} = (1 + R_{net})^{\frac{1}{n}} - 1 \quad (1)$$

Referring to Equation (1), the annualized rate of return ($R_{annualized}$) is calculated by annualizing the net return (R_{net}), which is the overall return (Cuthbertson et al., 2010). The overall return is the percentage gain (or loss) of the algorithm throughout the back-testing period. As the annualized return is adjusted according to the investment horizon (n), it can ensure a fair evaluation on back-testing results between the U.S. stock market and cryptocurrency market.

3.4.2 Sharpe Ratio

Sharpe ratio is the reward-to-risk ratio of the strategy. It determines the risk-adjusted returns of the algorithm.

$$Sharpe\ Ratio = \frac{R_{net} - R_f}{\sigma_R} \tag{2}$$

Equation (2) measures the reward-to-risk ratio. In this equation, the reward (return) is calculated as the difference between the net return (R_{net}) and the risk-free rate (R_f). Risk is defined by the standard deviation (σ_R) of the net return (Cuthbertson et al., 2010). The risk-free rate (R_f) is the rate of return of risk-free assets, which is typically estimated using U.S. Treasury Bonds as they are considered nearly risk-free.

3.4.3 Alpha

Alpha is often referred to as the abnormal return generated. It is defined as the incremental return generated in excess of the market return (Cuthbertson et al., 2010). Alpha (α) shows the degree of outperformance of a trading strategy against the market portfolio. Generally, a larger Alpha implies a better performing strategy. Alpha is represented by the following equation.

$$\alpha = R_{net} - R_f - \beta\ (R_m - R_f) \tag{3}$$

Referring to equation (3), R_{net} is the actual return of the evaluated strategy, R_f is the risk-free rate, R_m is the return of the market portfolio and β is the portfolio beta. Beta (β) is a term that is proportional to the correlation between the actual return and the market return.

3.4.4 Maximum Drawdown

The maximum drawdown is defined as the maximum drop in value of the investment, as shown in equation (4). It measures the downside risk of the algorithm.

$$Max\ Drawdown = \frac{Peak\ value\ - Trough\ value}{Peak\ value} \tag{4}$$

3.4.5 Win Rate

Win rate is defined as the percentage of profitable trades. It identifies the rate of success of a strategy. Win rate can be calculated with equation (5).

$$Win\ Rate = \frac{Number\ of\ trades\ with\ positive\ return}{Total\ number\ of\ trades} \tag{5}$$

It is a common metric to analyze the competence of trading strategies disregarding the return of the trades (Cuthbertson et al., 2010).

4. Trading Algorithms

Developing trading algorithms is an essential part for this project, as various parameters need to be optimized before performing subsequent evaluation. Moreover, the trading algorithms detailed in this section constitutes a major deliverable of the project.

As of the submission date of the report, four trading strategies have been developed. These strategies include the Moving Average Strategy (*Section 4.1*), the Machine Learning Strategy (*Section 4.2*), the Crypto Metadata Strategy (*Section 4.3*) and the Reversion Strategy (*Section 4.4*).

4.1 Moving Average Strategy

Based on Kilgallen (2012)'s study on the moving average strategy, this project has implemented the logic and calculations of the strategy accordingly. Five versions of enhancements have been developed and implemented to the strategy.

The moving average strategy involves generating trading signals with a simple moving average line. A *n*-day moving average (MA) is calculated by Equation (6).

$$\text{Moving Average}\ (n) = \frac{\mathrm{P_t} + \mathrm{P_{t-1}} + \cdots + P_{t-n+2} + P_{t-n+1}}{n} \tag{6}$$

The main idea behind the strategy is the momentum of the underlying security's price movement, meaning the security's price will move in the same direction following a trend.

4.1.1 Moving Average Version 0

Version 0 is the implementation based on Kilgallen (2012)'s study on the strategy and adding the short-selling feature to the algorithm. This allows the strategy to profit from downside risks of the security and to avoid bias to the upside of the markets. Version 0 serves as a baseline and comparison benchmark for further improvements to the algorithm.

4.1.1.1 Trading Logic

The trading logic follows Kilgallen (2012)'s study while incorporating the short-selling feature. The trading logic for the moving average strategy is as follows:

```
if price > moving average then
   if current position <= 0 then
      Buy back all current position
      Buy
else
   if current position >= 0 then
      Sell all current position
      Short Sell
```

4.1.1.2 Results

Table 1: Back-testing Results of Moving Average Trading Strategy version 0

	Version 0	
Evaluation Metrics	Cryptocurrency	U.S. Stock
Annualized Rate of Return	87.7%	3.4%
Sharpe Ratio	1.49	0.148
Drawdown	50.0%	26.1%
Win Rate	9%	17%
Alpha	0.682	0

The trading logic has been back-tested and the results are shown in **Table 1**. The result represents an optimized algorithm based on the logic. A 100-day MA is used in U.S. stock market and a 50-day MA is used in cryptocurrency market.

4.1.1.3 Key Findings and Evaluation

In the cryptocurrency market, version 0 has generated exceptional performance that is better than the underlying, which is BTCUSD. The annualized return and Sharpe ratio is much higher than the underlying. However, a huge drawdown and a low win rate is observed, showing the need for further development on the algorithm.

In the U.S. stock market, version 0 has a poor performance with a low annualized return, Sharpe ratio and win rate. The drawdown of the algorithm is also large compared to the market. This shows the lower applicability of momentum trading to the U.S. stock market. Moreover, due to its lower volatility, the algorithm is more prone to generating unwanted and unprofitable trades, due to the problems in version 0 mentioned below.

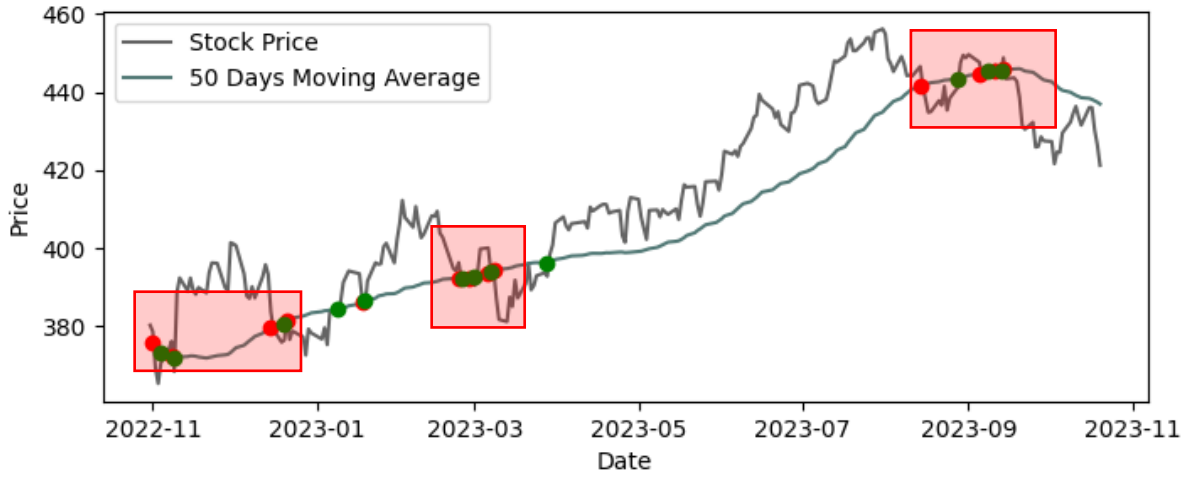

Figure 1: 50-Days Simple Moving Average plotted on the SPY, with labels of a major problem of frequent crossing around the MA. Green and Red dots are simulated signals. Calculated and plotted using Python codes with historical market data obtained from Yahoo Finance.

The first major problem of version 0 is that the algorithm tends to generate frequency trades with negative or close to zero return when the security prices move along the moving average line. For illustration, **Figure 1** shows the simulated trading signals generated with a 50-day MA, where the problem of a negative return generation occurs more often than a 100-day MA. The red-colored boxes indicate the instances when the stock price moves around the moving average line, leading to the generation of numerous trades. These trades are labelled and represented by the green dots and red dots, where green dots are buy signals and red dots are sell signals. Most of these trades have a negative return as they tend to "buy high, sell low". The reason accounting for such negative return is due to the trading logic of buying when the price is above the moving average and selling when the price is lower than the moving average. The "buy high, sell low" back-tested executions are shown in the yellow-colored boxes in **Figure 2**, where a 100-day MA is used. Moreover, a lot of transactional costs will be generated by these unprofitable trades.

***Figure 2**: 100-days simple moving average plotted on SPY, with labels of major problems on the lagging properties of MA and the frequent "Buy high, sell low" trades. Green and red dots represent buy and sell executions in back-testing respectively.*

The second major problem of version 0 is that the algorithm is unable to capitalize on opportunities brought about by the "lagging" property of moving average. As shown in the red boxes in **Figure 2**, the algorithm missed out major opportunities to make large profits. It is because the moving average does not move closely with the stock price. Consequently, before closing the position, the stock price will significantly move against the positions taken by the algorithm, incurring huge drawdowns.

Although the moving average strategy version 0 sometimes has a worse performance than the market, several critical problems are identified which laid the groundwork for subsequent variants and enhancements of the moving average strategy.

4.1.2 Moving Average Version 1

Version 1 is the implementation based on an update to enhance version 0. Targeting the major problem of having frequent unwanted trades with negative or no profits when the price is close to the MA, an idea of creating a pair of bands around the MA is formulated.

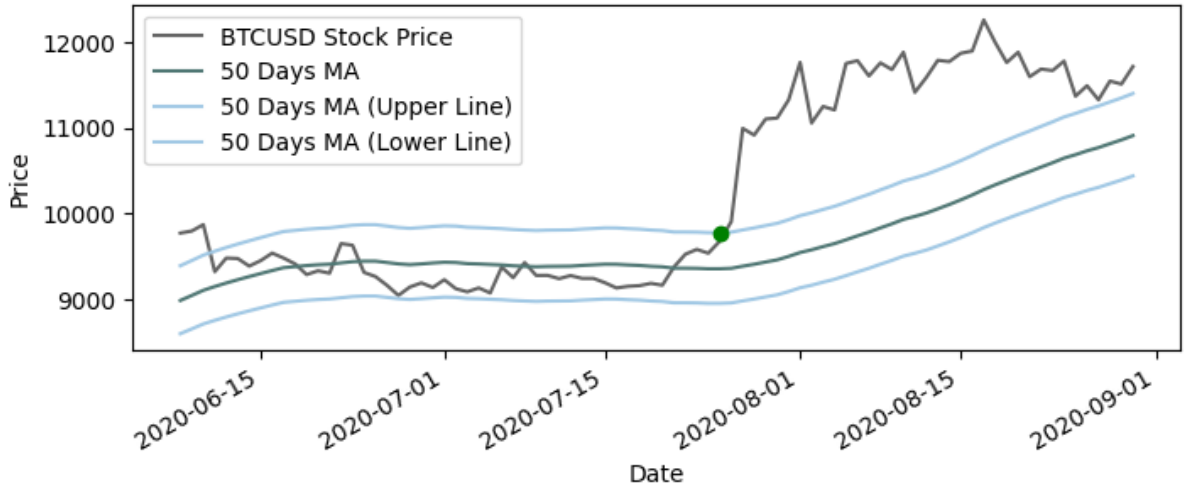

Figure 3: A simulated 50-days simple moving average with an upper and lower band plotted on BTCUSD

The pair of bands, as illustrated in **Figure 3**, could avoid the frequently unwanted trades when the price moves around the MA. Nevertheless, it can still provide a good entry point when the price moves out of the band, as illustrated by the green dot in **Figure 3**.

Hence, version 1 is created by shifting the moving average line up and down by a certain fixed percentage (k). The formula is shown in equation (7) and (8).

$$\text{Upper Band} = \text{Moving Average}\ (n) \times (1 + k\%) \quad (7)$$

$$\text{Lower Band} = \text{Moving Average}\ (n) \div (1 + k\%) \quad (8)$$

4.1.2.1 Trading Logic

The trading logic of version 1 is updated by introducing the new feature, illustrated as follows:

```
if current position = 0 then
    if price > moving average * (1+k%) then
        Buy
    else if price < moving average / (1+k%) then
        Short Sell
else
    if current position < 0 and price > moving average then
        Buy Back all current position
    else if current position > 0 and price < moving average then
        Sell all current position
```

Version 1 uses the original MA for closing out positions (buy back and sell) to avoid frequent trades happening around the upper or lower band.

4.1.2.2 Results

The trading logic of version 1 has been back-tested and the results are shown in **Table 2**.

Table 2: Back-testing Results of Moving Average Trading Strategy version 1

	Version 1	
Evaluation Metrics	Cryptocurrency	U.S. Stock
Annualized Rate of Return	115.1%	5.8%
Sharpe Ratio	1.855	0.365
Drawdown	39.6%	19.1%
Win Rate	19%	34%
Alpha	0.856	0

4.1.2.3 Key Findings and Evaluation

The results showed a major improvement in all of the metrics in both the U.S. stock market and cryptocurrency market, compared to version 0.

In the cryptocurrency market, the Sharpe ratio, annualized return and Alpha increased significantly. The drawdown has also decreased, which is believed to be the main factor in improving the return while reducing the losses caused by frequent unwanted trades.

In the U.S. stock market, the Sharpe ratio and annualized return also increased significantly. The drawdown has also decreased due to the same factor.

The major finding is the dramatical increase in the win rate, which doubled in both markets compared to version 0. It is believed that the increased win rate is due to the reduction in the unwanted trades, which are usually losses. This shows the robustness of the improvement and the importance of using a pair of bands in the trading logic, instead of a single moving average as conducted in version 0.

4.1.3 Moving Average Version 2

Version 2 is implemented based on an idea to test for a variant of version 1. In version 1, the original MA is used for closing out positions (buy back and sell) while the band is used for opening positions. The idea of using the bands (instead of the original MA) for closing out positions has emerged, which is implemented in version 2.

The idea is based on closing out positions early to generate high profits and reduce losses when the market moves against the positions of the algorithm. However, it is at risk of having the same problem of losses and transactional costs in frequent unwanted trades when prices move around the upper or lower band.

4.1.3.1 Trading Logic

The trading logic is updated by incorporating the use of moving average band in closing out positions. The trading logic for version 2 is as follows:

```
if current position = 0 then
    if price > moving average * (1+k%) then
        Buy
    else if price < moving average / (1+k%) then
        Short Sell
else
    if current position < 0 and price > moving average / (1+k%)  then
        Buy Back all current position
    else if current position > 0 and price < moving average * (1+k%)  then
        Sell all current position
```

4.1.3.2 Results

The trading logic of version 2 has been back-tested and the results are shown in **Table 3**.

Table 3: Back-testing Results of Moving Average Trading Strategy version 2

	Version 2	
Evaluation Metrics	Cryptocurrency	U.S. Stock
Annualized Rate of Return	109.1%	5.4%
Sharpe Ratio	1.797	0.336
Drawdown	32.2%	19.3%
Win Rate	15%	21%
Alpha	0.798	0

4.1.3.3 Key Findings and Evaluation

The results of version 2 failed to show a significant improvement compared to version 1. The annualized rate of return, Sharpe ratio and win rate all decreased compared to version 1. However, the drawdown in cryptocurrency has decreased significantly, showing that the original idea of version 2 has been implemented correctly.

The decrease in performance is mainly due to the frequently occurring trades when the stock price moves around the upper or lower band of the moving average. From **Figure 4**, the buy and sell trades from the back-test showed that when the price moves around the band, frequent "buy high, sell low" trades occurred, causing the algorithm to lose money. Compared to version 0, this is also the problem faced by version 2 of the moving average strategy.

Figure 4: Back-tested buy and sell trades on BTCUSD with version 2, plotted with 50 days MA and a band of upper and lower MA

However, version 2 provides valuable insights as it also successfully closes out the trades earlier than version 1, thus earning more profits in some of the trades. This has the potential for further development by reducing the number of unwanted trades, while successfully capturing the profits in trades by betting on the momentum. Although version 2 has a lower performance than version 1, the concept and idea behind version 2 is being adopted in further development.

4.1.4 Moving Average Version 3

Version 3 is considered as an enhancement to version 2. In version 1 and version 2, the value of k, which is the percentage above or below the MA to generate the lower and upper band, is arbitrary and fixed. This limits the ability of the algorithm to cater for different market situations. In this version, the percentage k is determined based on a fraction of the volatility in the underlying price movement for the last x days. The MA bands are generated based on the original MA with the dynamically calculated percentage k.

The idea behind version 3 is that the algorithm should be able to ignore the normal intraday or short-term volatility in the underlying security, but to trade on the momentum when the underlying security has a genuine and large movement. In times of lower volatility, the algorithm should have a narrower MA band and enter the trades early when the underlying breaks through the band. While in times of higher volatility, the algorithm should have a wider MA band to allow the underlying price to retreat and move around without triggering the algorithm to trade. This allows the algorithm to preserve its position on momentum while the stock price retreats momentarily. Moreover, during times of high volatility, the algorithm can prevent entering trades by setting a wider MA band, potentially reducing the problem of version 2. This allows the algorithm to cater for different market conditions.

4.1.4.1 Trading Logic

The trading logic is updated by incorporating the use of volatility in generating the MA bands. The trading logic for version 3 is as follows:

c = volatility coefficient (pre-specified)

σ = standard deviation of the percentage change in the close price for the last x days

$k = c\sigma$

if current position = 0 **then**
    if price > moving average * (1+k) **then**
        Buy
    else if price < moving average / (1+k) **then**
        Short Sell
else
    if current position < 0 **and** price > moving average / (1+k) **then**
        Buy Back all current position
    else if current position > 0 **and** price < moving average * (1+k) **then**
        Sell all current position

4.1.4.2 Results

The trading logic of version 3 has been back-tested and the results are shown in **Table 4**.

Table 4: Back-testing Results of Moving Average Trading Strategy version 3

	Version 3	
Evaluation Metrics	Cryptocurrency	U.S. Stock
Annualized Rate of Return	108.1%	7.8%
Sharpe Ratio	1.812	0.574
Drawdown	36.4%	10.3%
Win Rate	12%	24%
Alpha	0.769	0

4.1.4.3 Key Findings and Evaluation

The performance of version 3 has shown improvement from version 2. For the U.S. stock market, all evaluation metrics improved from version 2, which there is a significant increase in Sharpe ratio and Annual Return, and a significant decrease in drawdown. Version 3 also showed improvements compared to version 1 in most metrics. For the cryptocurrency market, the evaluation metrics showed slight improvements from version 2, mainly in the improved Sharpe ratio. Most of the other evaluation metrics stayed similar.

Figure 5: Comparison of executions of trades in version 2 and version 3

The improved Sharpe ratio and annualized return is mainly due to the capability of the algorithm to cater to different market situations. The version 3 algorithm is able to generate trading signals at a better time. From **Figure 5**, the version 2 algorithm used a fixed width of the MA band, which caused the algorithm to buy back early and executed its buy order at a higher price. The version 3 algorithm used a dynamic band, allowing the algorithm to narrow its MA band when the volatility drops, preventing it from closing the trades early. The figure also shows that the MA band widens when the volatility increases at around 20 July 2021 as the price moves against the position of the algorithm. Thus, in version 3, the algorithm could close the position earlier at a lower price compared to maintaining a fixed width of the MA band in version 2.

Version 3 showed significant improvements in the U.S. stock market by catering to different market conditions. It also proved that the performance of the algorithm is improved by introducing the concept of using the volatility in price change to create the MA band. This provides the groundwork for further improvements to version 3.

4.1.5 Moving Average Version 4

Version 4 enhances version 3 further by using a different MA and volatility calculation for generating signals to close the trades. Each MA can be calculated based on a different number of days. In total, there are two MAs, each with an associated upper and lower band.

The rationale behind this enhancement is based on the problem faced by version 2, of having frequent unwanted trades generated when the price of the security moves around the upper or lower band of the MA. This causes losses and high transactional costs to the algorithm, as the same MA, upper band and lower band is used to generate trading signals for opening and closing the trades. The idea of using 2 separate bands is tested to solve the problem.

4.1.5.1 Trading Logic

The trading logic is updated by introducing a new MA, upper band and lower band for closing the trades. The logic also separates the moving average and volatility calculation for opening and closing trades. The trading logic for version 4 is as follows:

c_1 = volatility coefficient (pre-specified) for opening trades

σ_1 = standard deviation (volatility) of the percentage change in the close price for opening trades

$k_1 = c_1\sigma_1$

c_2 = volatility coefficient (pre-specified) for closing trades

σ_2 = standard deviation (volatility) of the percentage change in the close price for closing trades

$k_2 = c_2\sigma_2$

if current position = 0 **then**
    if price > moving average (open trades) * (1+k_1) **then**
        Buy
    else if price < moving average (open trades) / (1+k_1) **then**
        Short Sell
else
    if current position < 0 **and** price > moving average (close trades) / (1+k_2) **then**
        Buy Back all current position
    else if current position > 0 **and** price < moving average (close trades) * (1+k_2) **then**
        Sell all current position

4.1.5.2 Results

The trading logic of version 4 has been back-tested and the results are shown in **Table 5** on the next page.

Table 5: Back-testing Results of Moving Average Trading Strategy version 4

	Version 4	
Evaluation Metrics	Cryptocurrency	U.S. Stock
Annualized Rate of Return	120.1%	8.7%
Sharpe Ratio	1.961	0.893
Drawdown	37.6%	6.3%
Win Rate	19%	30%
Alpha	0.856	0

4.1.5.3 Key Findings and Evaluation

From **Table 5**, the results of version 4 showed a significant improvement from version 3. In both markets, the Sharpe ratio, the annual return and the win rate improved significantly.

However, the optimized results in **Table 5** did not use the full implementation of version 4, by using the same MA for opening and closing trades.

Figure 6: Back-tested "Unwanted" trades generated by the algorithm with different MA for opening and closing trades, with green and red dots representing buy and sell executions respectively

It is found out that using 2 different moving averages will cause an abundance of "unwanted" trades, due to the misalignment between the 2 moving averages. The red box in **Figure 6** shows when the upper band of the MA for closing trades is higher than that for opening trades, all buy positions will be closed quickly. This is caused by the trading logic of closing the buy positions when the security price is below the upper band of the closing MA. With this misalignment, the algorithm generates a lot of "unwanted" trades, which will reduce the return and increase the transactional costs. Therefore, the same MA should be used for opening and closing trades.

However, it is discovered that, when using the same MA, a different calculation for the MA band for closing the trades can increase the performance. Specifically, a smaller volatility coefficient is used for calculating the width of the closing MA band. It is believed this solves the problem of version 2, by reducing the "unwanted" trades when the security price moves around the moving average band. Nevertheless, version 4 provided valuable insights for future development.

4.1.6 Moving Average Version 5

Version 5 is developed based on the insights from version 4. Only one MA will be used with a pair of MA bands, one above the MA and the other below the MA. The bands will be used for opening and closing trades. Throughout the previous versions, the number of days inside the MA is fixed. Version 5 introduces a dynamic MA instead of a static MA, so that the strategy can cater for different types of price movements.

The idea behind version 5 is to take price momentum into consideration and utilize the historical price information to determine the length of the MA. For instance, a longer MA is used to capture the momentum and not to close the trades. On the other hand, a shorter MA is used to close the trade quickly when the momentum is lost. The determination of the MA length is based on the local peaks and troughs in the historical price chart.

4.1.6.1 Trading Logic

With an extra logic implemented to determine the length of the two MAs dynamically, the trading logic for version 5 can be separated into two parts.

The first part involves calculating the length of the MA dynamically. The simplified calculation of the number of days used in the MA is as follows:

c_{MA} = coefficient (pre-specified) for calculating the MA

high_points = a list of local peaks based on the price chart

high_MA = average number of days between each local peaks in high_points

low_points = a list of local troughs based on the price chart

low_diff = average number of days between each local troughs in low_points

final_MA = int($c_{MA} *$ average(high_MA, low_MA))

For the second part of the trading logic, it will use an updated version of the trading logic in version 4. For opening and closing trades, it will use the same MA (using the same number of days calculated in final_MA).

c_1 = volatility coefficient (pre-specified) for opening trades

σ_1 = standard deviation (volatility) of the percentage change in the close price for opening trades

$k_1 = c_1\sigma_1$

c_2 = volatility coefficient (pre-specified) for closing trades

σ_2 = standard deviation (volatility) of the percentage change in the close price for closing trades

$k_2 = c_2\sigma_2$

if current position = 0 **then**
  if price > moving average (final_MA) * (1+k_1%) **then**
    Buy
  else if price < moving average (final_MA) / (1+k_1%) **then**
    Short Sell

else
  if current position < 0 **and** price > moving average (final_MA) / (1+k_2%) **then**
    Buy Back all current position
  else if current position > 0 **and** price < moving average (final_MA) * (1+k_2%) **then**
    Sell all current position

4.1.6.2 Results

The trading logic of version 5 has been back-tested and the results are shown in **Table 6**.

Table 6: Back-testing Results of Moving Average Trading Strategy version 5

	Version 5	
Evaluation Metrics	Cryptocurrency	U.S. Stock
Annualized Rate of Return	132.2%	8.8%
Sharpe Ratio	2.078	0.901
Drawdown	42.3%	7.0%
Win Rate	18%	31%
Alpha	0.947	0

4.1.6.3 Key Findings and Evaluation

Version 5 has improved from version 4 and it has the best evaluation results among all versions. This shows that with a dynamic MA, the algorithm can better cater to different market situations.

For the U.S. stock market, version 5 has the highest Sharpe ratio and annual return, while maintaining a low drawdown and high win rate. It is mainly due to the macroeconomic environment of the U.S. stock market, which has always been changing in the past. Having a dynamically set moving average can earn a higher profit in different market conditions.

For the cryptocurrency market, version 5 has the highest Sharpe ratio, annual return and Alpha, showing its robustness in different market conditions.

The major improvement of version 5 is that the dynamic MA allows the algorithm to use a longer MA during momentum, while using a shorter MA during range period. During momentum periods, where stock prices move a lot in the same direction, there are less peaks and troughs, thus a longer MA is used. It allows the algorithm to hold its position during small reversion in the momentum events and earn higher profits from the momentum of the security. During range events, where the security price moves within a certain range, there are many peaks and troughs. A shorter MA will be used such that the algorithm can close out positions as the momentum has lost, as well as to open trades earlier when the security breaks through the range.

4.1.6.4 Black Swan Analysis

COVID Lockdown:

BTCUSD: -46.3% in 4 days

Algorithm: +30.5% in 4 days

Negative news from Elon Musk and Chinese government:

BTCUSD: -37.1% in 9 days

Algorithm: +21.4% in 9 days

Rate hike and heavy regulations on crypto:

BTCUSD: -28.2% in 5 days

Algorithm: +21.0% in 5 days

FTX Collapse:

BTCUSD: -25.6% in 3 days

Algorithm: -11.3% in 3 days

Figure 7: Black Swan analysis with major BTC crashes, showing the drawdown in BTC comparing to the return/loss of the algorithm

From **Figure 7**, different major BTC crashes are shown. Version 5 has profited largely in 3 of the crashes, when BTC has dropped by around 30% to 45%. In the recent major event of FTX collapse, the algorithm was able to reduce the loss to only 11.3% compared to BTC's drop of 25.6%. This is mainly due to the rapid drop of BTC, which caused the algorithm being unable to open a short position on BTC after closing its current position. The algorithm currently refreshes at a 1-hour frequency and it is unable to react timely in such rapid events. Therefore, it can only reduce the loss but is unable to profit from the FTX collapse event. Future developments are needed to capture such rapid events.

4.1.7 Live Trading

The moving average strategy was deployed for live trading. This allows a more comprehensive evaluation by taking into consideration of effects in real live trading, including the market impact, depth of order book, real live transactional costs and the leakage of information.

The version 5 moving average strategy was deployed on BTCUSDT (equivalent to BTCUSD) on a Binance margin account. The live trading started on 8 January 2024 with USD 961 investment (around 7500 HKD).

As of submission date, the algorithm has a 0.46% loss, with the results plotted in **Figure 8**. The algorithm currently has a short position on BTC. The algorithm will continue to live trade until the end of the final year project for a better evaluation on the algorithm.

Figure 8: Live Trading Results on BTCUSDT with Version 5 as of 20 Jan 2024

4.1.8 Summary of Moving Averages

Table 7: Heat map of all versions of moving average strategy

Asset	Metrics	v0	v1	v2	v3	v4	v5
Crypto	Sharpe	1.49	1.855	1.797	1.812	1.961	2.078
	Annual return	87.7%	115.1%	109.1%	108.1%	120.1%	132.2%
	Drawdown	50.0%	39.6%	32.2%	36.4%	37.6%	42.3%
	Win Rate	9%	19%	15%	12%	19%	18%
	Alpha	0.682	0.856	0.798	0.769	0.856	0.947
Stock	Sharpe	0.148	0.365	0.336	0.574	0.893	0.901
	Annual return	3.4%	5.8%	5.4%	7.8%	8.7%	8.8%
	Drawdown	26.1%	19.1%	19.3%	10.3%	6.3%	7.0%
	Win Rate	17%	34%	21%	24%	30%	31%
	Alpha	0	0	0	0	0	0

From **Table 7**, the enhancements in each version have contributed to the improvement in the performance of the strategy. Version 5, with the most enhancements, showed the best performance compared to other versions. All versions of the moving average strategy are also better than the strategy in the original study.

4.2 Machine Learning Strategy

The second type of strategies developed is based on various machine learning algorithms, including K-Nearest Neighbours (KNN), Support Vector Machines (SVM), Decision Tree (DT), Random Forest (RF), Gradient Boosting (GB) and Logistic Regression (LR). These machine learning models and algorithms are widely used in practice. This project has developed a trading strategy based on various technical indicators to trade on both U.S. stock and cryptocurrency, replicating a similar strategy that was conducted by Lv et al. (2019).

4.2.1 Training Process

During the training process for each machine learning model, different technical indicators were included as features in the dataset. These technical indicators include the 50, 100 and 200-day moving average (MA), the Moving Average Convergence/Divergence (MACD), the Relative Strength Index (RSI), the 1-day and 5-day price percentage change, and the percentage differences between the close price and the moving averages. The output variable is a variable with three classes: buy signal (1), hold signal (0) or sell signal (-1). This variable is created the following procedure:

Input: A 2D array of technical indicators and price percentage change for each trading day
Output: A 1D array of trading signals
for $i = 1$ to T **do**
  calculate $d =$ 5-day price percentage difference
  if $d \in (-3\%, 3\%)$ **then**
    $y[i] = 0$
  else if $d < -3\%$ **then**
    $y[i] = -1$
  else
    $y[i] = 1$
return y

The sample size of the data in the U.S. stock market and cryptocurrency market are three trading years and one trading year respectively.

After aggregating the data with technical indicators, the data (X, y) is then trained and fitted to each of the machine learning models specified above. X is the trading day data aggregated with technical indicators and y is the corresponding buy/hold/sell signal. The model is retrained at regular intervals.

4.2.2 Trading Logic

The trading signals generated by various machine learning models is then used to facilitate the trading algorithm. Instead of solely relying on predicted signal of the models, other factors are taken into account before trading.

The trading algorithm for the machine learning strategy is as follows:

```
trading_signal = model.predict(latest_technical_indicator_data)
if currently has underlying position then
    if trading_signal aligns with the current underlying position then
        Extend the holding time
    else
        Close the position
else
    Trade according to the signal predicted
```

4.2.3 Comparison with Existing Research

The strategy developed by the project provides a more comprehensive and extensive approach compared to existing research.

In terms of the algorithm implementation, some academic research only considered model accuracy (Madan et. al, 2015) which may not truly reflect the profitability of the trading algorithm. This project considers the implementation of the trading algorithm on top of the accuracy of various machine learning models. Specifically, the implementation of the algorithm in *Section 4.2.2* considers subsequent updates in the predicted signal, which significantly affects the profitability and Sharpe ratio of the trading algorithm.

Moreover, the timing of when to buy or sell the underlying asset is considered in the project. Some existing research may not consider the time when the buy or sell order is placed. For instance, a buy order placed during market open may have a different price executed compared to the same order placed just before market close, which slightly affects the calculation of the Sharpe ratio and other evaluation metrics. The project considers whether the buy or sell order should be placed during market open or market close. This consideration is considered as an improvement because it can better mimic the real-life trading situation across both the U.S. stock and the cryptocurrency market.

The trading signal history is also considered in the project. Trading algorithms may implement different approaches when there are changes in trading signals. For example, it is possible that an algorithm only considers the latest predicted signal to take positions during trading. This project enhances the profitability of the algorithms by consider the history of predicted trading signals. Specifically, if there is a change in the trading signal from buy to sell, the algorithm will close the position accordingly. This is to ensure the stability of the algorithm and consistency of different trades.

4.2.4 Back-testing Results

Table 8 summarizes the back-testing results for the machine learning strategy.

Table 8: Back-testing results of the Machine Learning Strategy

Asset	Metric	KNN	SVM	DT	RF	GB	LR
U.S. Stock	Sharpe	0.179	0.081	0.252	0.441	0.391	0.082
	Return	3.82%	2.27%	5.04%	7.24%	8.12%	2.31%
	Drawdown	21.5%	13.5%	29.7%	13.5%	28.2%	22.4%
	Win Rate	32%	56%	39%	60%	55%	51%
	Alpha	-0.015	-0.002	0.02	0.032	-0.019	-0.025
Crypto	Sharpe	1.913	1.271	0.755	0.825	0.628	0.023
	Return	99.7%	114.1%	23.21%	25.22%	23.77%	-8.69%
	Drawdown	55.4%	19.9%	31.5%	17.3%	74.7%	59.6%
	Win Rate	29%	54%	45%	49%	33%	48%
	Alpha	0.7	0.242	0.15	0.195	0.091	-0.094

4.2.5 Key Findings and Evaluation

From the back-testing results outlined above in **Table 8**, several findings can be drawn.

The most apparent fact is that the annualized return on cryptocurrency far exceeds the return on U.S. stock. This is attributed to the idea of risk-return trade off where assets with higher return should associate with a higher risk. Since cryptocurrency price movements is more volatile than stocks, the level of risk involved in cryptocurrency is higher, which contributes to a higher return.

The machine learning strategy did not perform well as compared to the moving average strategy in terms of drawdown. Some models, including KNN, GB and LR, experienced a drawdown greater than 50% in both the stock and cryptocurrency market. These models are prone to overfitting and may not generalize well to an unseen market situation, which leads to consecutive wrong signal predictions. These false predictions caused the significant drawdown during back-testing.

Moreover, some technical indicators were not representative and did not serve as a predictive input to the model. Particularly, the RSI gives false signals of being overbought or oversold. When the RSI is below 30, it may not truly represent an oversold situation, while the model may generate a buy signal. This creates multiple false signals which further exacerbates the performance of the algorithm.

It is also notable that some algorithms only have a few trades throughout the back-test period as defined in *Section 3.1*. This is due to low model confidence which prevents trades to be executed.

4.3 Crypto Metadata Strategy

The third type of strategy is the crypto metadata strategy. It is similar to the machine learning strategy, but with the technical indicator features replaced by cryptocurrency metadata.

The trading logic behind this strategy is the assumption that some metadata features, such as the number of transactions on the Bitcoin blockchain, has an impact on the price movement on Bitcoin. Thus, it may be profitable to trade based on these features that correlates with the Bitcoin price movement.

4.3.1 Data Sources and Models Used

The data sources for this strategy include a total of 23 Bitcoin Metadata features and daily Bitcoin price data, which are all obtained from the back-testing platform QuantConnect.

A list of cryptocurrency metadata, which are used as input features to the machine learning models, are tabulated in **Table 9**[1]:

Feature	**Meaning/Explanation**
Difficulty	A relative measure of the difficulty to find a new block.
MyWalletNumberofUsers	Number of wallet hosts using the wallet service provided by QuantConnect.
AverageBlockSize	Measured in megabytes (MB).
BlockchainSize	The total size of all block headers and transactions.
MedianTransactionConfirmationTime	The median time for a transaction to be accepted into a mined block and added to the public ledger.
MinersRevenue	Total value of block rewards and transaction fees paid to miners.
HashRate	The number of tera-hashes per second that the Bitcoin network is currently performing.
CostPerTransaction	The ratio between the miners revenue to the number of transactions.
CostPercentofTransactionVolume	The ratio between the miners revenue to the transaction volume, expressed as a percentage.
EstimatedTransactionVolumeUSD	The estimated transaction value expressed in USD.

[1] Bitcoin Metadata – QuantConnect.com: https://www.quantconnect.com/docs/v2/writing-algorithms/datasets/blockchain/bitcoin-metadata

EstimatedTransactionVolume	The estimated transaction value of transactions on the Bitcoin blockchain.
TotalOutputVolume	The total value of all transactions outputs per day.
NumberofTransactionperBlock	The average number of transactions per block.
NumberofUniqueBitcoinAddressesUsed	The total number of unique addresses used on the Bitcoin blockchain.
NumberofTransactions ExcludingPopularAddresses	The total of number of Bitcoin transactions, excluding the transactions involving in the 100 most popular addresses in the network.
TotalNumberofTransactions	The total number of transactions.
NumberofTransactions	The number of daily confirmed Bitcoin transactions.
TotalTransactionFeesUSD	The total value of all transaction fees in USD paid to miners.
TotalTransactionFees	The total value of all transaction fees in Bitcoin paid to miners.
MarketCapitalization	The total USD value of Bitcoin supply in circulation, calculating across major crypto exchanges.
TotalBitcoins	The remaining supply of Bitcoins on the network.
MyWalletNumberofTransactionPerDay	Number of transactions made by wallet users per day.
MyWalletTransactionVolume	Transaction volume of the web wallet service in the past 24 hours.

Table 9: A list of Bitcoin Metadata available in QuantConnect

In terms of the models used, this strategy replicates the research done by Madan et al. (2015), where the Logistic Regression (Binomial GLM) and Random Forest models are adopted. On top of the two models above, the project has also tested the strategy on a voting classifier with hard prediction. The voting classifier combines results from the Random Forest, Logistic Regression, SVM, Gradient Boosting and KNN models.

4.3.2 Training Procedure and Trading Logic

Similar to the machine learning strategy, the output variable is created using a similar method outlined in *Section 4.2.1*. This variable is created the following procedure:

Input: A 2D array of Bitcoin metadata and price percentage change for each trading day
Output: A 1D array of trading signals
for $i = 1$ to T **do**
  calculate $d =$ 5-day price percentage difference
  if $d \in (-3\%, 3\%)$ **then**
    $y[i] = 0$
  else if $d < -3\%$ **then**
    $y[i] = -1$
  else
    $y[i] = 1$
return y

The sample size of the data is approximately one trading year, or 365 days since the cryptocurrency market trades every day, including holidays.

After aggregating the data with Bitcoin metadata, the data (X, y) is then trained and fitted to each of the machine learning models specified above. X is the trading day data aggregated with Bitcoin metadata and y is the corresponding buy/hold/sell signal. The model is retrained at regular intervals.

The trading logic for this strategy is identical to the logic outlined in *Section 4.2.2*.

```
trading_signal = model.predict(latest_technical_indicator_data)
if currently has underlying position then
  if trading_signal aligns with the current underlying position then
    Extend the holding time
  else
    Close the position
else
  Trade according to the signal predicted
```

4.3.3 Back-testing Results and Evaluation

Table 10 summarizes the best-performing back-testing results for the crypto metadata strategy.

Sharpe Ratio	Annual Return	Drawdown	Win Rate	Alpha
0.875	42.91%	75.5%	34%	0.252

Table 10: Back-testing Results of the Crypto Metadata Strategy

Although the back-testing results above shows a promising performance, there are several problems and drawbacks associated with this strategy.

The models adopted in this strategy generally do not work well. Although the model accuracy findings were consistent with the existing research, the model potentially overfits to its training data, meaning that the model is less robust and cannot generalize well to unseen market scenarios. Moreover, a small change in the model hyperparameters and algorithm parameters can result in a significantly different evaluation result. In some back-tests, the model is not confident enough to trade, resulting in only a few trades throughout the 3 years of back-test history.

Regarding the usage of Bitcoin metadata, some features are correlated with each other and affect the model performance. For example, the total transaction fees (expressed in BTC and USD) are two similar features with high correlation. Under a regression context, this may be considered as multicollinearity, which affects the model confidence to trade. With such highly correlated features, the entire dataset of 23 metadata features can be compressed without losing much variance in the data. On the other hand, some metadata features, such as total unmined Bitcoins, may not have strong correlation with Bitcoin price movement. Since the total unmined Bitcoins represents the supply of Bitcoin, this quantity is expected to monotonically decrease over time, which is expected to exhibit a mild correlation with the price movement of Bitcoin.

Furthermore, the project cannot exactly replicate the strategy done by the existing research, due to the varying data source. In the existing research, 26 metadata features are considered and 16 of them are used in modelling. While in this trading strategy, the data source from QuantConnect has only 23 of these features. This suggests a misalignment in the data source.

4.4 Reversion Strategy

The reversion strategy is a trading strategy utilized in the financial industry based on mean reversion in financial markets, where mean reversion refers to the tendency of security values to move back to their moving average levels after deviating from the averages for a period of several trading days.

The crucial underlying assumption in the reversion strategy is the belief that the asset will revert to its average position, for instance, the n-day moving average. This strategy makes a bet that the price will converge towards the average value of the asset.

4.4.1 Trading Logic

The trading logic for the reversion strategy is as follows:

k = a pre-specified number of trading days history to obtain

x = an array storing the past price-to-MA ratios

for i **in** 1 to k do

  $x[i] = \frac{Price_{t-k+i}}{MA}$

fit a statistical distribution to x

obtain the 1st, 5th, 95th and 99th percentile of the fitted distribution

calculate $y = \frac{Price_t}{MA}$ which is the latest price-to-MA ratio

if 1st percentile $< y <$ 5th percentile **then**

  Buy

else if 95th percentile $< y <$ 99th percentile **then**

  Short Sell

else if $y <$ 1st percentile **then**

  Buy back all current position

else if $y >$ 99th percentile **then**

  Sell all current position

The rationale behind the implementation of the trading logic is that, if the price-to-MA ratio is significantly large, then it means that the current price of the underlying asset is significantly deviating from its average value. Thus, it is viable to short sell the asset to make a bet that the asset would reverts to its original, average position.

Moreover, this strategy has also considered the effect of extreme price movements. In some occasions, the price of the asset would deviate from the average position for a lingering period. Under a scenario where the price is significantly larger than the average position at a 1% level of significance, the algorithm would recurringly short sell the underlying position while the price movement exhibits an upward trend, causing losses to the strategy. Thus, the stop loss mechanism is implemented to account for these extreme price movements.

4.4.2 Back-testing Results and Evaluation

Table 11 summarizes the best-performing back-testing results for the reversion strategy, using the SPDR S&P 500 Index ETF and Bitcoin.

Instrument	Sharpe Ratio	Annual Return	Drawdown	Win Rate	Alpha
SPY	0.713	14.7%	33.7%	83%	0.028
BTC	-0.123	-3.9%	39.5%	13%	0.004

Table 11: Back-testing Results of the Reversion Strategy

Based on the results above, the reversion strategy works generally well in the stock market but poor in the cryptocurrency market. This is due to the difference in price stability in the stock and cryptocurrency market. In the cryptocurrency market, the price movements of Bitcoin and other cryptocurrencies are extreme and volatile. Consequently, a continuous rise or drop in the underlying cryptocurrency price would lead to false signals, where the algorithm is triggered to short sell while the price is rising, limiting the algorithm to profit from the upside. Even with the stop loss implementation, the return for Bitcoin is still negative. This can be attributed to the problem that the algorithm has been triggered to stop loss multiple times, leading to the negative annualized return.

Regarding the applicability of the reversion strategy in the stock market, this algorithm works well in an aggregated index/ETF but not in individual stocks. The reason is that ETFs such as the SPY is a weighted portfolio comprising of the largest individual stocks based on market capitalization. Compared to individual stocks such as Tesla and Amazon, SPY has a lower volatility because it effectively eliminates the idiosyncratic and firm-specific risk that is present in individual stocks. Due to the low volatility of SPY, the percentiles calculated when fitting the price-to-MA ratio distribution is less extreme. This can give more accurate signals to take positions in the underlying with the implemented stop loss mechanism.

5. Project Schedule

The project schedule is shown in **Table 12**, outlining the progress to be achieved in stages.

This project is on schedule and is currently working on the enhancement of various trading strategies. After the strategies are enhanced, the best-performing ones will be combined in February. Live trading has been initiated in January 2024, which is ahead of schedule. The algorithm under live trading and its profits will be monitored. In March, the back-testing results for the new strategies will be evaluated and forward testing will be conducted. Finally, in April, the final report and presentation will be completed before the project exhibition.

Time	**Objectives**	**Deliverables**
Jun – Aug 2023	Initial Research, literature review and feasibility study	- Feasibility report
Sep 2023	Preliminary back-testing on existing strategies; Consult project supervisor	- Project proposal for supervisor
Late Sep 2023	Initial Proposal and Project Website	- Detailed project plan - Project web page
Sep – Oct 2023	Code and back-test for existing strategies, machine learning strategies	- Back-testing results of existing strategies
Nov – Dec 2023	Research, develop and back-test for new strategies and arbitrage	- Back-testing results of new strategies
Mid Jan 2024	Write interim report and prepare for first presentation	- Preliminary implementation - Detailed interim report - First presentation slides
Jan – Feb 2024	Combine strategies to create a feasible one	- Back-testing results of combined strategies
Feb – Mar 2024	Simulate strategies using forward testing and live trading	- Forward testing results of combined strategies
Feb – Apr 2024	Display results of different strategies	- Final presentation slides
Late Apr 2024	Final report and deliverables due	- Final Report - Finalized tested implementation

Table 12: Project Schedule

6. Conclusion

This interim report outlines the work done by the project team since project inception. The project aims to provide a comprehensive overview of the return profile of existing trading strategies after taking transaction costs and brokerage fees into account, and to develop a proprietary profitable trading strategy that generates excess returns for retail investors.

Research on common algorithmic trading strategies had been discussed and a comprehensive evaluation for each trading strategy had been presented in this report. The back-testing result of the moving average strategy suggest that this strategy has improved performance after adopting a dynamic moving average. A higher Sharpe ratio and Alpha had been achieved. However, the project team will take into consideration the increased drawdown in future versions of the moving average strategy. Machine learning strategies, the cryptocurrency metadata strategy and the reversion strategy yielded lower returns compared to the moving average strategy. These strategies will serve as a foundation for future algorithm development and possible integration of multiple strategies in the pursuit of higher Sharpe ratio and excess returns.

The immediate next steps will include developing a new version of the moving average strategy, initiating forward testing for back-tested strategies, monitoring live trading and developing a proprietary trading algorithm by combining multiple strategies. These steps will be completed by March 2024 and will be presented during the final presentation in mid-April 2024. Most importantly, these result will be included in the final report for further discussion.

An important avenue for future work is to enhance the back-testing engine. A more comprehensive back-testing engine should be developed for a better strategy evaluation to achieve better back-testing and optimization efficiency. Back-testing engine that is capable of simulating the market impact of each trade would allow the project team to achieve more accurate and reliable results.

References

Alexandru, M. (2016). Algorithmic and high-frequency trading strategies: A literature review. https://www.econstor.eu/bitstream/10419/144690/1/860290824.pdf

Cboe. (2018). *Real-Time Latency Monitoring.* https://cdn.cboe.com/resources/features/Cboe_USE_RTLM.pdf

Chen, B., Gan, Q., & Vasquez, A. (2023). Anticipating jumps: Decomposition of straddle price. *Journal of Banking & Finance, 149,* 106755. https://doi.org/10.1016/j.jbankfin.2022.106755

Chu, J., Chan, S., & Zhang, Y. (2020). High frequency momentum trading with cryptocurrencies. *Research in International Business and Finance, 52,* 101176. https://doi.org/10.1016/j.ribaf.2019.101176

Cuthbertson, K., Nitzsche, D., & O'Sullivan, N. (2010). Mutual fund performance: Measurement and Evidence. *Financial Markets, Institutions & Instruments, 19(2),* 95187. https://doi.org/10.1111/j.1468-0416.2010.00156.x

Feng, Y., Rubio, F., & Palomar, D. P. (2012). Optimal order execution for algorithmic trading: A CVaR approach. *2012 IEEE 13th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC).* https://doi.org/10.1109/spawc.2012.6292954

Fraenkle, J., & Rachev, S. T. (2009). Review: algorithmic trading. *Investment Management and Financial Innovations, 6(1),* 720. https://www.researchgate.net/publication/283166465_Review_Algorithmic_trading

Gatheral, J., & Schied, A. (2013). Dynamical models of market impact and algorithms for order execution. *Handbook on Systemic Risk,* 579602. https://doi.org/10.1017/cbo9781139151184.030

Hendershott, T., & Riordan, R. (2012). Algorithmic trading and the market for liquidity. *SSRN Electronic Journal.* https://doi.org/10.2139/ssrn.2001912

Huang, J., & Huang, Z. (2020). Testing moving average trading strategies on ETFs. *Journal of Empirical Finance, 57,* 16-32. https://doi.org/10.1016/j.jempfin.2019.10.002

Huang, P., & Wang, T. (2016). On the Profitability of Optimal Mean Reversion Trading Strategies. *SSRN Electronic Journal.* https://doi.org/10.2139/ssrn.2719182

Kilgallen, T. (2012). Testing the simple moving average across commodities, global stock indices, and currencies. *The Journal of Wealth Management*, *15*(1), 82-100. https://doi.org/10.3905/jwm.2012.15.1.082

Lv, D., Yuan, S., Li, M., & Xiang, Y. (2019). An empirical study of machine learning algorithms for stock daily trading strategy. *Mathematical Problems in Engineering, 2019,* 130. https://doi.org/10.1155/2019/7816154

Menkveld, A. J., & Zoican, M. A. (2017). Need for speed? Exchange latency and liquidity. *The Review of Financial Studies, 30(4),* 1188-1228. https://doi.org/10.1093/rfs/hhx006

Nimalendran, M., & Ray, R. (2011). Informed Trading in Dark Pools. *SSRN Electronic Journal.* https://econfin.massey.ac.nz/school/documents/seminarseries/manawatu/Informed_Trading_in_Dark_Pools_nimal&Sugata.pdf

Nuti, G., Mirghaemi, M., Treleaven, P., & Yingsaeree, C. (2011). Algorithmic trading. *Computer*, *44*(11), 61-69. https://doi.org/10.1109/mc.2011.31

Ozbayoglu, A. M., Gudelek, M. U., & Sezer, O. B. (2020). Deep learning for financial applications : A survey. *Applied Soft Computing*, *93,* 106384. https://doi.org/10.1016/j.asoc.2020.106384

Shen, S., Jiang, H., & Zhang, T. (n.d.). Stock Market Forecasting Using Machine Learning Algorithms. https://cs229.stanford.edu/proj2012/ShenJiangZhang-StockMarketForecastingusingMachineLearningAlgorithms.pdf

Théate, T., & Ernst, D. (2021). An application of deep reinforcement learning to algorithmic trading. *Expert Systems with Applications, 173,* 114632. https://doi.org/10.1016/j.eswa.2021.114632