G01

Seeking Alpha Amidst

Machine Learning-based Quantitative Trading Strategies

By

LEUNG Pak Hei, Marco

LEUNG Cheuk Hei, Victor

KONG Siu Hei, Ricky

FUNG Kam Kwan, Thomas

Advised By

Mordecai J. GOLIN

Submitted in fulfillment

of the requirements for COMP 4981

in the

Department of Computer Science and Engineering

The Hong Kong University of Science and Technology

(2022-2023)

Date of submission: April 20, 2023

Table of Contents

Abstract

In the rapidly changing landscape, Algorithmic Trading minimizes human variance. In fact, it is exceptionally difficult for a retail trader to create an algorithm trading model with relatively fewer funds on a small scale. Thus, this report aims to present a profitable algorithmic trading strategy in the face of maximizing financial gains.

Our project is separated into two major sections - models and software implementation. Empowered by statistics and mathematical models, AI leverages small datasets and generates insightful knowledge. Hereby, we believe the machine learning model would be a useful tool in the creation of a successful trading strategy.

In this report, the superiority of machine learning among traditional models in trading is further discussed. Honed with effort, we will introduce a simple trading platform for automated trading. Several strategies are refined and integrated with our corresponding machine-learning models. A series of comparisons will be conducted between machine learning and traditional models. Only models with enlightening results will be embedded into our trading platform. After all, the experiment result showed most ML models outperform non-machine learning models.

Acknowledgment

We would like to express our gratitude toward our supervisor, Professor Golin. All of his guidance inspired us with ideas and passion for the development of machine learning models and trading systems. Without this help throughout the model design and system implementation, we would not have completed our project with speed and precision.

Our communication tutor Ms. Noorliza Daveau also helped us with conceptualizing our system, concluding our work, and structuring the reports. Thus, we would also like to appreciate all of her help on this FYP.

Introduction

Overview

Algorithmic Trading has become an indispensable and essential part of the trading market, where over 70% of trading since 2019 was algorithmic trading [1]. Algorithmic Trading refers to traders using algorithms to make trade decisions, which may even place orders with it. It has numerous advantages for traders, ranging from accuracy, efficiency, profit, and many more pros. It has undeniably gained traction among institutional traders and retail traders. This knowledge of quantitative trading has become known as Computational Finance.

In the era of Artificial Intelligence(AI), Machine Learning (ML) Algorithmic Trading has risen in the 1990s [2]. This field has accumulated over 350 papers since 2019 and the popularity of ML-based Trading is undeniably receiving more attention in recent years due to its potential of exploring patterns that are unable to be discovered by humans [2].

However, most journal publications and researches in this field suffer from two limitations, large fund base or small liquidity backtest. ML Algorithm Trading is more commonly adopted by large firms with an enormous amount of cash flow that could be used to create these models. Thus, a lot of research is based on having this large amount of funds available. On the other hand, journal publications that focused on ML Algorithm Trading for individual traders mostly backtested in a small liquidity market, or even didn't implement and test it. Thus, most of them do not work for individual traders.

The goal of this project is to present profitable ML-based trading strategies that are superior to the return of the corresponding comparison asset that we chose (e.g., S&P 500 for U.S. Stocks or futures). This ML-based model will be based on Clustering-based models, LSTM, and Deep-Q Learning. We will also evaluate the effectiveness of ML-based compared to non-ML-based trading approaches and compare their performance.

Objectives

The objective of this project is to design and develop machine learning-based trading strategies for various financial markets. We aim to first start with non-machine learning techniques, including statistical arbitrage trading and defensive portfolio strategy. Then, we will do machine learning-based strategies, such as multiple classifier trading, clustering-based pairs trading, reinforcement learning trading, and RNN-based portfolio optimization. We will backtest each of these strategies using historical data to evaluate their performance and determine their potential for use in real-world trading scenarios. After the backtesting, a trading system is developed for the real-time trading environment.

Specifically, we aim to achieve the following objectives in non-machine learning research, machine learning research, and software development:

A. Non-machine learning models

1. Investigate and evaluate the effectiveness of statistical arbitrage pair trading strategies, which seek to profit from mispricings in the market using quantitative models.
2. Develop defensive portfolio optimization strategies which seek to optimize the portfolio using mathematical models.

B. Machine Learning models

1. The Multiple Model-based Strategies iterate through most sci-kit-learn models, with the comparison of classifiers and regressors predictions. With more granular feedback, this model is useful for badly defined classes with unclear decision boundaries.
2. Investigate the effectiveness of clustering-based pairs trading strategies, including K-means Clustering, Hierarchical Clustering, and Affinity Propagation, which group similar stocks together and identify trading opportunities based on the relationship between them.
3. Develop and test reinforcement learning-based trading strategies, including Markov Decision Process, and deep Q Learning network to optimize trading decisions based on market conditions and investor preferences.

4. Develop and test RNN-based portfolio optimization strategies, which leverage temporal dependencies in financial data to optimize portfolio allocations and maximize returns.

C. Software Development

1. Build a Trading Server as an order management system to handle orders placed by different models
2. Develop a Restful API Server to handle data creation and requests from the user interface
3. Develop a user interface for simple user control of adding models and monitoring model performance

Through these objectives, we aim to design and develop a comprehensive trading system that incorporates multiple machine-learning techniques and can effectively generate trading signals for a range of financial markets.

Literature Survey

Our motivation to work on machine learning-based trading strategies stems from the fact that traditional trading strategies often rely on human expertise and subjective decision-making, which can be biased and error-prone. Machine learning, on the other hand, has the potential to overcome these limitations by providing objective and data-driven decision-making processes.

Statistical arbitrage trading

It has gained significant attention in recent years due to its ability to profit from market inefficiencies. This approach involves identifying pairs of securities with similar price movements and exploiting the temporary price discrepancies between them. Traditional statistical arbitrage strategies rely on statistical methods such as cointegration and correlation analysis to identify such pairs[1]. However, machine learning-based approaches can improve upon these methods by identifying more complex relationships between securities and incorporating a larger set of features for analysis[3].

Defensive Portfolio Allocation

Portfolio Allocation is a well-known phenomenon that the economy that behaves in cycles. During economic expansions, stocks outperform other assets e.g. gold and bond. However, economic conditions are uncertain. e.g during stagflation, other defensive assets e.g. gold might be preferred. It allows investors to reallocate capital to less risky assets during different periods of economic cycles and manage the risk of investment accordingly based on overall economic health. We would like to include this non-ML model, which is not included in our initial report since it is one of the most common strategies used by investors and we would like to provide a more diverse comparison to ML models.

Multiple Model-based Binary Classification

Multiple classifier trading involves combining the predictions of multiple classifiers to improve trading performance. In fact, converting a regression model to a classification model through output discretization can also be beneficial for improving the model's performance. This approach has gained attention in recent years, due to its ability to reduce overfitting and improve robustness. It involves

training multiple regressors/classifiers on different subsets of the data with different algorithms. This approach can also be applied to feature selection, where those algorithms are useful to identify the most relevant features for trading.

Clustering-based Pairs Trading

Traditional pair trading is a popular quantitative trading strategy that involves identifying a pair of highly correlated assets and taking long and short positions in them simultaneously. The aim of this strategy is to profit from the divergence of the two assets by taking advantage of the correlation between them[]. This method requires a significant amount of historical data to identify the correlation between the assets, and it relies on the assumption that the correlation between the assets will remain constant.

Clustering-based pair trading, on the other hand, is a more recent development that uses machine learning techniques to identify groups or clusters of assets that exhibit similar behavior. This method aims to exploit the correlation within clusters rather than between individual assets. Clustering algorithms, such as k-means and hierarchical clustering, are used to group assets based on their historical performance, volatility, and other characteristics[7][8]. The motivation behind clustering-based pair trading is to improve the accuracy of identifying and exploiting market inefficiencies, which can result in higher profits.

RNN-based Portfolio Optimization

Traditional portfolio optimization involves constructing a portfolio that maximizes returns while minimizing risk, based on expected returns, volatilities, and correlations of assets. However, traditional models may fail to capture the complex dynamics of financial markets and may not be robust to sudden changes in market conditions.

To overcome these limitations, researchers have explored the use of recurrent neural networks (RNNs) for portfolio optimization. RNNs can model the temporal dependencies and non-linear relationships in time-series data, making them well-suited for predicting financial market trends.

RNN-based portfolio optimization involves training an RNN on historical asset prices to predict future prices and compute the optimal allocation of funds to each asset. The approach improves the accuracy and robustness of portfolio construction

by leveraging the power of deep learning algorithms.

Recent studies have shown promising results with RNN-based portfolio optimization, and the approach has the potential to provide significant benefits[9][10]. However, further research is needed to explore its limitations and potential applications in different market conditions.

Momentum trading

Traditional momentum trading is a strategy that involves buying assets that have exhibited positive returns in the recent past and selling assets that have exhibited negative returns. The motivation behind this strategy is based on the belief that assets that have performed well in the past are likely to continue performing well in the future, while assets that have performed poorly are likely to continue performing poorly.

Reinforcement learning trading is a relatively new approach that uses machine learning algorithms to optimize trading decisions. The motivation behind reinforcement learning trading is to create an adaptive trading system that can learn and adjust to changing market conditions, without the need for explicit rules or assumptions.

Reinforcement learning trading involves training a model on historical market data to make decisions based on reward signals received from the market. The approach can potentially provide a more robust and adaptive trading strategy than traditional momentum trading, which relies on simple rules and assumptions[9].

Motivation

Our interest in these topics stems from the potential to develop more accurate and efficient trading strategies by leveraging the power of machine learning. We believe that by combining the strengths of various machine learning techniques, we can develop a more robust and effective trading system that can adapt to changing market conditions and achieve superior performance. We will further test multiple ML Models in past papers and evaluate them as we create different models. They will then be implemented and linked to a real-time trading system.

Methodology

Non-ML Trading Strategies

We select statistic arbitrage pair trading and defensive portfolio allocation because these two strategies are common strategies used by institutional investors. Although defensive portfolio allocation is not included in our initial proposal, we would like to test its performance and compare it with machine learning model-based strategy e.g. RNN Portfolio Management Strategy.

1. Statistic Arbitrage Pair Trading

1.1 Dataset

Ticker data [Open, High, Low, Close, Volume]
Stocks under S&P500 in day intervals from Yfinance API between 2013-2023

1.2 Logic

We use the classic approach of pair trading consisting of two steps, identification of a pair of securities, for which the corresponding price series display similar behavior. This indicated that both securities are exposed to related risk factors and tend to react identically.
Algorithm for Pair Selection:

1. Select the universe of assets from SPY500
2. Apply and validate the cointegration relationship using the cointegration test.
3. If cointegrated then eligible for pair trading.

After pair selection, we would apply a pair trading strategy to select pairs and evaluate the performances and past price data.

Reason for Cointegration Pair Selection

Cointegration is a statistical technique that measures the long-term equilibrium relationship between two or more time series. The cointegration approach is often used in pairs trading to identify pairs of assets that are likely to maintain a stable long-term relationship.

Pairs of assets that are cointegrated tend to exhibit mean reversion, which is the tendency for the assets to move back toward their long-term equilibrium relationship after short-term deviations. This means that if one asset in the pair experiences a significant price movement, there is a good chance that the other asset will also move in the opposite direction, creating opportunities for profitable trades.

By selecting pairs that are cointegrated, traders can reduce the risk of taking positions in assets that are not truly related. This can help to minimize losses due to unexpected market movements or changes in market conditions.

The cointegration approach provides a more accurate measure of the relationship between two assets than simple correlation analysis. This is because cointegration takes into account the long-term dynamics of the relationship between the two assets, rather than just looking at short-term fluctuations.

1.3 Model's Context

Pair trading is a market-neutral trading strategy that involves identifying pairs of assets that are related and trading them based on their relative price movements.
We first filter the stocks from SPY500 based on the cointegration method:

if cointegration between stock1 prices and stock2 prices < 0.02:
we select stock1 and stock2 as potential pairs.

After selecting a pool of potential pairs, we try to exploit the mean reversion property of the price ratio of 2 stocks:

<u>Algorithms for Pair Trading</u>

t_short = short look back period
t_long = long look back period
Price ratio = price of stock1 /price of stock2
z-score = (t_short moving average of price ratio- t_long moving average of price ratio)/ t_long std of price ratio
if z-score> upper_threshold:
buy stock2 with 2 % of current cash and short stock1 with the same value.
if z-score<- upper_threshold:
buy stock1 with 2 % of current cash and short stock2 with the same

value.

else if -medium_threshold <z-score<medium_threshold

close all position

The parameters [*t_short, t_long, upper_threshold, medium_threshold*] of the above strategy is acquired through optimization of strategy.

1.4 Optimization for Pair Trading

After selecting asset pairs, the entry and exit conditions of the strategy can be decided. We fine-tuned the model. The goal was to find the parameters such that the profit was maximized for selected pairs.

After trials and errors, we discover that t_short = 4, t_long = 140, upper_threshold = 1, and medium_threshold = 0.5 gives the most satisfying result in terms of portfolio returns and risk-adjusted return.

1.5 Evaluation

We backtested the pair trading strategy based on more updated data from 2013-2023.

Here is the list of backtest results obtained:

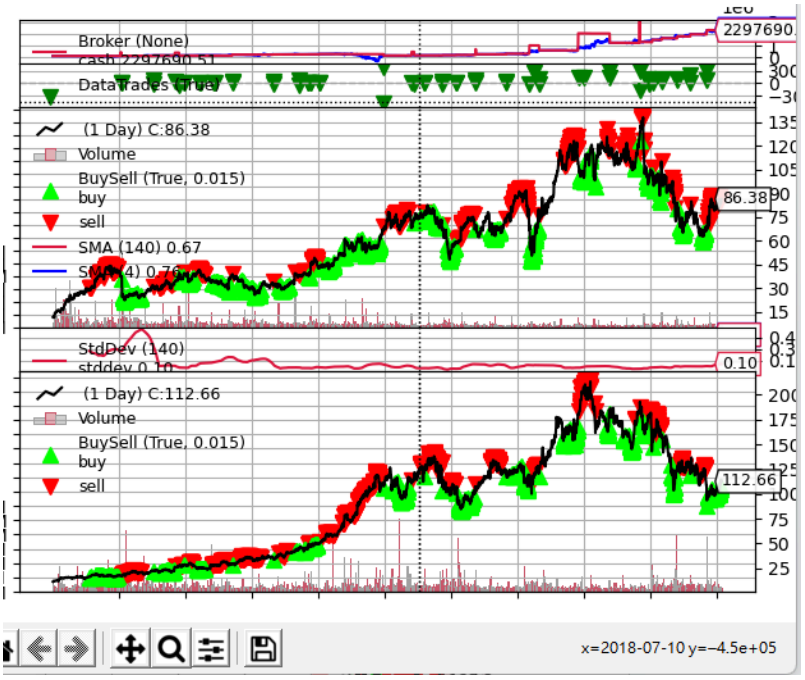


Figure 1: Backtest report for (BBY, TTWO) 2013-2021

(BBY, TTWO) Cointegration Value: 6.372*e-05

Sharpe Ratio: 0.790359

Figure 2:Backtest report for (ARE, ADI) 2013-2021

(ARE, ADI) Cointegration Value: 0.0015890

Sharpe Ratio: 1.125

Figure 3: Backtest report for (ADBE, RMD) from 2013-2021

(ADBE, RMD) Cointegration Value: 0.000275646489533357
Sharpe Ratio: 0.83182

1.6 Conclusion

We backtested pair trading strategies with a list of selected pairs. It provides significant returns within the time frame of 2013-2023. Based on backtesting, statistical arbitrage has shown the potential to provide significant returns for investors.

The results of backtesting have indicated that statistical arbitrage can be an effective strategy for generating alpha, or excess returns, over a given benchmark. By using quantitative models to identify mispricings in the market, statistical arbitrage traders can capture profits from small price movements that are not easily detectable by human traders.

2. Defensive Portfolio Allocation(Asset Rotation Strategies)

2.1 Dataset

2.1.1 Signal Generation

1. 10-Year Treasury Constant Maturity minus 2-Year Treasury Constant Maturity
2. U.S. Treasury Securities at 20-Year Constant Maturity
3. 10-Year Breakeven Inflation Rate

Data of the above index is downloaded from the federal government website[30][31][32].

2.1.2 Trading Assets

1. SPDR S&P 500 ETF
2. SPDR Gold Shares
3. iShares 20+ Year Treasury Bond ETF

Data of trading assets are downloaded using Yfinance API.

2.2 Logic

The defensive portfolio allocation strategy allows rotation to low-risk assets during times of economic uncertainty based on macroeconomic indicators e.g. 10-Year Breakeven Inflation Rate.

We develop the rule-based defensive rotation strategy based on federal data of treasury 10-Year Treasury Constant Maturity, 2-Year Treasury Constant Maturity, and U.S. Treasury Securities at 20-Year Constant Maturity and 10-Year Breakeven Inflation Rate. We want to develop this strategy since the reallocation of capital in the strategy is based on economic cycles [28], which is a well-known macroeconomic phenomenon. More importantly, it allows comparison with other machines learning-based portfolio management strategies e.g RNN portfolio optimization.

There are several macroeconomic indicators that can be used to construct the strategy.

1. High yield curve signal stronger economic growth, we bet on US stocks. In order to, obtain the slope of the yield curve, we use

$$Yield\ Curve\ Signal\ = \ 10\ Year\ Treasury\ Constant\ Maturity - \ 2\ Year\ Treasury\ Constant\ Maturity$$

   The yield curve signal data can be downloaded from federal government website[30].
2. Break-even inflation rate represents a measure of expected inflation[31].

$$10\ Year\ Breakeven\ Inflation\ Rate\ = \ 10\ Year\ Treasury\ Constant\ Maturity\ Securities - \ 10\ Year\ Treasury\ Inflation\ Indexed\ Constant\ Maturity\ Securities$$

$$Stagflation\ signal\ = \ 10\ Year\ Breakeven\ Inflation\ Rate - \ Yield\ Curve\ Signal$$

   Stagflation is an economic cycle characterized by slow growth and a high unemployment rate accompanied by inflation[29]. Gold is preferred since investor tends to invest in gold when the economic condition is uncertain.
   The 10-Year Breakeven Inflation rate data can be downloaded from the federal government website[31].
3. Lastly, we use 20-year bond yield as signals for buying ETFs for 20-year bonds.
   The 20-year bond yield data can be downloaded from the federal government website[32].

2.3 Model's Context

2.3.1 Visualize Trading Signals

```
[79] fig,ax = plt.subplots(2,figsize=(20,10))
     ax[0].plot(df.index, df["DFII20"].values,color="blue")
     ax[0].plot(df.index, df["yieldcurve"].values,color="red")
     ax[0].plot(df.index, df["Inflation_yield"].values,color="green")
     ax[1].plot(df.index, df["Trade Signal"].values,color="orange")
     plt.show()
```

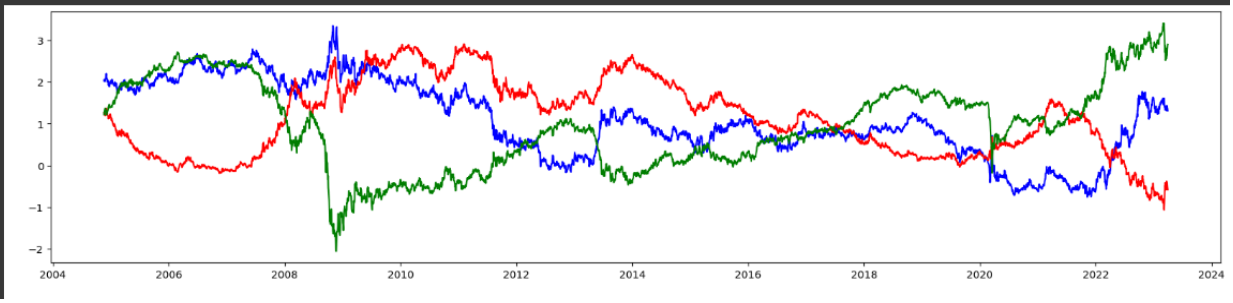


Figure 4: Graph for trading signals. The red line indicates the Yield Curve Signal. The green line indicates Stagflation Signal. The blue indicates a 20-year maturity bond yield.

2.3.2 Construct Defensive Strategy

Algorithm for Defensive Portfolio Allocation

1. If the Yield Curve Signal is the highest compared to the other 2 signals, indicates a strong economy, hence invest all capital in stock
2. If Stagflation Signal is the highest compared to the other 2 signals, indicate stagflation, and invest all capital in gold.
3. If the 20-year bond yield is the highest compared to the other 2 signals, invest all capital in a 20-year bond.

We reallocate capital with the selected assets based on the 3 signals in Figure 4. As shown in Figure 5, during an economic downturn low-risk assets e.g. bond, gold is selected such that it allows protection of portfolio value during periods of high economic volatility. The blue sections of Figure 5, indicate a period of investment in corresponding assets in the graphs.

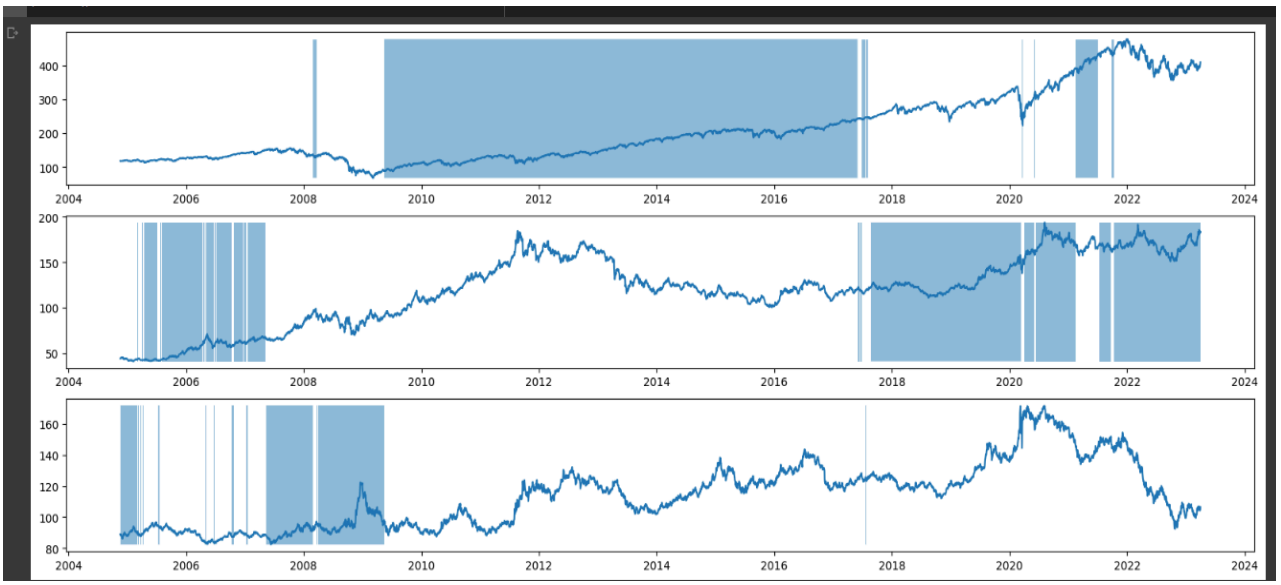


Figure 5: The upper graph is SPDR S&P 500 ETF Trust from Yahoo Finance. The middle graph is SPDR Gold Shares. The bottom graph is iShares 20+ Year Treasury Bond ETF.

2.4 Evaluation

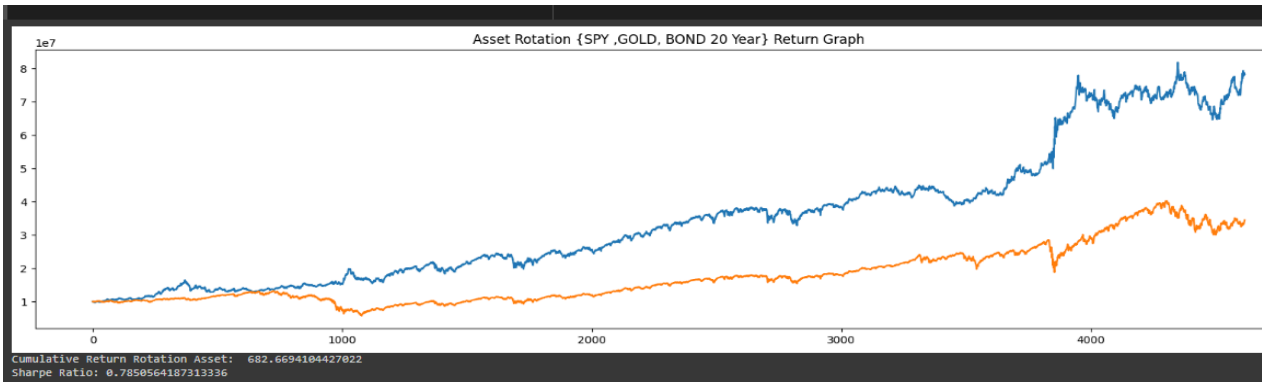


Figure 6: Orange curve is the return for reference strategy for using buy-holding strategies for SPDR S&P 500 ETF. The blue curve shows the return of the asset rotation strategy.

The graph above included the defensive asset rotation strategy and buy-hold strategy for SPDR S&P 500 ETF as a reference. Both of the strategies start with $10000000 capital, it is shown that defensive asset rotation strategies demonstrate significant returns compared to the reference strategy.

The cumulative return of the strategy is 682.669%.

Sharpe Ratio: 0.78595

2.5 Conclusion

The defensive portfolio allocation strategy allows the avoidance of major drawdown during the economic crisis and provides a significant return compare to the buy and hold strategy of SPY500 ETF.

ML Trading Strategies

1. RNN with Markowitz portfolio optimization

1.1 Dataset

Ticker data [AdjClose] of stocks under S&P500 in day intervals from Yfinance API between Jan 2021- Mar 2023.

1.2 Logic

The traditional Markowitz portfolio optimization theory, developed by Markowitz in 1952, posits that investors should seek to invest in stocks with the lowest standard deviation (risk) and highest return (profit) based on rationality. This model relies on historical return and standard deviation to construct an optimal portfolio. However, we propose a novel modification to this traditional approach by incorporating an RNN model to generate "predicted stock prices.

1.2.1 Reason for RNN Portfolio Optimization

RNNs offer a unique advantage in that they are capable of capturing temporal dependencies within the data, enabling the model to learn and adapt to changing market conditions. By leveraging RNNs, we can construct more accurate predictions of future stock prices and returns, allowing for the creation of optimized portfolios that are better suited to current market conditions. Furthermore, RNNs offer a more flexible and adaptable approach to portfolio optimization as they can incorporate a variety of data sources such as news sentiment, macroeconomic indicators, and technical analysis. This provides investors with a more comprehensive and dynamic view of the market, enabling them to make better-informed investment decisions.

This allows for the construction of a Markowitz portfolio with greater accuracy and a higher potential for profitability (Appendix C). Additionally, to mitigate the risk of overfitting or sentiment-driven volatility, multiple random sampling techniques are employed to select the best global portfolio. This approach offers a more robust and reliable framework for optimizing investment portfolios.

1.3. The Model's Context

1.3.1 Choose appropriate RNN models

Some of the RNN models that can be compared for stock prediction include simple RNN, LSTM, GRU, Bidirectional LSTM, and Lasso-LSTM. The models can be implemented using Keras. For a fair comparison, the fully connected layers are the same for each model. The performance of the RNN-based model will be primarily based on Mean-Squared Error (MSE). The goal is to get a rough performance of what RNN models fit the best in the stock market. Only a promising RNN model will then be used in portfolio optimization.

1.3.2 Construct Markowitz Portfolio optimization using promising RNN model

To apply these optimization techniques using LSTM models, we can use the predicted future returns from the model as inputs to the optimization problem. The weights of the assets in the portfolio can then be calculated using the optimization algorithm, which will give us the optimal portfolio that maximizes returns and minimizes risk.

Once we have the optimal portfolio, we can backtest its performance using historical data and evaluate its performance metrics such as the Sharpe ratio, return on investment, and maximum drawdown. By optimizing the portfolio using LSTM models and advanced optimization techniques, we may be able to outperform traditional methods of portfolio optimization and achieve better investment returns.

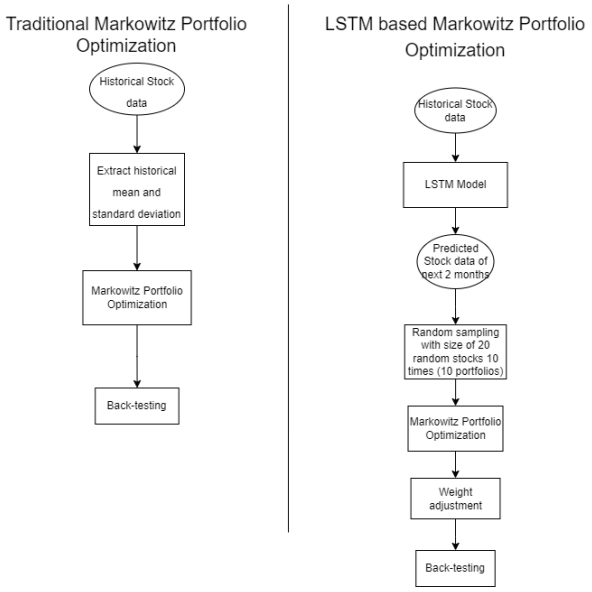


Figure 08: Logic of RNN based Markowitz Portfolio Optimization

1.4 Evaluation

1.4.1 Comparison of RNN models - LSTM

The performance of RNN models are shown as follows:

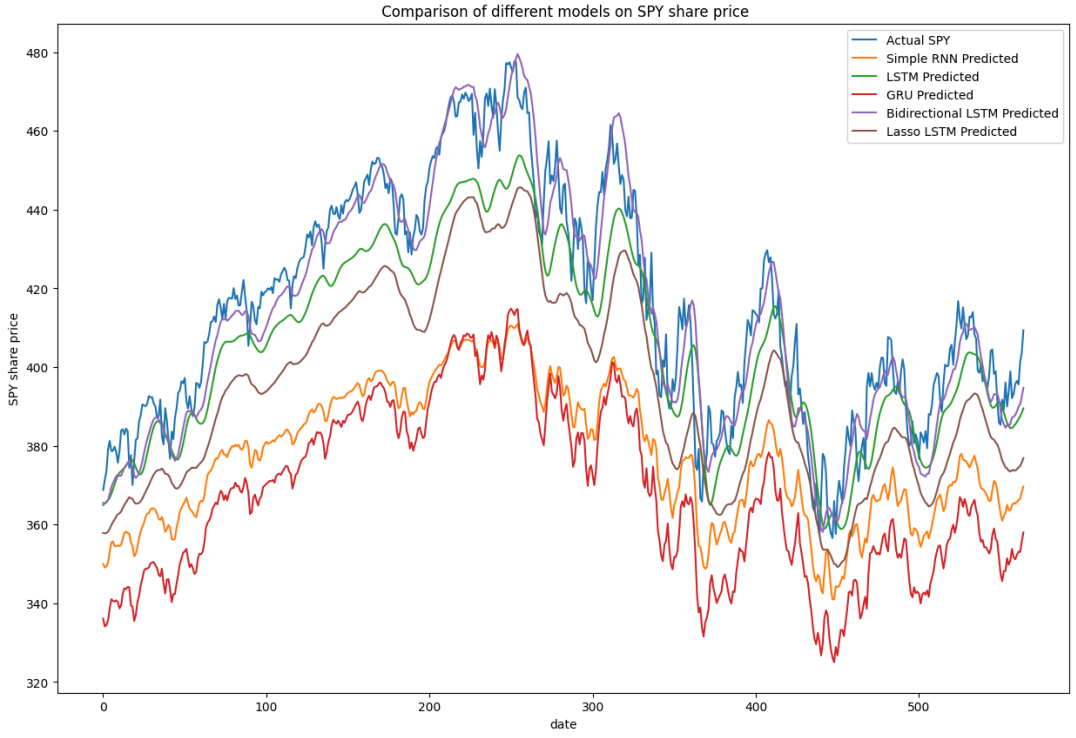


Figure 09: Logic of RNN based Markowitz Portfolio Optimization

RNN models comparison	
RNN models	MSE
Simple RNN	1607.730
GRU	2350.158
LSTM	118.734
Bidirectional-LSTM	68.373
Lasso-LSTM	519.108

Figure 10: Logic of RNN based Markowitz Portfolio Optimization

Among the models tested, the Bidirectional LSTM had the lowest MSE, indicating the best overall performance. The LSTM model had the second-lowest MSE, the third is Lasso-LSTM, followed by simple RNN and GRU. Despite the superior performance of the Bidirectional LSTM, we still have reasons for choosing the **LSTM** model due to the interpretability, training time, and complexity. LSTM proves to have predictive power and fit well in the stock market environment.

1.4.2 Evaluation of LSTM Portfolio Optimization(TBC)

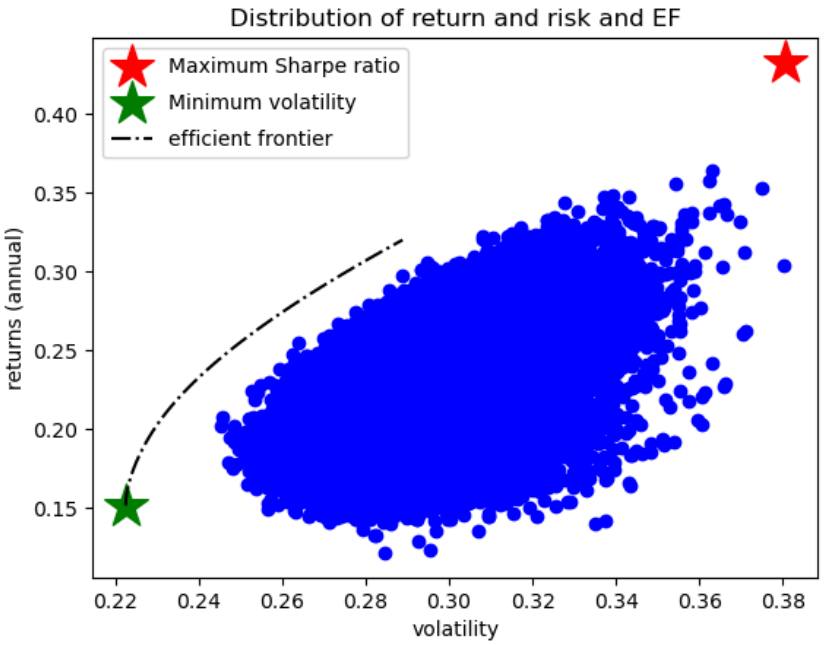


Figure 11: Example of one of the portfolios generated by Markowitz Portfolio Optimization

RNN-based Markowitz Portfolio Optimization Summary			
Model	Combination of tickers and weights	Annualized Returns	Annualized s.d
Sharpe ratio	[AAPL,AMZN,JNJ,NFLX,TSLA,TSM]=[14,17,14,17,14]	0.27	0.253
Sharpe ratio	[NET,NVDA,ORCL,OXY,TGT]=[26,13,14,7,39]	0.32	0.36
Sharpe ratio	[JNJ,KO,TSLA,TSM,XOM,ZM]=[6,31,41,11,8,2]	0.32	0.32
Sharpe ratio	[A,TSM,XOM,ZM]=[49,22,16,10]	0.19	0.25
Sharpe ratio	[A,NET,NVDA,OXY]=[43,32,16,8]	0.19	0.25
Min risk	[AMZN,JNJ,KO,NLFX,TSM,XOM]=[5,42,38,1,7,5]	0.12	0.15
Min risk	[JNJ, KO, TSM, XOM, ZM]=[43,35,5,7,7]	0.12	0.17
Min risk	[A,BCA,F,GOOGL,SPG,TSM,XOM,Z,ZM]=[31,3,2,15,2,8,30,8]	0.34	0.40
Min risk	[F,ICE,ORCL,SPG,TCOM,TGT]=[2,50,22,3,8,16]	0.14	0.22
Min risk	[A,BCA,F,GOOGL,SPG,TCOM]=[43,10,6,28,5,6]	0.15	0.25

table 1: Example of portfolios generated by Markowitz Portfolio Optimization

1.4.2 Evaluation of Trading Performance of LSTM portfolio optimization

The LSTM-based portfolio optimization was back-tested against two traditional portfolio optimization, including cap-weighted portfolios and an equally weighted portfolio. The results are as follows:

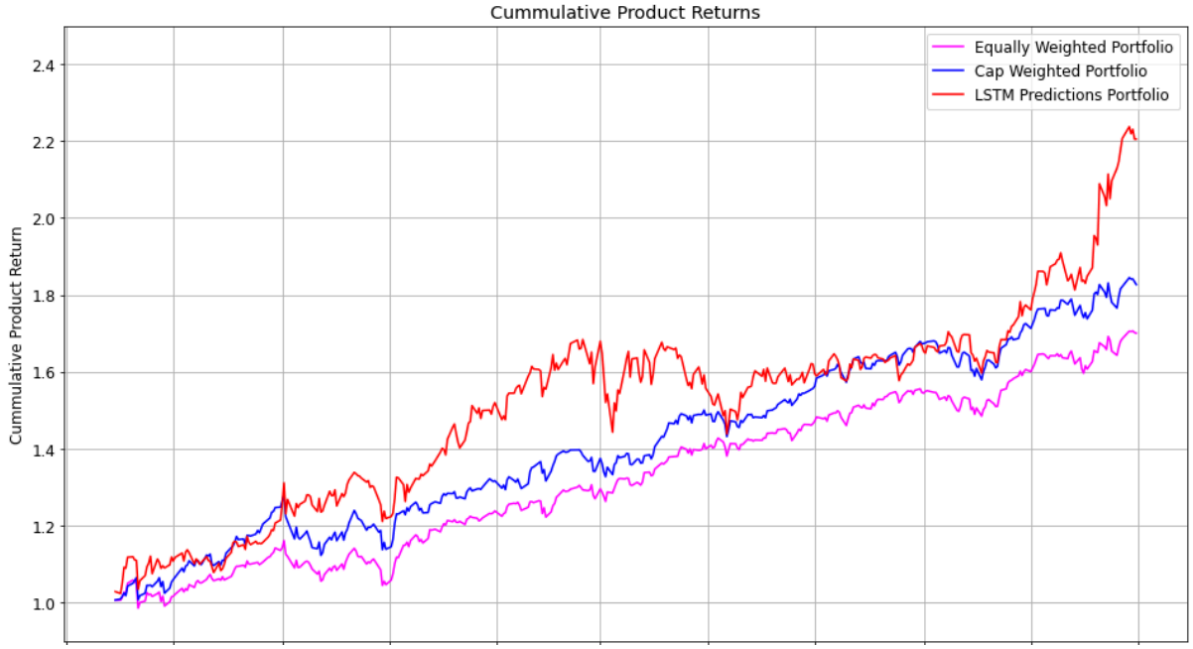


Figure 12: Comparison of different portfolio optimization

The LSTM-based Markowitz portfolio optimization has shown great capacity in constructing a good portfolio with a high Sharpe ratio of 1.325.

1.5 Conclusion

In conclusion, our experiment has shown that machine learning-based portfolio optimization techniques outperform traditional portfolio optimization methods. By leveraging the power of advanced algorithms and statistical models, machine

learning-based approaches can analyze complex data sets, identify patterns, and make predictions that are difficult or impossible for traditional methods. The results of our evaluation demonstrate that these techniques can lead to superior investment outcomes, with higher returns and lower risk. As such, machine learning-based portfolio optimization holds great promise for improving the performance of investment portfolios.

2. Clustering-based PairTrading (K-means, Hierarchical Clustering, and Affinity Propagation Clustering)

2.1 Dataset

Ticker data [AdjClose] of stocks under S&P500 in day intervals
Yfinance API between Jan 2021- Mar 2023.

2.2 Logic

The clustering will be implemented with Pairs Trading. The Model follows the distance-based approach to find the similarity of pairs, calculated by the Euclidean Distance of their return and stand deviation, and evaluate the convergence and divergence behavior of the trading pairs. Different clustering methods will be used: K-means, Affinity Propagation, and Hierarchical Clustering.

2.2.1 Reason for Clustering-based Pair Trading

Machine learning clustering-based pair trading can offer several advantages over traditional approaches. For example, clustering-based pair trading uses a data-driven approach to identify pairs of assets that are likely to be co-integrated. This approach can be more effective than traditional methods that rely on a priori assumptions about asset relationships. Clustering algorithms can also capture non-linear relationships between assets that may not be apparent with traditional methods. This can result in more accurate identification of pairs with strong co-integration. Clustering-based pair trading can even adapt to changes in market conditions and identify new pairs as they emerge. Traditional approaches may not be as effective in adapting to changing market conditions. Machine learning clustering-based pair trading can help to reduce human bias in the selection of pairs. Traditional methods may be subject to human biases such as familiarity with certain assets or past experiences.

2.2.2 Concepts of clustering models

<u>K-means clustering</u>

It is a popular clustering algorithm that partitions a dataset into k clusters by minimizing the sum of squared distances between the data points and their corresponding cluster centers. The number of clusters (k) is specified in advance and

can be determined by the Elbow Method or Silhouette Method.

Affinity Propagation

It is a clustering algorithm that does not require the number of clusters to be specified in advance. Instead, the algorithm uses a similarity matrix to find the exemplars, which are the most representative points of each cluster. The algorithm iteratively updates the responsibility and availability matrices until convergence.

Hierarchical Clustering

It is a clustering algorithm that builds a tree-like structure of clusters by merging or splitting them based on their similarity. The algorithm can be used for both agglomerative (bottom-up) and divisive (top-down) clustering. Agglomerative clustering is more commonly used in practice and starts by considering each data point as a separate cluster and then merges the most similar clusters until a stopping criterion is met.

The quality of the clustering results can be evaluated using two metrics, namely the K-value and silhouette score. On the other hand, the silhouette score is a measure of how well a data point fits into its assigned cluster. It ranges from -1 to 1, with values closer to 1 indicating a better fit. A high silhouette score suggests that the data point is well-clustered and that the clustering is accurate.

2.2.3 Formation of trading strategies

Once the clustering algorithm has been applied to the stock price data, the next step is to identify the pairs of stocks that have similar price patterns. This is done by calculating a similarity score between each pair of stocks within each cluster. This score can be based on various metrics such as correlation, cointegration, mean-reversion, or Hurst exponent (Appendix C).

Different criteria and indicators will also be used in both Models to find the few most promising pairs, such as Hurst Exponent and Cointegration test. After the pairs are found, pairs X/Y will be normalized by the historical mean and standard deviation (Appendix C).

Then the trading strategy is based on the following: assumed that trades are made only near to market close if the normalized value of today's close price of X/Y is higher/lower than 1.5 or -1.5, the pairs are shorted/longer accordingly.

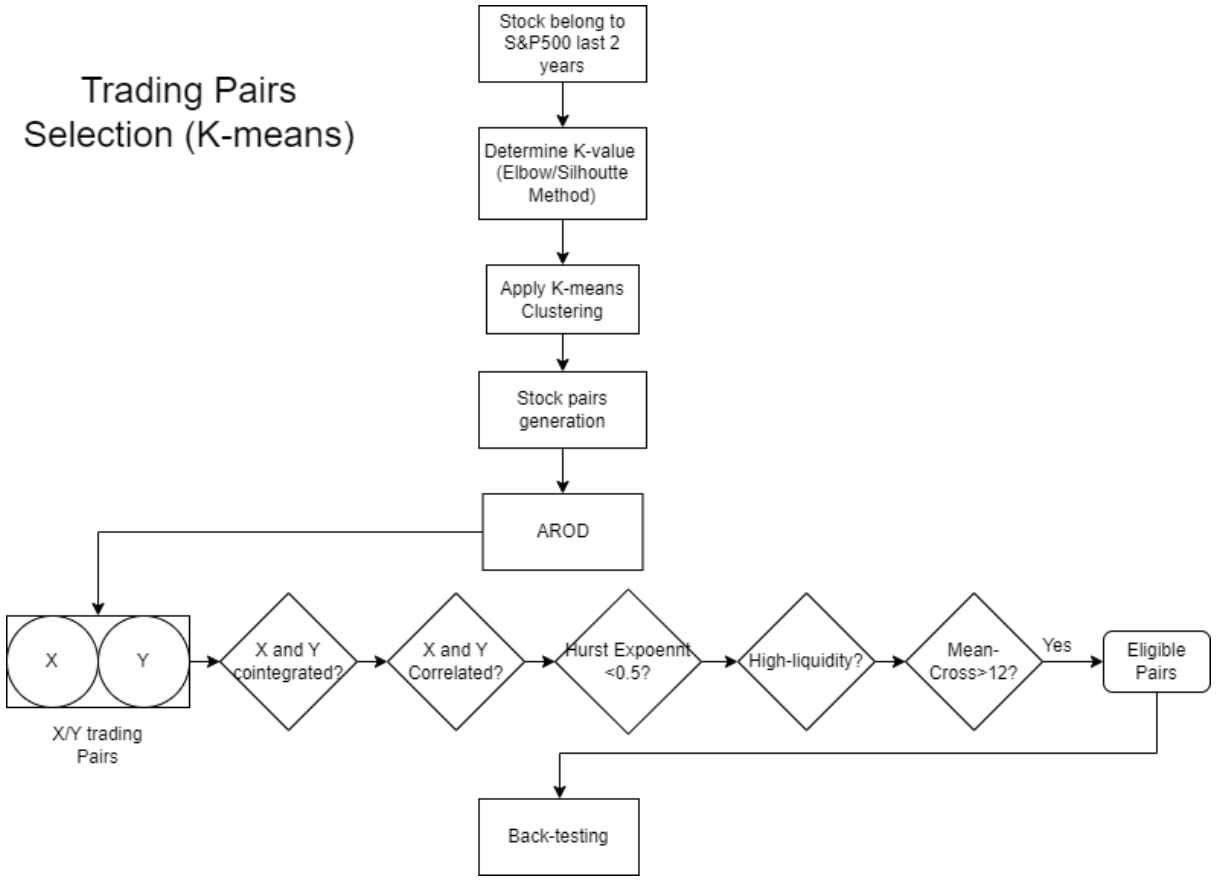


Figure 13: logic of Pairs Selection

2.3 Model's Context

Algorithm of the K-means clustering:

1. Initialize k cluster centroids randomly from the dataset.
2. Assign each observation to the nearest centroid.
3. Update the centroids by computing the mean of all observations assigned to each centroid.
4. Repeat steps 2 and 3 until the centroids no longer change or a maximum number of iterations is reached.

The objective function of K-means is defined as follows: $J = \sum_{i=1}^{k} \sum_{x \in C_i} ||x - \mu_i||^2$

where J is the objective functions, k is the number of clusters, C_i is the ith cluster, x is an observation in C_i, and μi is the centroid of C_i. The algorithm aims to minimize

this objective function by iteratively updating the centroids and re-assigning observations to the nearest centroid.

Algorithm of the Affinity Propagation Clustering:

1. Define the similarity matrix S, where S(i, j) represents the similarity between data points i and j.
2. Initialize the responsibility matrix R and availability matrix A to zero.
3. Update the responsibility matrix: $R(i,\ k)\ =\ S(i,\ k)\ -\ max(a\ +\ S(i,\ a), a)$
4. Update the matrix A: $R(a,\ k)))\ -\ max(0,\ R(i,\ k)\ + max(0,\ R(i,\ a))),\ a)$
5. Update the responsibility matrix R: $R(i,\ k)\ =\ (1\ -\ \lambda)\ R(i,\ k)\ +$ $\lambda\ (S(i,\ k)\ -\ max(a\ +\ A(i,\ a)),\ a)$
6. Update the availability matrix A as follows: $A(i,\ k)\ =\ (1\ -\ \lambda)\ A(i,\ k)\ +\ \lambda\ min(0,\ R(k,\ k)\ +\ sum(max(0,\ R(a,\ k)))$ $-\ max(0,\ R(i,\ k)\ +\ max(0,\ R(i,\ a)))$
7. Repeat steps 5-6 until convergence, where convergence is reached when the assignments stop changing.
8. Calculate the exemplars by finding the data points with the highest sum of R and A matrices: $e(i)\ =\ argmax(R(i,\ k)\ +\ A(i,\ k))$
9. Assign each data point to its corresponding exemplar

Algorithm of the hierarchical clustering:

1. Calculate the distance matrix D between all pairs of data points
2. Initialize each data point as a cluster
3. Find the closest pair of clusters C_i and C_j using a Single linkage: $min(D(i,j))$
4. Merge the closest pair of clusters into a new cluster
5. Update the distance matrix D to reflect the new cluster: D(k,l) = linkage(C_k, C_l)
6. Remove the old clusters C_i and C_j from the list of clusters
7. The final cluster is the root of the dendrogram, which can be cut at a desired height to obtain the desired number of clusters.

2.4 Evaluation

2.4.1 Comparison of clustering method

The experiment conducted in this study aimed to evaluate the effectiveness of clustering-based pairs trading. Specifically, various clustering methods, and metrics such as correlation, cointegration, and Hurst exponent were tested. The main

objective was to determine the optimal parameters that yield the highest Sharpe ratio and generate profitable trading signals. To achieve this, we applied these parameters to a dataset of stock prices and conducted backtesting to assess the performance of our proposed clustering-based pairs trading strategy. The primary clustering results and comparison are as the following:

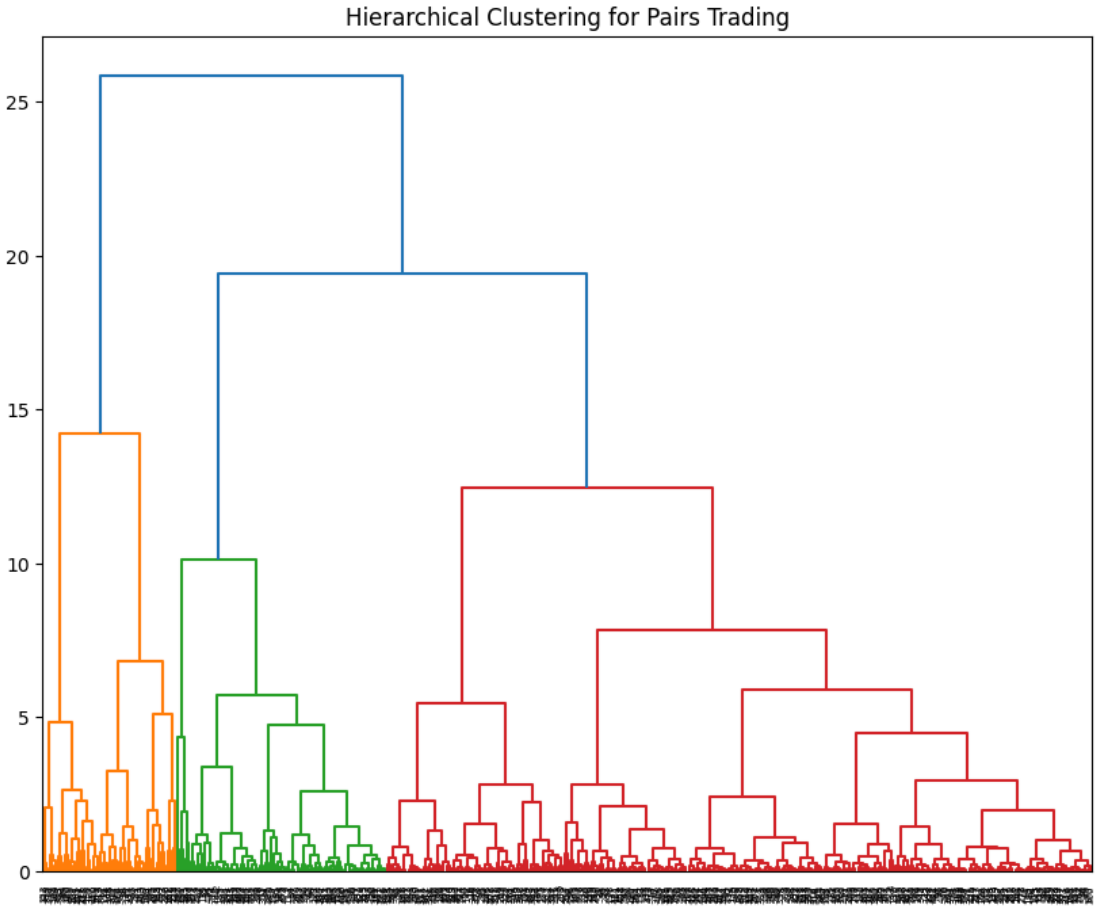


Figure 14: Hierarchical structure

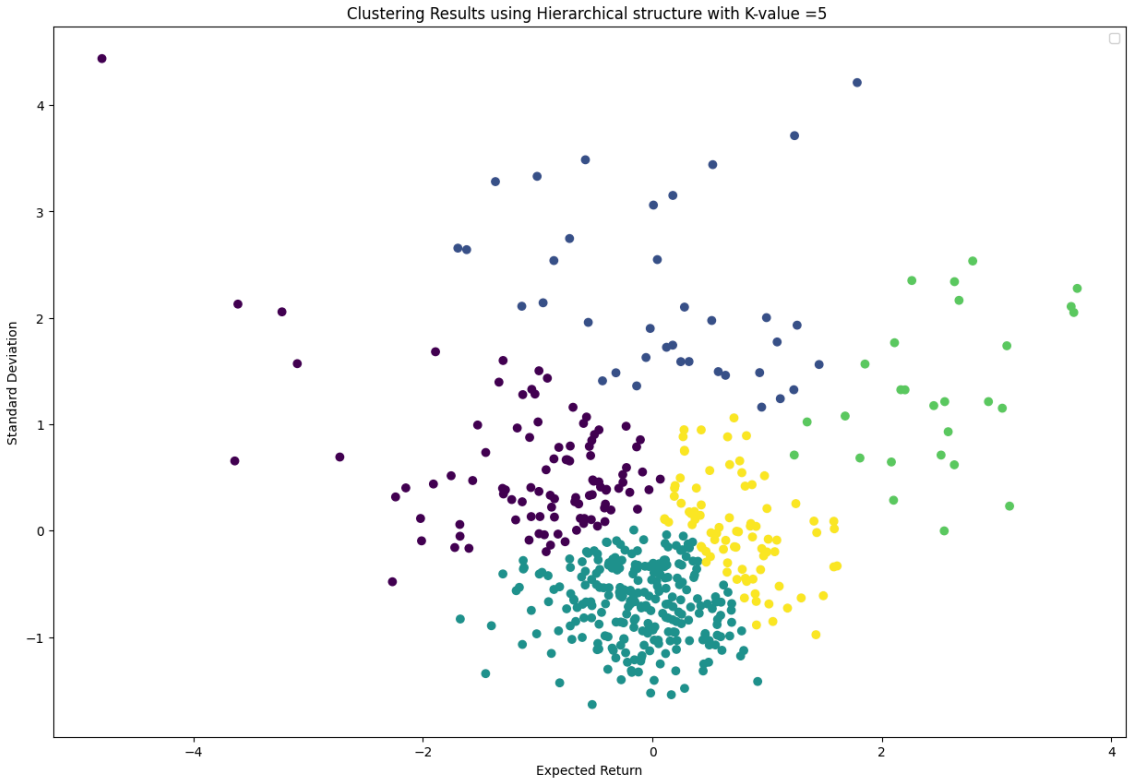


Figure 15: Hierarchical clustering result

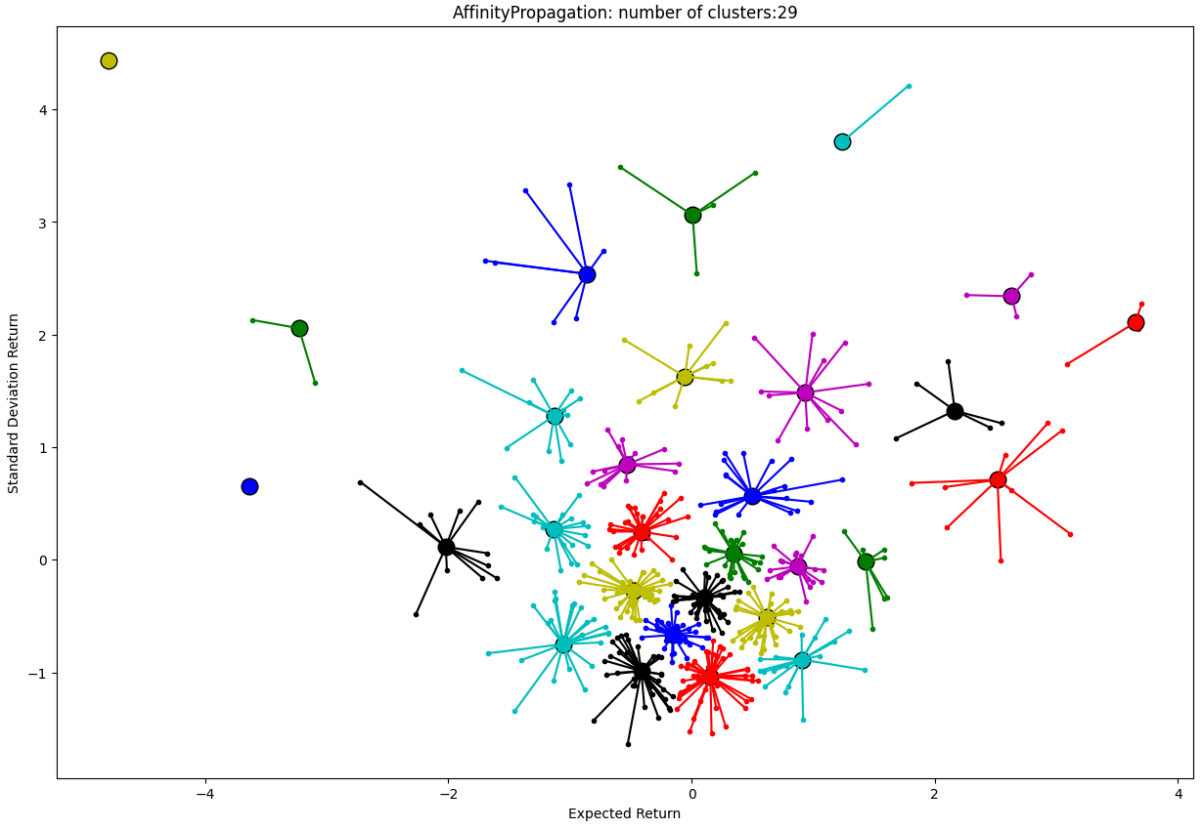


Figure 16: Affinity Propagation result

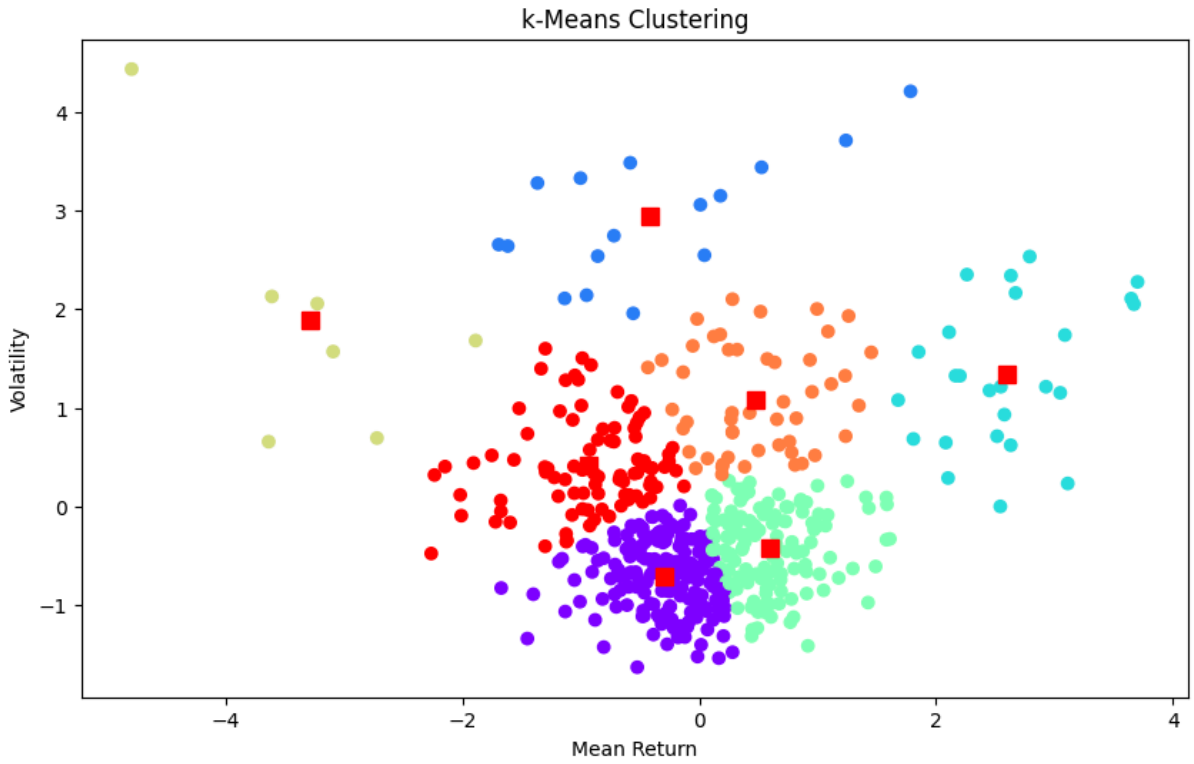


Figure 17: K-means clustering result

Clustering models comparison (Number of Clusters)	
Clustering mechanism	Clusters
K-means	7
Hierarchical Clustering	5
Affinity Propagation Clustering	29

Table 2: Comparison of a number of clusterings

Clustering models comparison (Silhouette score)	
Clustering mechanism	Silhouette score
K-means	0.359156
Hierarchical Clustering	0.349695
Affinity Propagation Clustering	0.324264

Table 3: Comparison of silhouette score

From the above, the K-means clustering has the higher silhouette score, indicating that the data point fits the best in the K-means clustering. Therefore, the clustering-based pair trading will be primarily based on the **K-means clustering** for the selection of the pair. The number of clusters is now 7 and the Number of Pairs is 60798 after sorting.

2.4.2 Performance of parameters and threshold

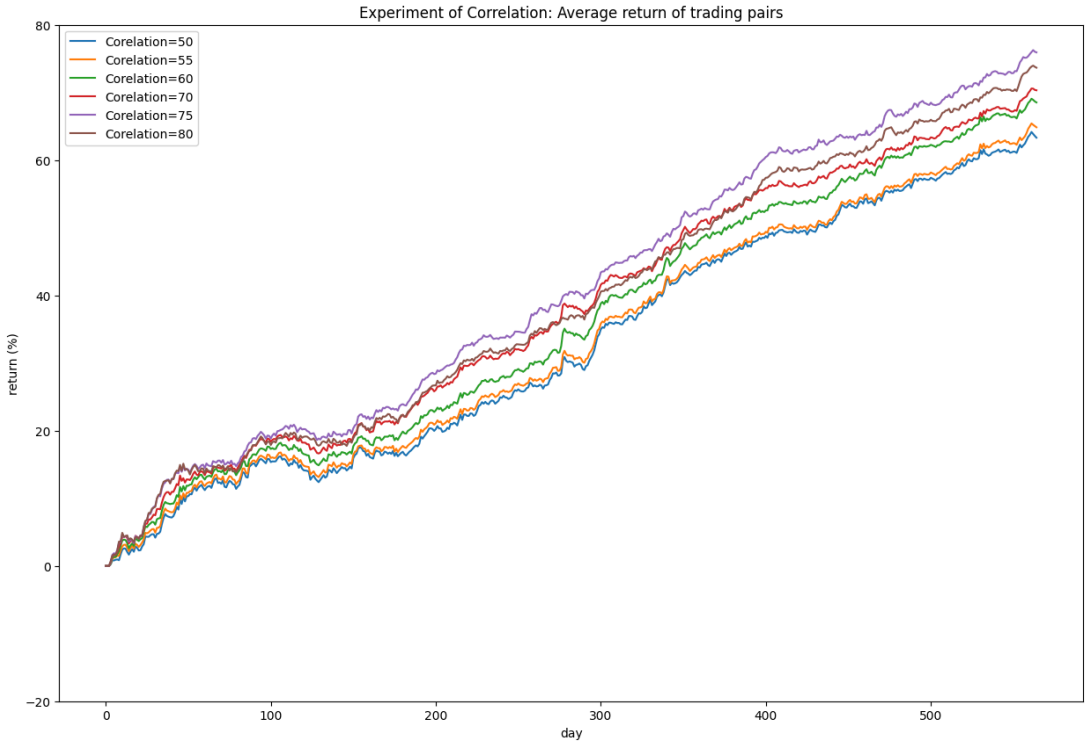


Figure 18: Experiment of correlation

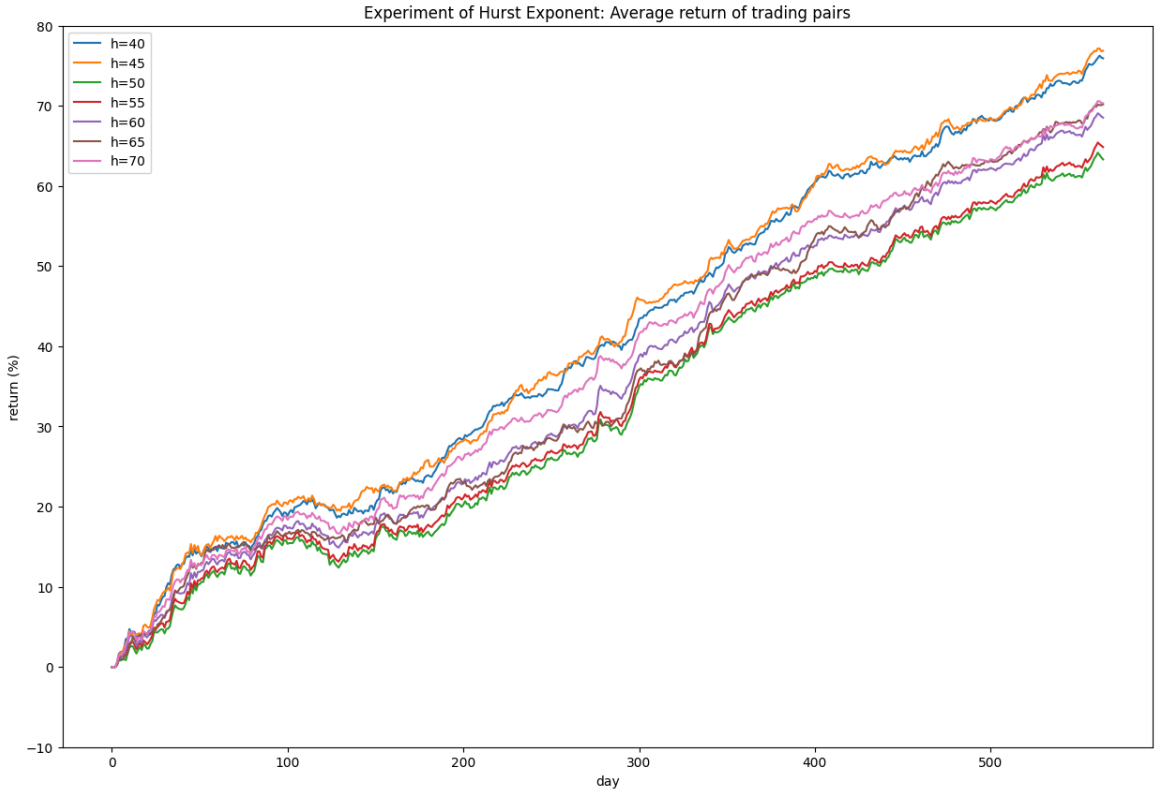


Figure 19: Experiment of correlation

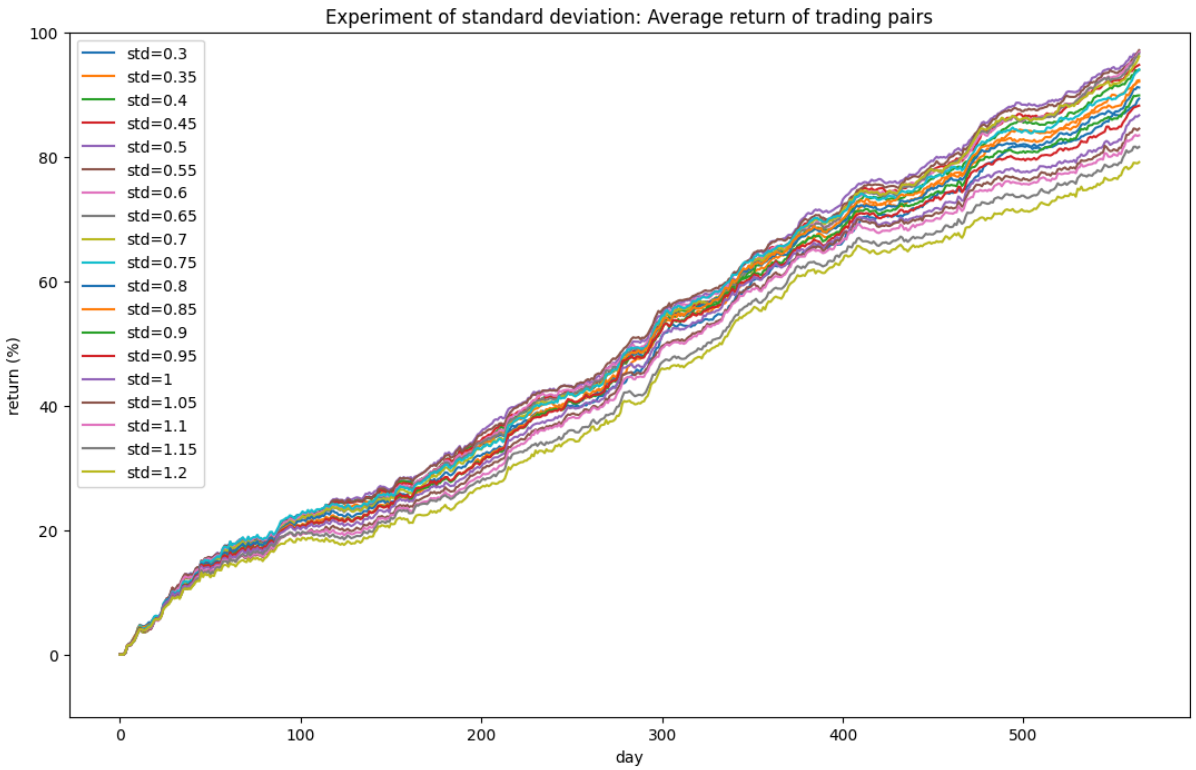


Figure 20: Experiment of correlation

2.4.3 Evaluation of the trading performance

After the testing by trial and error, we can ascertain that the optimal correlation is 0.7, the Hurst exponent is 0.4, and the standard deviation used is 0.5. The final number of pairs is 65. By implementing a back-testing strategy that employs 6-month rolling windows, the results with the comparison of the benchmark S&P500 are as follows:

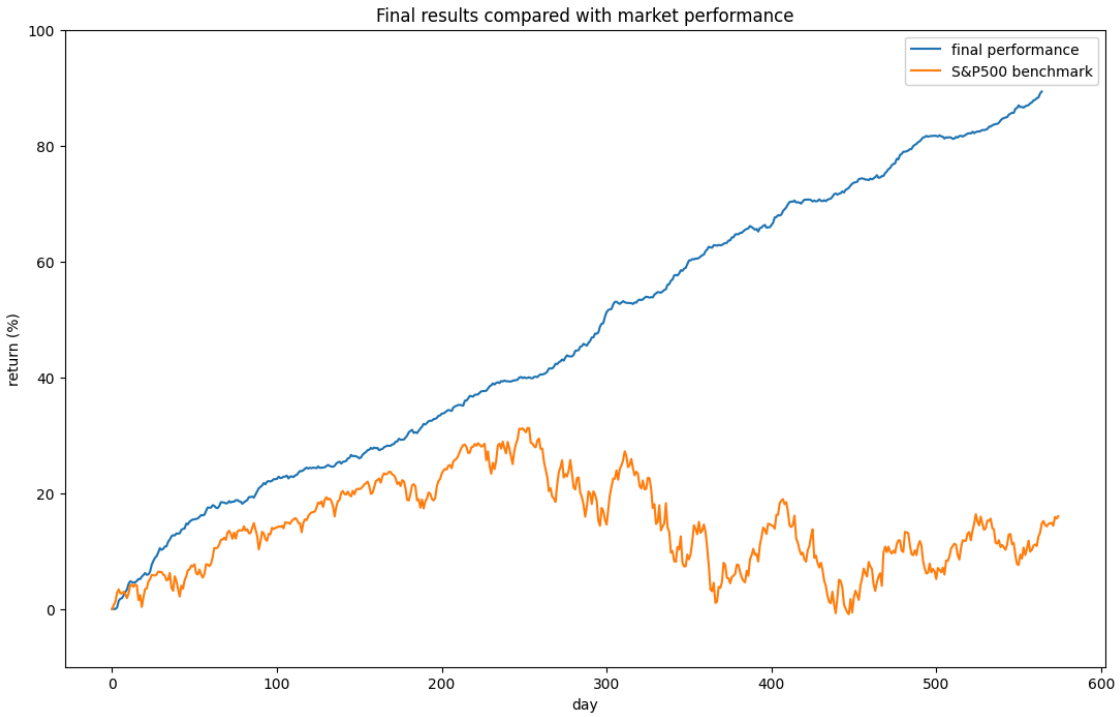


Figure 21: Comparison of final performance to S&P500 benchmark performance

2.5 Conclusion

Our strategy boasts a stable uptrend and a Sharpe ratio of 1.923. With the utilization of machine learning-based pairs trading, we can identify promising pairs with greater efficiency and potentially yield higher Sharpe ratios, surpassing those achieved through traditional pairs trading methods.

3. Reinforcement learning-based momentum trading

3.1 Dataset

Ticker data [Open, AdjClose, Volume] SPX futures in day intervals

Yfinance API between 2021-2023

3.2 Logic

3.2.1 Reason for RL-based

Momentum trading typically involves identifying assets that have exhibited strong performance in the recent past and betting that this trend will continue. However, this approach may not always work, as market conditions can change quickly and unpredictably. RL, on the other hand, uses a trial-and-error approach to learn the optimal trading strategy based on current market conditions. RL can potentially handle more complex scenarios where traditional trading strategies may struggle.

3.2.2 Concepts for the reinforcement learning model

The basic idea behind RL trading is to model the trading process as a Markov decision process (MDP), where the state of the system is described by a set of variables that capture relevant market data, such as asset prices and volumes, and the actions of the trader. The goal is to learn a policy that maps states to actions, such that the trader can maximize its cumulative reward over time. In this strategy, we will explore two reinforcement models, the Markov decision process and Deep Q-Learning.

In the reinforcement learning framework, we use a discount factor, denoted as γ, to discount future rewards. This is because, in a long-term trading scenario, rewards obtained in the distant future are less valuable than the rewards obtained in the immediate future. The discounted cumulative reward at a particular time step t is given by the formula: $R_t = \sum_{i=t}^{T} \gamma^{i-t} r_i$ where T is the final time step, γ is the discount factor, and r_i is the reward obtained at time step i.

To train the agent to maximize the expected cumulative reward, we use a loss function. One commonly used loss function in reinforcement learning is the mean

squared error (MSE) between the predicted Q-value and the target Q-value, denoted as L(Q,Q_{target}):

L(Q,Q_{target})=$(Q_{target} - Q)^2$ where Q is the predicted Q-value and Q_target is the target Q-value. The target Q-value is calculated as: $Q_{target} = r + \gamma\, max_{\alpha'} Q(s', a'; \theta^{-})$. By minimizing the MSE loss function, the agent learns to accurately predict the optimal action-value function and chooses actions that maximize the expected cumulative reward. In this project, we would try to use different ensemble methods and reinforcement algorithms to test the performance.

3.3 Models' Context

Markov decision process (MDP)

In an MDP, the state of the system at time t is denoted by s_t. The trader selects an action a_t from a set of possible actions A, based on its current observation of the state. The action causes the system to transition to a new state s_t with probability $p(s_{t+1}|s_t, a_t)$, and results in a reward r_t. The goal is to learn a policy $\pi(s)$ that maps states to actions, such that the expected cumulative reward over time is maximized. The Q-learning algorithm iteratively updates the action-value function using the Bellman equation:

$$Q(s_t, a_t) = Q(s_t, a_t) + \alpha\,(r_t + \tau\, max_{a'} Q(s_{t+1}, a') - Q(s_t, a_t))$$

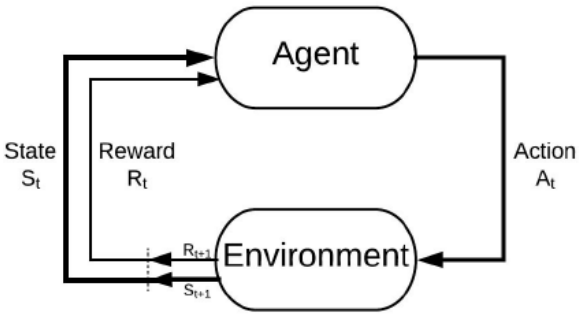


Figure 22: simple diagram of MDP process

In our RL model, the agent is the trader. The environment is the market. The state is the technical indicators such as daily moving averages, AdjClose, etc. The actions could be held, long or short, denoted as {-1,0,1}. The reward is profit. The training topics are shown in the following

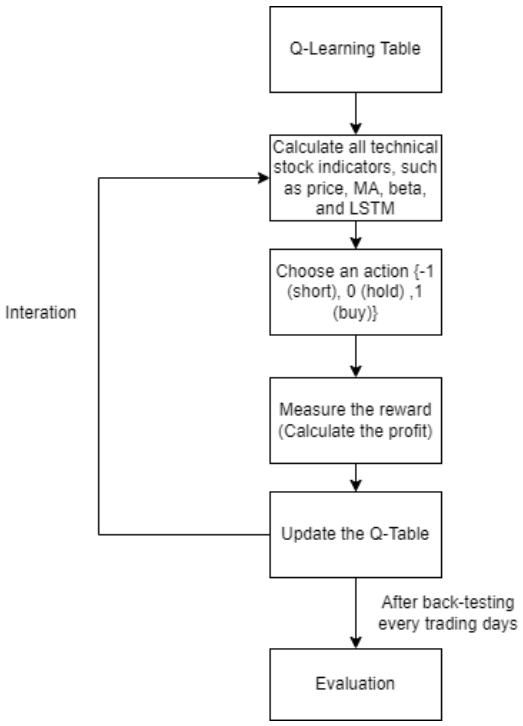


Figure 23: Example of trading logic of MDP with Q-Learning

First, the data and the technical data are fed into the models. Then the agent will decide whether buy, sell, or hold from the condition of the environment. The reward is calculated as a return and the agent is supposed to have the ability to adjust the action by the Q-learning table modeling by a loss function, discount factor, policy gradient, and learning rate. Finally, we would evaluate the performance in the back-testing system.

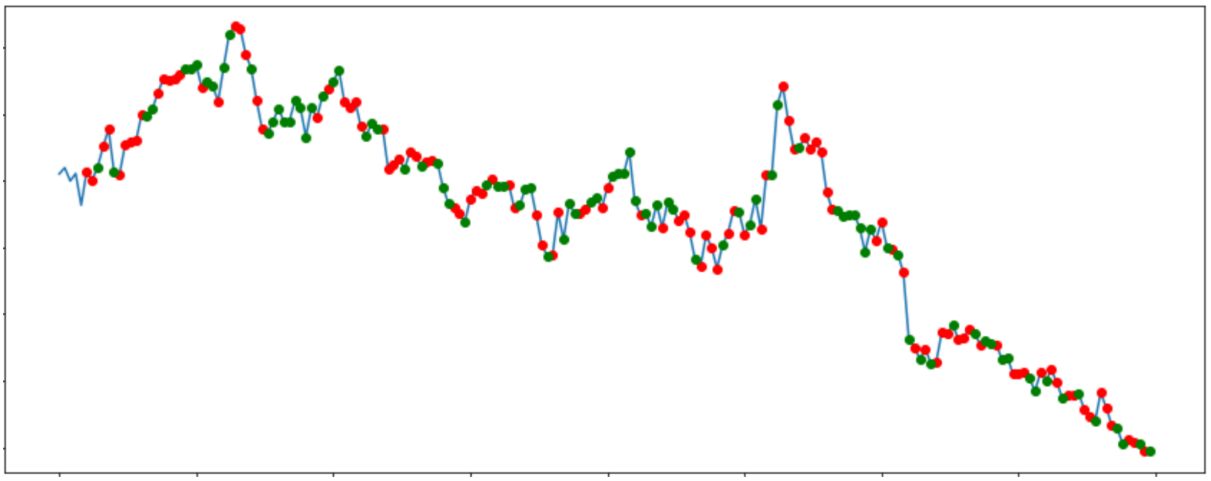


Figure 24:Example of trading action made by the agent

3.4 Evaluation

3.4.1 Comparison of reinforcement learning model

The experiment conducted in this study aimed to evaluate the effectiveness of different reinforcement learning models. Specifically, various algorithms such as, PPO, A2C, DDPG, Q-learning, and SARSA were tested. The main objective was to determine the reinforcement yield with the highest Sharpe ratio and generate profitable trading signals. To achieve this, we applied these parameters to a dataset of stock prices and conducted backtesting to assess the performance of our proposed clustering-based pairs trading strategy. The primary clustering results and comparison are as the following:

Reinforcement learning algorithm comparison	
Algorithm	Sharpe ratio
Market	0.102
PPO	0.425
A2C	0.402
DDPG	0.304
Q-learning	0.555
SARSA	0.582

Table 4: Comparison of Trading performance of different reinforcement learning algorithm

From above, among all five algorithms, SARSA has the highest Sharpe ratio of 0.782, which is better than the market return. It may be attributable to the fact that ARSA is an on-policy RL algorithm that it learns a policy by updating the Q-value estimates based on the current policy being followed. This can be advantageous in situations where the behavior of the agent affects the environment or when exploration is necessary.

3.5 Conclusion

Although reinforcement learning trading yields a positive Sharpe ratio, the average Sharpe ratio is around 0.45 which could not be classified as a promising trading strategy compared to most of the other existing strategies or other strategies in this project.

4. Multiple Model-based Binary Classification

4. 1 Dataset

Yfinance's Ticker data (From 2007 to 2017)

Stocks and forex in daily intervals with the top 10 market capitalization

4. 2 Logic

The utilization of all sci-kit-learn's models for binary classification

1. **Classifiers for classification problem**
2. **Regressors for classification problem**

4.2.1 Reason for Regressors as classifiers

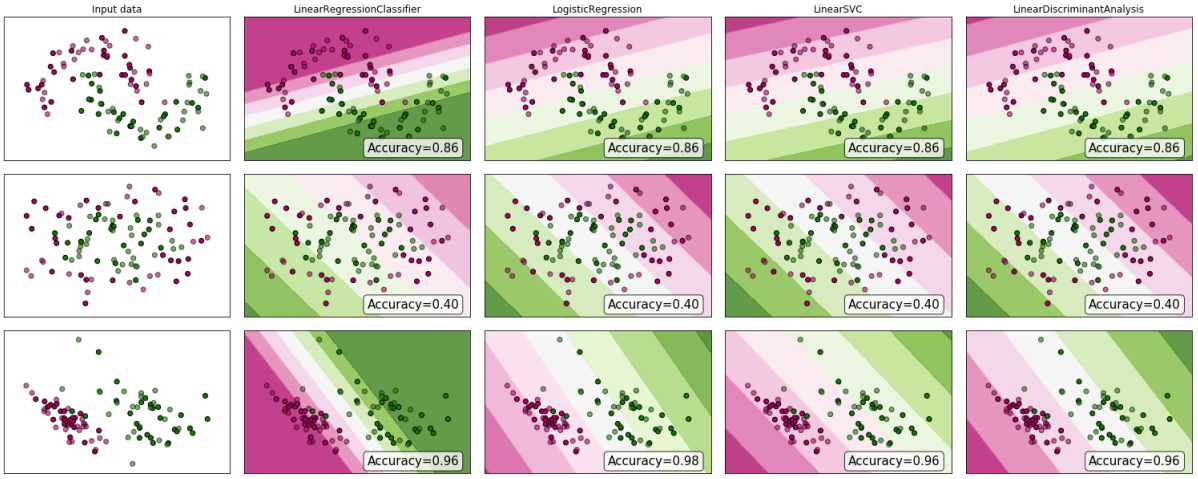


Figure 25 : Classification's accuracy of classifiers and regressors

In the above feature space, a regression model can behavior similar to classifiers, providing a hyperplane that can be used to partition the classes. Thus, regressors can have alike accuracy with classifiers.

Feature discretization

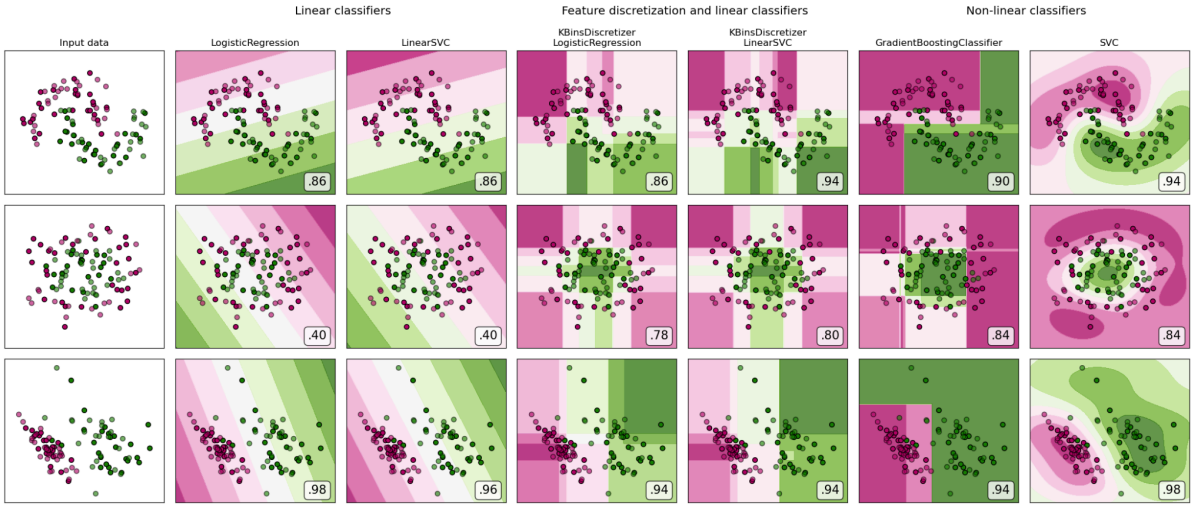


Figure 26: Feature discretization on linear and non-linear separatable datasets

4.2.2 Concepts of Dimension Reduction

In this instance, the initial two rows show datasets that are not linearly separable, specifically the moons and concentric circles. The third row represents a dataset that is approximately linearly separable. Employing feature discretization significantly enhances the performance of linear classifiers on non-separable datasets. However, on the linearly separable dataset, the performance of linear classifiers is diminished by feature discretization.

4.3 Models' Context

With the hope of enhancing performance, we employ a similar ideology of feature discretization to regressors' prediction. Prediction of classifiers and regressors are provided for comparison purposes.

Our models intend to fit the data with continuous regression, truncating the continuous prediction to yield discrete classifications. In fact, the backbone of logistic regression is pure linear regression. Therefore, the logistic function is a linear function wrapped in a sigmoid function. Albeit with a narrower decision boundary, regressors can achieve the same accuracy as classifiers. Our models hereby approach classification through regression.

<u>Models' result combinations</u>

(68 models * (10 forexes + 10 stocks) * 2 scalers)

results = 2720 [theory] > 2689 [reality]

- 68 regressors/classifiers
- 20 assets - major forexes + stocks
- 2 scalars - standard + max min (0, 100)

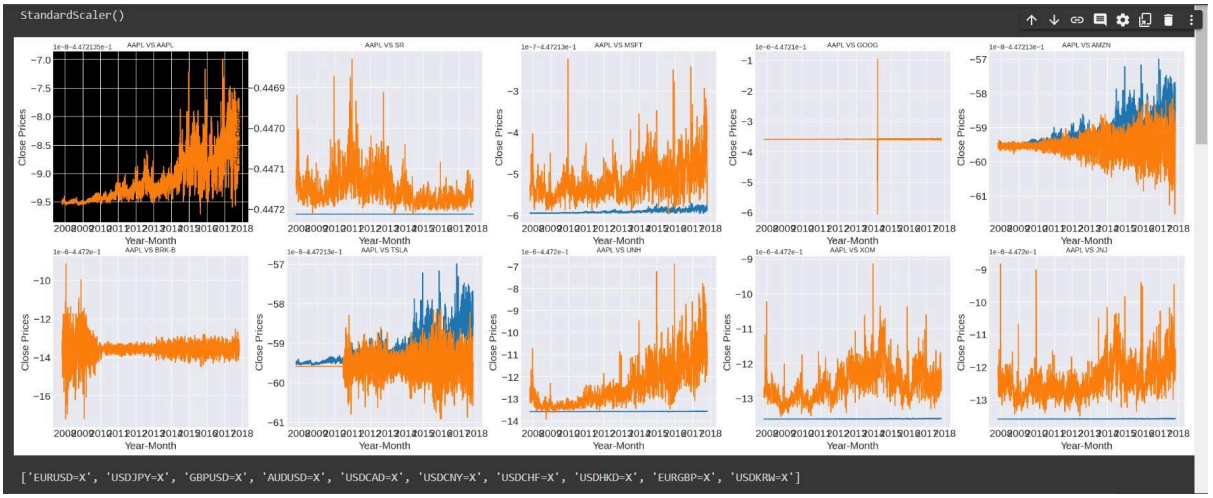


Figure 27: Scaled data shape

The data is obtained from yahoo finance. It only has numeric features. I generate two new datasets from the original datasets as I want to inherit the historical impact of price data from a long and short timeframe. I use both the min max scaler and standard scaler to scale the data. Also, I shift all the Decision columns up by one row as a prediction should be made one day before the result has appeared. NA values are filled with median or None words.

I feed scalers, models, and metrics into the array to evaluate their independent impacts. You can observe that simple classifiers already perform well on the datasets. It may be overkill to use complex models like CNN with customized layers. Generally, my script can be applied to both regression and classification problems with slight modification as most models are modulized into arrays.

<u>Scikit-learn models</u>

We have utilized most models from Probability Calibration, Dummy estimators, Ensemble Methods, Gaussian Processes, Linear classifiers, Classical linear regressors, Regressors with variable selection, Bayesian regressors, Outlier-robust regressors, Generalized linear models (GLM) for regression, Miscellaneous, Naive

Bayes, Nearest Neighbors, Neural network models, Semi-Supervised Learning, Support Vector Machines, and Decision Trees to compare their effectiveness.

Implementation

The models are supervised learning models. They use all of the [sklearn] models with default parameter settings. Prediction results of regressors are normalized to binary output [0, 1] to compare with classifiers. Our datasets are separated into two main categories - datasets that inherit only yesterday's price impacts and datasets that inherit the historical price impact. My models analyze the closing price of the historical market. If the previous closing price is lower than the current closing price, it is classified as a buying opportunity and vice versa.

These models treat trading as a binary classification problem and run at fixed intervals - (every day/week/months).

Notice

1. Please notice regressors are used with normalization.
2. All classification metrics are used
3. Use asset_report(, True) to see the detailed report
4. Scaling is done horizontally
5. Forexes and Stocks with the top 10 market capitalization are tested

Scalar

1. MinMaxScaler(feature_range=(0, 100)),	2. StandardScaler()

Asset

Forex	Stock
1. 'EURUSD=X', # EUR/USD (euro/US dollar) 2. 'USDJPY=X', # USD/JPY (US dollar/Japanese yen)	1. 'AAPL', # AAPL 2. 'SR', # SR 3. 'MSFT', # MSFT 4. 'GOOG', # GOOG 5. 'AMZN', # AMAZ

3. 'GBPUSD=X', # GBP/USD (British pound/US dollar) 4. 'AUDUSD=X', # AUD/USD (Australian dollar/US dollar) 5. 'USDCAD=X', # USD/CAD (US dollar/Canadian dollar) 6. 'USDCNY=X', # USD/CNY (US dollar/Chinese renminbi) 7. 'USDCHF=X', # USD/CHF (US dollar/Swiss franc) 8. 'USDHKD=X', # USD/HKD (US dollar/Hong Kong dollar) 9. 'EURGBP=X', # EUR/GBP (euro/British pound sterling) 10. 'USDKRW=X', # USD/KRW (US dollar/South Korean won)	6. 'BRK-B', # BRK-B 7. 'TSLA', # TSLA 8. 'UNH', # UNH 9. 'XOM', # XOM 10. 'JNJ', # JNJ

Model

Probability Calibration	Dummy estimators	Ensemble Methods
1. calibration.CalibratedClassifierCV(),	2. dummy.DummyClassifier(), 3. dummy.DummyRegressor(),	4. ensemble.AdaBoostClassifier(), 5. ensemble.AdaBoostRegressor(), 6. ensemble.BaggingClassifier(), 7. ensemble.BaggingRegressor(), 8. ensemble.ExtraTreesClassifier(), 9. ensemble.ExtraTreesRegressor(), 10. ensemble.GradientBoostingClassifier(), 11. ensemble.GradientBoostingRegressor(), 12. ensemble.IsolationForest(),

		13. ensemble.RandomForestClassifier(), 14. ensemble.RandomForestRegressor(), 15. ensemble.HistGradientBoostingRegressor(), 16. ensemble.HistGradientBoostingClassifier(),
Gaussian Processes 17. gaussian_process.GaussianProcessClassifier(), 18. gaussian_process.GaussianProcessRegressor(),	Linear classifiers 19. linear_model.LogisticRegression(), 20. linear_model.LogisticRegressionCV(), 21. linear_model.PassiveAggressiveClassifier(), 22. linear_model.Perceptron(), 23. linear_model.RidgeClassifier(), 24. linear_model.RidgeClassifierCV(), 25. linear_model.SGDClassifier(), 26. linear_model.SGDOneClassSVM(),	Classical linear regressors 27. linear_model.LinearRegression(), 28. linear_model.Ridge(), 29. linear_model.RidgeCV(), 30. linear_model.SGDRegressor(),
Regressors with variable selection 31. linear_model.ElasticNet(), 32. linear_model.ElasticNetCV(), 33. linear_model.Lars(), 34. linear_model.LarsCV(), 35. linear_model.Lasso(), 36. linear_model.LassoCV(),	Bayesian regressors 42. linear_model.ARDRegression(), 43. linear_model.BayesianRidge(),	Outlier-robust regressors 44. linear_model.HuberRegressor(), 45. linear_model.TheilSenRegressor(),

37. linear_model.LassoLars(), 38. linear_model.LassoLarsCV(), 39. linear_model.LassoLarsIC(), 40. linear_model.OrthogonalMatchingPursuit(), 41. linear_model.OrthogonalMatchingPursuitCV(),		
<u>Generalized linear models (GLM) for regression</u> 46. linear_model.PoissonRegressor(), 47. linear_model.TweedieRegressor(),	<u>Miscellaneous</u> 48. linear_model.PassiveAggressiveRegressor(),	<u>Naive Bayes</u> 49. naive_bayes.BernoulliNB(), 50. naive_bayes.GaussianNB(),
<u>Nearest Neighbors</u> 51. neighbors.KNeighborsClassifier(), 52. neighbors.KNeighborsRegressor(), 53. neighbors.RadiusNeighborsRegressor(), 54. neighbors.NearestCentroid(),	<u>Neural network models</u> 55. neural_network.MLPClassifier(), 56. neural_network.MLPRegressor(),	<u>Semi-Supervised Learning</u> 57. semi_supervised.LabelPropagation(), 58. semi_supervised.LabelSpreading(),
<u>Support Vector Machines</u> 59. svm.LinearSVC(), 60. svm.LinearSVR(), 61. svm.NuSVR(), 62. svm.OneClassSVM(), 63. svm.SVC(), 64. svm.SVR(),	<u>Decision Trees</u> 65. tree.DecisionTreeClassifier(), 66. tree.DecisionTreeRegressor(), 67. tree.ExtraTreeClassifier(), 68. tree.ExtraTreeRegressor(),	

4.4 Evaluation

```
total result: 2689

Best 10 models:

 NuSVR                13
MLPClassifier         12
GaussianNB            11
LinearSVR             11
ARDRegression         10
TweedieRegressor      10
LassoLars             10
ElasticNetCV          10
MLPRegressor          10
RidgeCV               10
Name: model, dtype: int64

Worst 10 models:

 KNeighborsClassifier              4
KNeighborsRegressor                4
RandomForestClassifier             4
DecisionTreeClassifier             4
GradientBoostingRegressor          4
BaggingClassifier                  4
HistGradientBoostingRegressor      4
AdaBoostClassifier                 4
GradientBoostingClassifier         2
SGDOneClassSVM                     2
Name: model, dtype: int64
```

Figure 28: Experiment of correlation

4.4.1 Comparison among Distinctive Models

With over 2000 results generated, we only evaluate the top 500 results. For instance, NuSVR has 13 prediction results that are within the top 500 results. Therefore, it is highly rated. SGDOneClassSVM, on the other hand, is rated poorly as it fails to deliver constancy. It has only two results that are within the top 500 models. Therefore, the higher the occurrence rate, the better the classifiers.

<u>Top 10 models</u>

1. **NuSVR - (13)**

In a similar fashion to NuSVC, the regression model employs a nu parameter to govern the number of support vectors. Contrary to NuSVC, where nu takes the place of C, the regression model displaces the epsilon parameter of epsilon-SVR with nu. lib SVM is the backrock of NuSVR.

$$[\frac{1}{n}\sum_{i=1}^{n} max(0,\ 1 - y_i(w^T x_i - b))] + \lambda||w||^2$$

(Support Vector Classifier)

2. **MLPClassifier - (12)**

It utilizes an iterative training process, where the model parameters are updated at each time step by computing the partial derivatives of the loss function. Additionally, a regularization term can be incorporated into the loss function to reduce overfitting by shrinking the model parameters.

The classifier is compatible with both dense numpy arrays and sparse scipy arrays.

3. **GaussianNB - (11)**

It is an implementation of the Gaussian Naive Bayes algorithm, which is used for classification. This algorithm makes the assumption that the likelihood of the features follows a Gaussian distribution.

$$P(x_i|y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} exp(-\frac{(x_i-\mu_y)2}{2\sigma_y^2})$$

(Gaussian Naive Bayes)

4. **LinearSVR - (11)**

It is a variant of SVR that uses the linear kernel parameter. However, it is implemented in lib Linear instead of lib SVM, which provides greater flexibility in loss functions and penalty selection. It can better handle large datasets. This class is compatible with sparse and dense input data.

5. **ARDRegression - (10)**

It is a technique used to train regression models by fitting their weights with an ARD prior. Gaussian distributions are used to calculate the models' weights. Additionally, estimates for the parameters lambda (which represent the precisions of the weight distributions) and alpha (which represent the precision of the noise distribution) are computed. These estimates are obtained through an iterative procedure called Evidence Maximization.

$$\hat{y} = \underset{k\in\{1,\dots K\}}{\arg max}\ p(C_k) \prod_{i=1}^{n} p(x_i|C_k)$$

(Naive Bayes classifier)

6. TweedieRegressor - (10)

The Tweedie distribution-based Generalized Linear Model is a statistical tool that can be employed to construct various GLMs, depending on the power parameter. By adjusting the power parameter, one can determine the underlying distribution that is best suited for a given analysis.

$$P_{\theta,\sigma^2}(Y \in A) = \int_A exp\left(\frac{\theta \bullet z - \kappa_p(\theta)}{\sigma_y^2}\right) \bullet v_\lambda(dz)$$

(Tweedie distribution)

7. LassoLars - (10)

It utilizes the Least Angle Regression (LARS) algorithm - a type of linear model that is regularized using an L1 prior.

$$\min_{\beta,\beta_0} \left\{\sum_{i=1}^{N} (y_i - \beta_0 - x_i^T\beta)^2\right\} subject\ to \sum_{j=1}^{p} |\beta_j| \leq t$$

(Least absolute shrinkage and selection operator)

8. ElasticNetCV - (10)

ElasticNet is a linear regression model that employs both L1 and L2 regularization methods to regulate the coefficients. This combination enables the model to learn a sparse representation, similar to Lasso, while also retaining the regularization properties of Ridge regression. The trade-off between L1 and L2 regularization is controlled by the l1_ratio parameter.

$$\min_w \frac{1}{2n_{samples}}||Xw - y||_2^2 + \alpha\rho||w||_1 + \frac{\alpha(1-\rho)}{2}||w||_2^2$$

(Elastic Net)

The ElasticNet algorithm is utilized for highly correlated data with multiple features. Lasso may arbitrarily select one of these features, while ElasticNet tends to select both.

Another practical advantage of ElasticNet is that it can inherit some of the stability properties of Ridge regression when dealing with rotation.

9. MLPRegressor - (10)

The Multi-layer Perceptron regressor is a type of machine learning model that minimizes the squared error using either LBFGS or stochastic gradient descent optimization algorithms.

10. RidgeCV - (10)

Ridge regression is a machine learning model that includes a cross-validation estimator. This estimator is designed to efficiently perform Leave-One-Out Cross-Validation by default.

$$\widehat{\beta}_R = (X^T X + \lambda I)^{-1} X^T y$$

(Simple ridge estimator)

Among the 500 results, below listed the worst 10 models. However, they may not be the worst models among the 2000 results. These models will not be investigated.

Worst 10 models

1. KNeighborsClassifier - (4)
2. KNeighborsRegressor - (4)
3. RandomForestClassifier - (4)
4. DecisionTreeClassifier - (4)
5. GradientBoostingRegressor - (4)
6. BaggingClassifier - (4)
7. HistGradientBoostingRegressor - (4)
8. AdaBoostClassifier - (4)
9. GradientBoostingClassifier - (2)
10. SGDOneClassSVM - (2)

Difficulties

As the scalars, assets, models, and metrics are modulized into arrays, I have to be

careful to ensure the correctness of multi-dimensional arrays. Due to the complex shape of the arrays, I have not included a K-fold cross-validation method in our training sets. A slightly non-static result is yielded as a compromise.

$$\frac{1}{n}\sum_{i=1}^{n}(y_i - \widehat{y_i})^2$$

(Mean Squared Error)

Originally, I intend to create a dataset that inherits the price impact of the historical date range. Although I have successfully created the dataset, the preprocessing time is too long so I finally abandon this method.

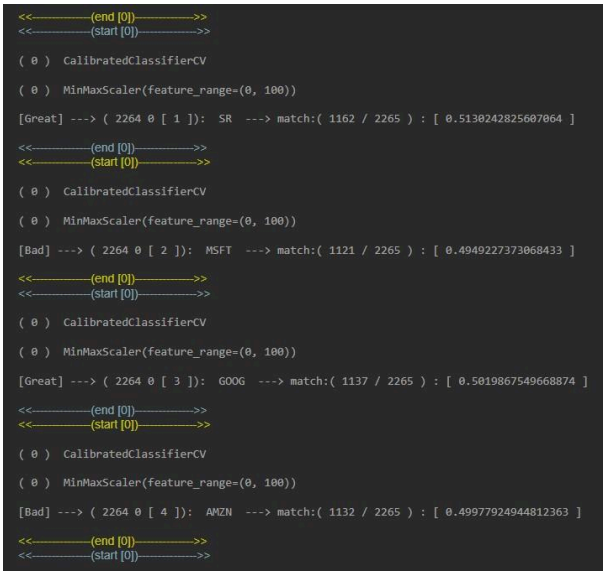


Figure 29: Classification of the same models on various assets

Also, I have decided to use Google Sheet API for data visualization and Redis for data storage. The application will be made with typescript. As we have to migrate the models with both frontend and backend, it is challenging for us.

4.5 Conclusion

Metric - accuracy

Top 100 models:

	accuracy	match	model	scaler	asset
2288	0.771100	1809/2346	LabelSpreading	StandardScaler()	USDCNY=X
2208	0.771100	1809/2346	MLPRegressor	StandardScaler()	USDCNY=X
2326	0.771100	1809/2346	LinearSVC	StandardScaler()	USDCNY=X
1219	0.771100	1809/2346	ElasticNet	StandardScaler()	USDCNY=X
2524	0.771100	1809/2346	SVR	StandardScaler()	USDCNY=X
861	0.771100	1809/2346	Perceptron	StandardScaler()	USDCNY=X
901	0.771100	1809/2346	RidgeClassifier	StandardScaler()	USDCNY=X
941	0.771100	1809/2346	RidgeClassifierCV	StandardScaler()	USDCNY=X
785	0.771100	1809/2346	LogisticRegressionCV	StandardScaler()	USDCNY=X
1535	0.771100	1809/2346	LassoLarsIC	StandardScaler()	USDCNY=X
1209	0.771100	1809/2346	ElasticNet	MinMaxScaler(feature_range=(0, 100))	BRK-B
1099	0.771100	1809/2346	Ridge	StandardScaler()	USDCNY=X
775	0.771100	1809/2346	LogisticRegressionCV	MinMaxScaler(feature_range=(0, 100))	BRK-B
1327	0.771100	1809/2346	LarsCV	MinMaxScaler(feature_range=(0, 100))	BRK-B
1690	0.771100	1809/2346	BayesianRidge	StandardScaler()	USDCNY=X

Figure 30: Models Ranking based on accuracy metrics

The final ranking is created according to the occupancy of models in the top 500 results. Although the dummy regressor has enlightening performance in USDCNY=X. Its rare occupancy is an implication of successful bad trade. As dummy classifiers fail to deliver constancy, our result is valid. Thereafter, regressors could have a similar effect with classifiers for classification.

Deprecated Models - Monte Carlo Simulation

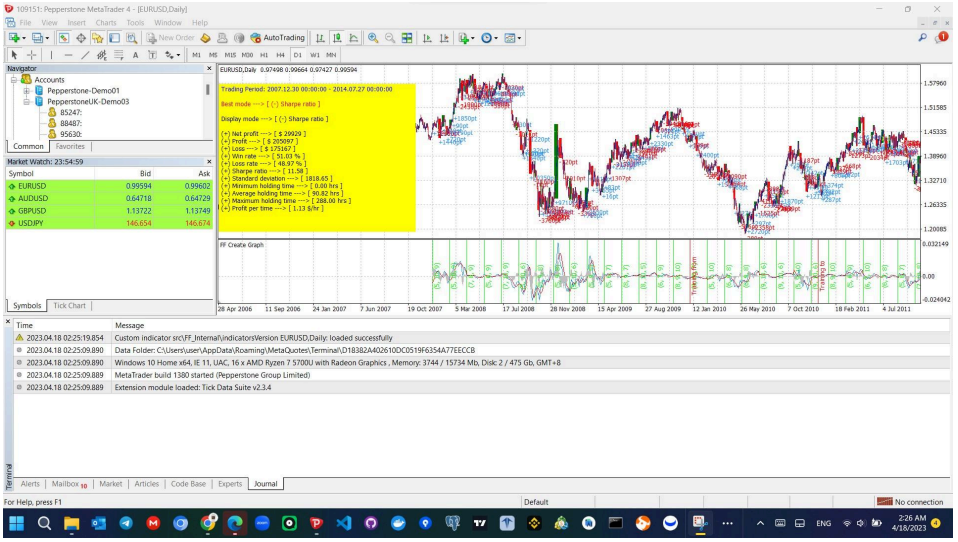


Figure 31: Trading on fixed batch with Approximation of MACD

Monte Carlos Method has been implemented in MT4 (explanation in Appendix C). Due to the law of probability, we could always get the correct answer with a large number of guesses. This is a MetaTrader indicator that approximates the best three MACD line values in a defined trading period. It is deprecated as our teams have decided to use Python in later meetings. As we need to remake its dependency in Python, this method is adopted.

Software Development

1 Overview

In this project, we aimed to develop user-friendly software for individual traders to trade with machine learning models without requiring extensive programming knowledge. To achieve this, we developed a user interface that provides a simple and abstract way for users to control the models.

The software we developed only provides the ability to control over strategy that is already created. Traders can then check the strategy performance by watching the return of the strategy against old data after adding the model to their account. They can then permit the model to trade for them if they believe that the model has a good backtest result. The system will do the trade on their behalf based on the model they turned on. To conclude, our system provides the following services:

1. Register and Login as different users
2. Add/Delete Model from user account (Direct Trade Decision is not permitted)
3. Start or Stop the Model from time to time
4. On-paper Trading/Real Trade

The system consists of three components: **the User Interface**, **the RESTful API Server**, and **the Trading Server**. Users can request to add or remove a model to their account through the user interface, and the RESTful API server will save the states into the database. The Trading server will then use the available models to make trading decisions, and the orders placed by each model will be saved in the database and linked to the users who added the model to their account.

We used data from the YFinance API to train our models, and we fetch the data when needed to make trade decisions. The trade decisions are then sent to the IBKR Broker API and saved in the database for future access.

Overall, this project aims to provide individual traders with practical and easy-to-use software that allows them to trade with advanced machine-learning models. By the end of this section, readers will have a better understanding of the development and implementation process of such software and its evaluation from a different perspective.

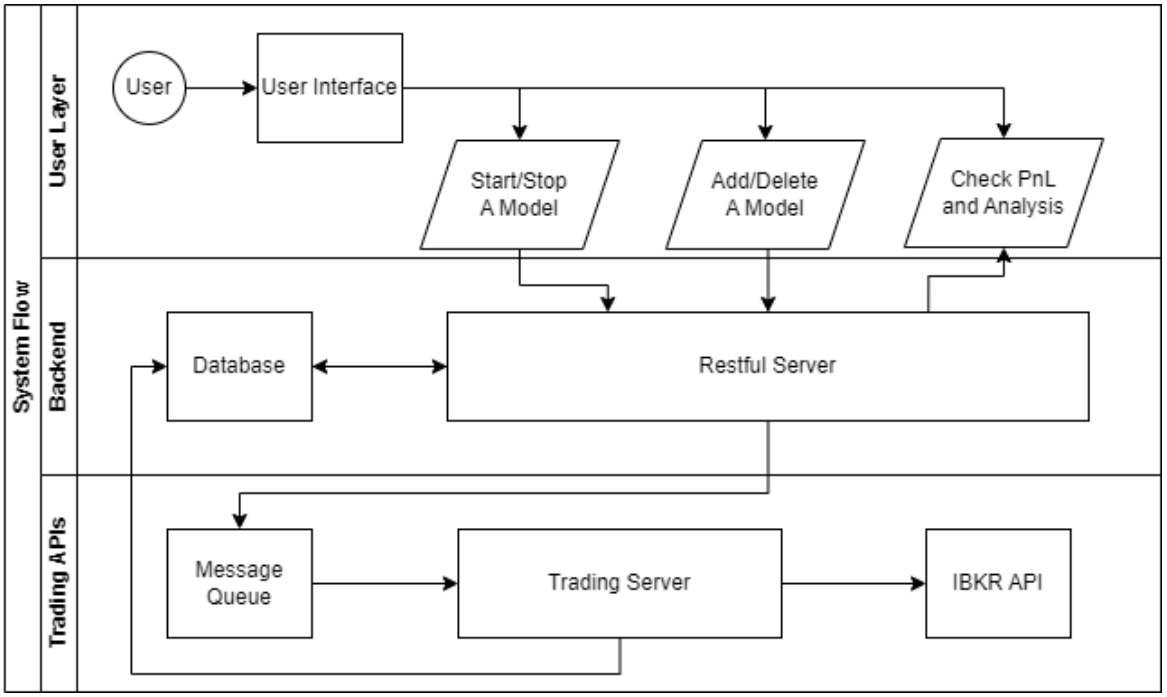
2 Design

2.1 Design - Data Flow

Our project aims to create a comprehensive platform with a web interface that provides multiple machine-learning and non-machine-learning models for trading. The platform allows users to choose from a variety of models to create a trading portfolio that best suits their needs.

Once a user selects a model, the platform will provide the corresponding profit and loss (PnL) data and analysis for the user. This will enable users to make informed decisions about which models to use and how to optimize their trading strategies.

To ensure ease of use, we have designed a user-friendly interface that allows users to navigate the platform and access the necessary information with ease. The overall flow of the platform is displayed in the diagram below:



(Figure 32: Unified view of system components)

By providing a comprehensive platform with a variety of models and analysis data, we aim to empower traders of all levels to make informed decisions and achieve their financial goals.

2.2 Design - User Interface

In order to provide users with a user-friendly and intuitive interface, we have designed a visualization tool that allows users to control the models and analyze the statistics of the implemented model through visual analysis. This visualization tool aimed to provide users with a clear and concise overview of the performance of each model, enabling them to make informed decisions about which models to use and how to optimize their trading strategies.

While the platform is designed to be accessible to traders of all levels, it is primarily intended for individual traders with a sufficient level of professional knowledge of both stocks and machine learning models. By starting or terminating models according to the market status, users can fully exploit the potential of the platform and achieve better trading outcomes.

Furthermore, the platform is designed to be flexible and adaptable to the needs and preferences of individual users. Users can customize their trading portfolio by adding/removing models according to their risk tolerance, investment objectives, and other factors, enabling them to achieve their financial goals in a way that is tailored to their unique circumstances.

System Requirements Specifications:

Features	Description
Overview	-
Add Models	Allow users to add their trained models into the system
Start/Terminate Models	Allow users to turn on/off a model according to the market state
Model Analysis	Allow users to view and analyze the model performance easier

(Table 5: System Requirement Specification)

Figma Design:

(Figure 33: Design of the user interface page)

3 Implementation

3.1 Implementation - Database

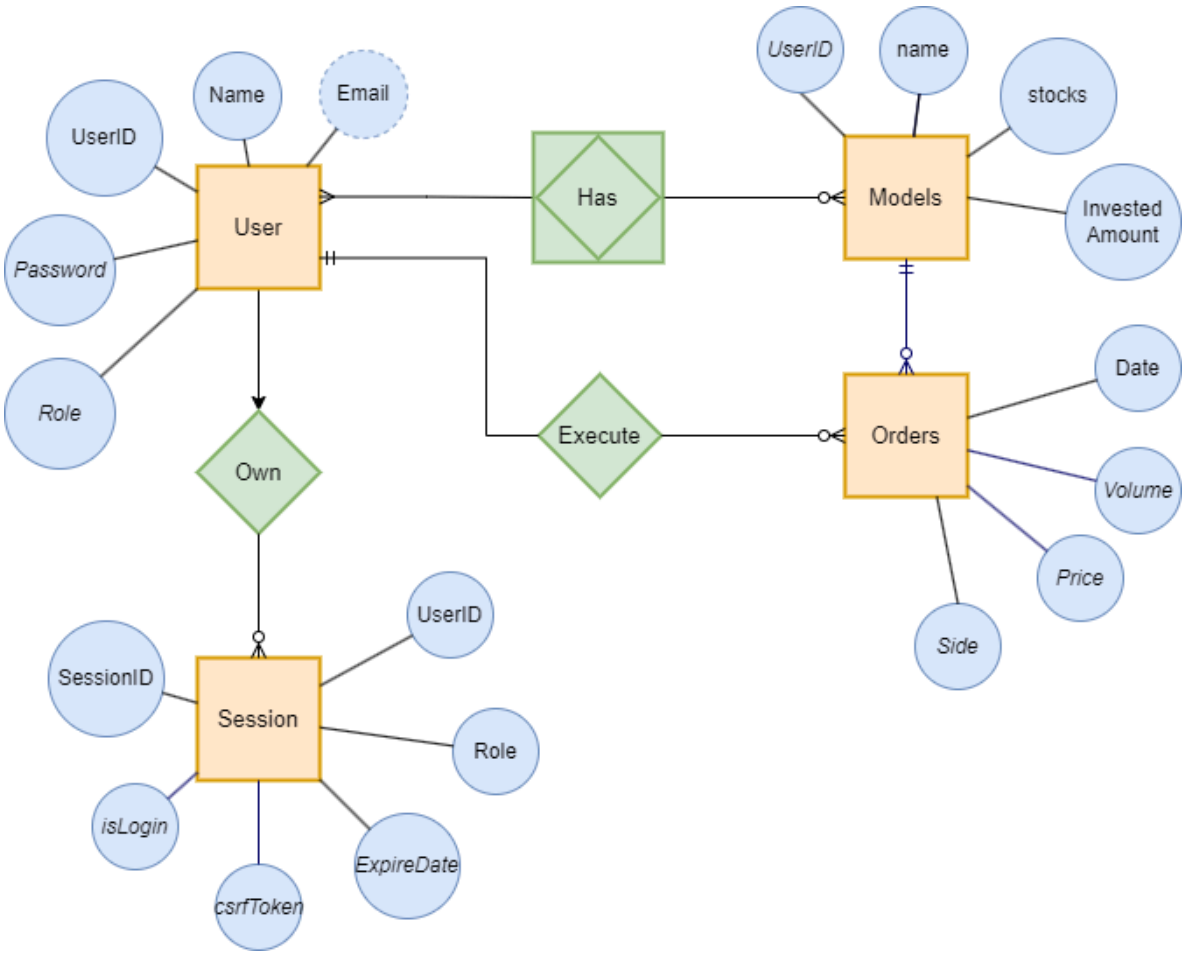
We leveraged MongoDB for efficient retrieval of user and model data, as well as Cassandra for ticker data retrieval.

The MongoDB implementation consists of two databases: Users and Models.

The Users database comprises two tables. The first table, "session," is responsible for managing login sessions and associated session data. The second table, "users," stores user data such as email, hashed passwords, and other relevant information.

The Models database also consists of two tables. The first table, "models," maintains details about the models, such as the model name, model path, corresponding user ID, and other relevant information. The second table, "trade_history," records trading history, including trading price, trading volume, and the ID of the order-placing model.

Following the construction of the database, we added multiple indexes, such as referencing ID fields, to improve query performance. Additionally, we implemented an internal cache for sessions to enhance the request time of APIs.



(Figure 34: ER Diagram of Mongodb)

3.2 Implementation - User Interface

For the development of the user interface, we utilized Next.js/Typescript. We selected this library primarily for its ability to load data both server-side and client-side, enabling us to provide a seamless user experience. Additionally, Next.js/Typescript offers numerous packages for drawing charts and manipulating data, making it an ideal choice for our project.

Some of the main packages we used include:

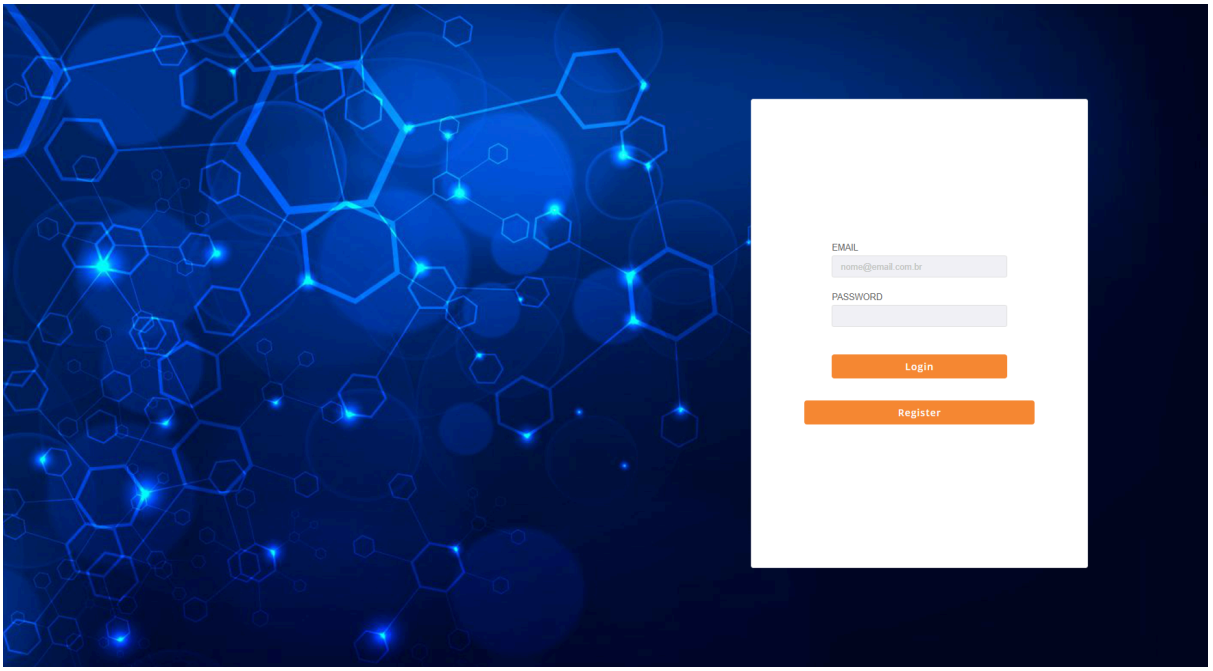
- Mui 2 - a library with multiple preset frontend components to provide professional interface
- Apex charts - a popular library for creating interactive data visualizations with React/Next.
- formik - a library for form-handling functions that provides a simpler and clearer form to verify and process data.
- Moment.js - a library for manipulating dates and times in JavaScript
- Vercel - a library for instant deployment

By leveraging these packages and the capabilities of Next.js/Typescript, we were able to create a highly functional and intuitive user interface that enables users to control the models and analyze the statistics of the implemented model through visual analysis. The result is a platform that empowers traders with the knowledge and tools they need.

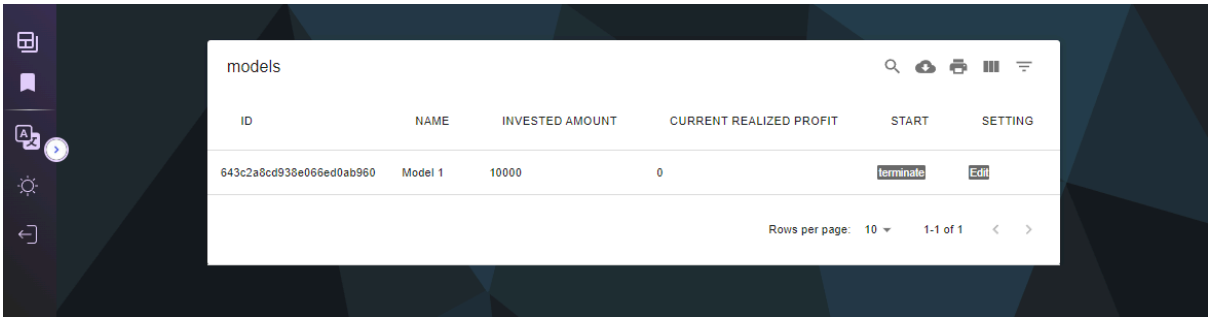
Performance optimization is achieved through the following technique:

1. React Hooks - useMemo(), useCallback(), useContext(), useReducer()
2. Lazy Loading - Suspense component & Skeletoning
3. Serverside Rendering - Load Big Data Set in Server

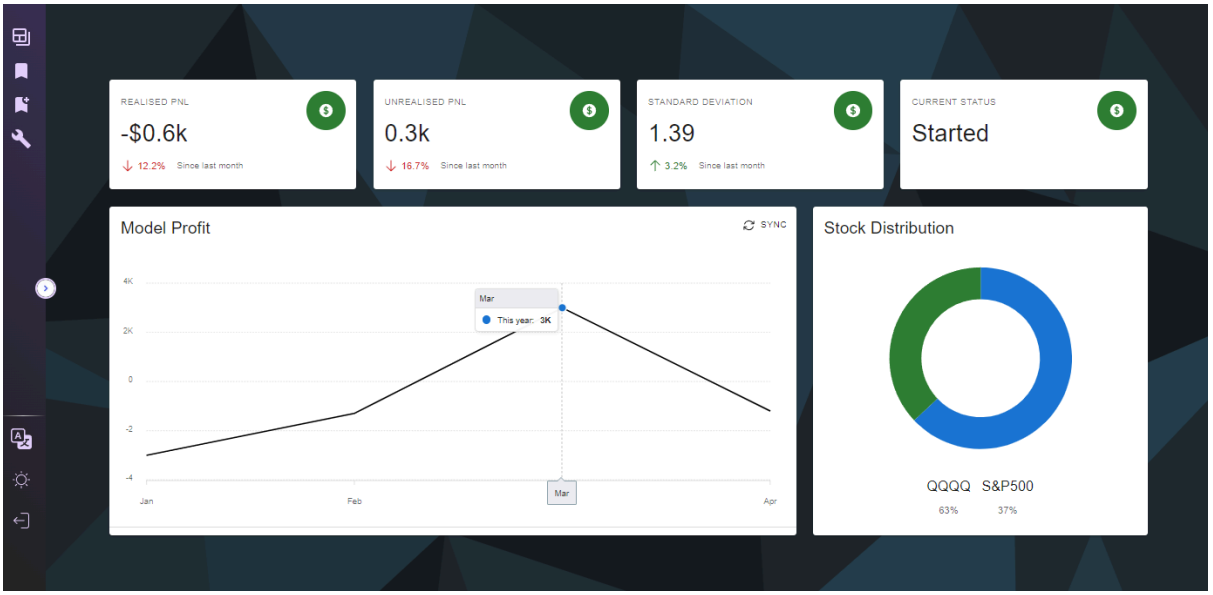
Some of the pages we developed:



(Figure 35: Login Page)



(Figure 36: Model View Page)



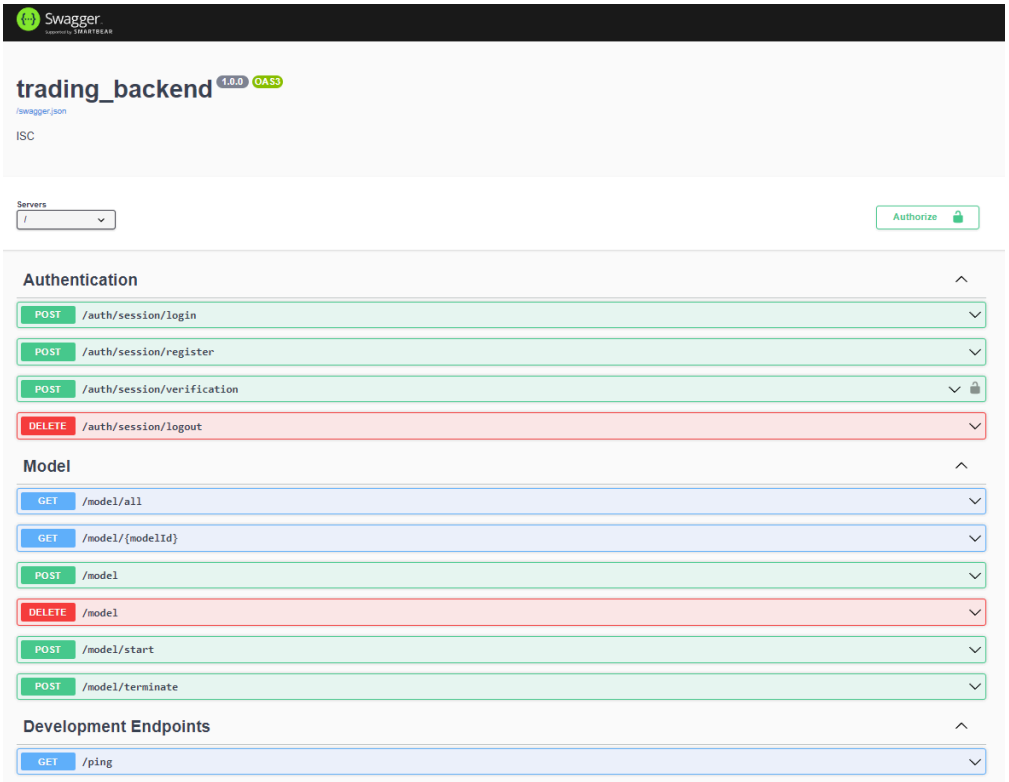
(Figure 37: Model Analysis Page)

3.3 Implementation - Restful API

To facilitate communication between the different components of our system, we developed an API server using Express.js/tsoa with RabbitMQ and MongoDB.

This API server enables users to upload and delete models, start or terminate models, and retrieve model data from the MongoDB database. We store the model and user data in MongoDB, and the API server updates the database upon receiving a POST request and retrieves data upon receiving a GET request. In addition, most requests generate a request JSON that is sent to RabbitMQ.

The trading server component of our system listens to the RabbitMQ queue and performs the corresponding actions in response to incoming requests. This approach enables us to efficiently manage user requests and ensure that the system is responsive and scalable.

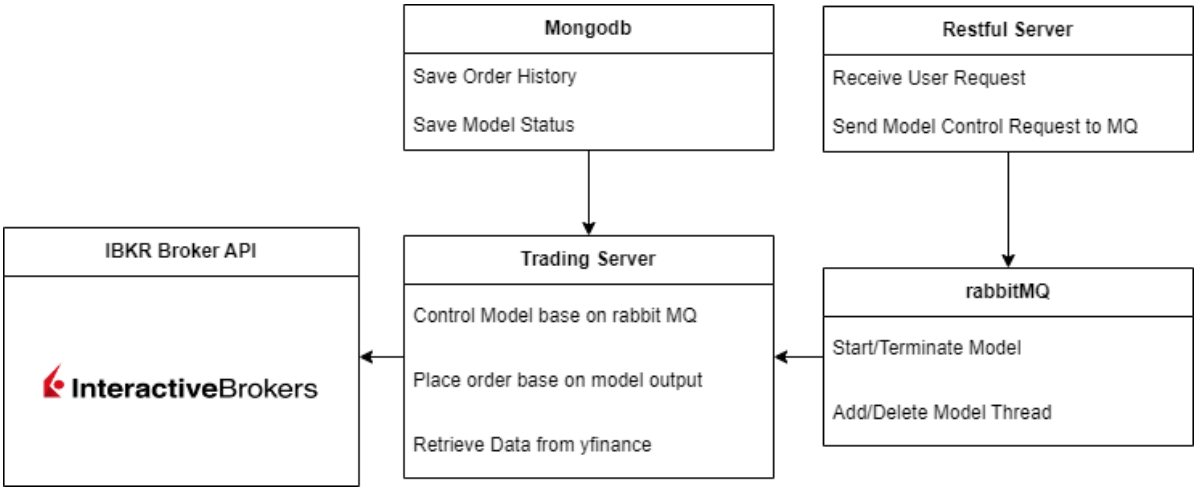


(Figure 38: Resful Server Swagger Endpoint Page)

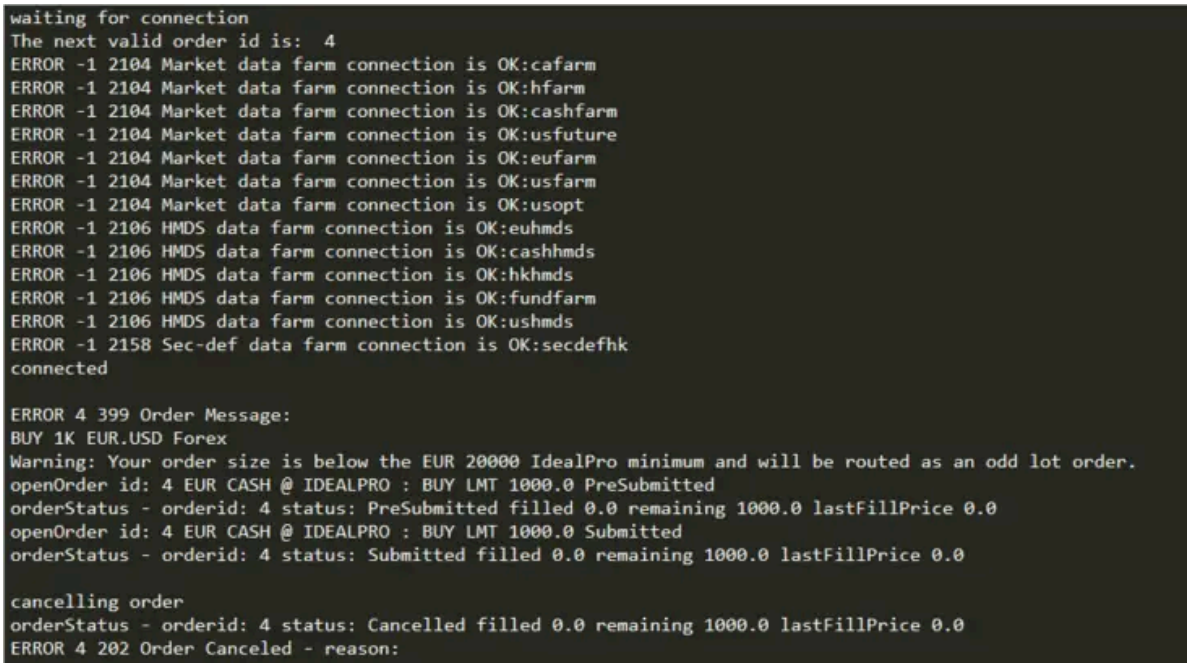
3.4 Implementation - Trading Server

We developed a Python trading system that receives structured JSON requests from RabbitMQ and executes the corresponding actions using packages such as Pika, NumPy, and TensorFlow.

The system is capable of adding, deleting, starting, or terminating a model based on user requests. Every model takes up one thread and do trading decisions on schedule. If the trading decision is made, the system will then sends an order to IBKR API Services.



(Figure 39: Trading Server Data Flow)



(Figure 40: Trading Server Log)

4 Testing

We carried out different tests based on the progress of our project. We conducted a unit test on each component to guarantee each part works properly, the integration test will confirm the system works. A detailed testing walk-through is listed below.

4.1 Test the Database

As our system mainly used database data storage and frontend data fetching, the testing of our database mainly focused on whether the data can be fetched properly.

4.2 Test the Trading Server

For testing on the trading server, we checked if the models can be start and end properly with message queue requests and order properly with IBKR API.

4.3 Test the Backend

It mainly serves the purpose of frontend data display and action handling. There are two controllers and two corresponding services for Authentication and Model

operations. Unit testing was conducted for each Service. We tested with some standard input and compare the output with the expected result for each service. The test result shows that the tests cover 92% of the services and our system passes all the tests.

4.4 Integration Testing

After each individual component has been tested, we tested the integration of all parts, and the front end controls them properly. To make sure everything works properly. We tested every button on the front end and monitor the backend log, trading server log, and database status.

5 Evaluation- User Acceptance

We conducted a small survey among five traders to assess the user experience of our trading application. The participants unanimously agreed that the user interface (UI) was user-friendly and the overall user experience (UX) was wonderful.

However, some traders reported that the on/off functionality of the system was slow, taking around 3-4 seconds on average to respond, which could negatively impact the user experience and fail to meet the trading requirements. To address this issue, we implemented lazy loading functionality to display the loading response when the button is pressed and return the response once the message is sent to the message queue, instead of returning a successful termination message.

In addition, a trader suggested further enhancements to the backtesting help and web model creation. We believe that these suggestions could be valuable features and will investigate them further.

Overall, the survey indicated that the web application was successful, with good acceptance by traders. We will continue to work on improving the functionality of the platform to provide a seamless and intuitive user experience.

Discussion

Overview

The discussion section of our thesis will focus on comparing the performance of our machine learning-based stock trading strategies with traditional trading methods. We will analyze the results obtained from our experiments and discuss the strengths and limitations of our approach. Specifically, we will compare our reinforcement learning momentum trading strategy with traditional momentum trading, our RNN-based portfolio optimization with traditional portfolio optimization, and our clustering-based pairs trading with traditional pairs trading, and with our traditional trading models and other traditional models done by other research. We will also suggest potential improvements that can be made to our models.

Comparison of our models:

1) Statistical pair trading & Clustering based pair trading

In our study, we developed two different pair trading strategies, including a statistical pair trading strategy and a K-means clustering-based pair trading strategy. For a fair comparison, our two-pair trading performance will be primarily based on the the stocks under S&P500 from Jan 2021 to Mar 2023. Our metric would be the Sharpe ratio. Our results are as follows:

Pair Trading comparison					
Name	Time period	Asset	technique	Number of pairs	Sharpe ratio
Statistical(ours)	Jan2021-Mar 2023	SP500	correlation, cointegration	102	0.918
K-means(ours)	Jan2021-Mar 2023	SP500	Clustering, correlation, cointegration, mean-reversion, mean-cross	66	1.675
Gatev et al. (2006) [17]	1962-2002	all US stocks	spread	29	0.41
Avellaneda and Lee (2010) [18]	1990-2008	SP100	cointegration, mean reversion	60	0.7
Cakici et al. (2011) [19]	1990-2008	SP500	cointegration, mean reversion	35	0.42
Bonomo and Garcia (2013) [20]	2002-2011	Bovespa index	cointegration	11	0.46

Table 6: Comparison of pair tradings

Our results show that both strategies achieved higher Sharpe ratios compared to the traditional pair trading strategies reported in the literature. Specifically, the statistical pair trading strategy achieved a Sharpe ratio of 0.918, while the K-means clustering-based strategy achieved a Sharpe ratio of 1.675. However, the assets studied in our models are different from the ones studied in the literature. Therefore, further research is needed to validate our findings and explore the effectiveness of these strategies in different market conditions and with different asset classes. But overall, the results are satisfactory and in line with our expectation that machine learning can improve trading performance.

2) Defensive portfolio optimization & RNN-based portfolio optimization

In our study, similar to the other comparison, we developed two different portfolio optimization, including Defensive portfolio optimization & RNN-based portfolio optimization. For a fair comparison, our two portfolio performances will be primarily based on the the stocks under S&P500 from Jan 2021 to Mar 2023. The metric we use are the Sharpe ratio. Our results are as follows:

Portfolio optimization comparison				
Name	Time period	Asset	technique	Sharpe ratio
Defensive(ours)	Jan2021-Mar 2023	SP500	Sharpe Ratio and Return	0.915
LSTM(ours)	Jan2021-Mar 2023	SP500	RNN, Random sampling and Markowitz optimization	1.146
Brinson et al. (1986) [22]	1974-1983	U.S. pension funds	mean-variance optimization	0.27
Fama and French (1993) [23]	1963-1990	US stock	three-factor model	0.6
Black and Litterman (1992) [24]	1972 to 1990	SP500	Black-Litterman model	0.72

Table 7: Comparison of portfolio optimization

Our results show that both strategies achieved higher Sharpe ratios compared to the traditional portfolio optimization strategies reported in the literature. Specifically,

the defensive portfolio optimization achieved a Sharpe ratio of 0.915, while the RNN-based portfolio optimization strategy achieved a Sharpe ratio of 1.146.
However, our models and data are different from the ones studied in the literature, which can affect the results. Therefore, further research is needed to validate our findings and explore the effectiveness of these strategies in different market conditions and with different asset classes. But overall, the results are satisfactory and in line with our expectation that machine learning can improve trading performance.

3) Binary Classification-based momentum trading & RL-based trading

Similar to the other comparison, we developed two different momentum tradings, including Multi-Model-based Binary Classification momentum trading & Reinforcement learning-based trading. For a fair comparison, our two portfolio performances will be primarily based on the the stocks under S&P500 from Jan 2021 to Mar 2023. The metric we use is the Sharpe ratio. Our results are as follows:

Momentum trading comparison				
Name	Time period	Asset	technique	Sharpe ratio
Multi-Classifier (ours)	Jan 2021-Mar 2023	SP500 index	68 Classifier to generate signals	1.182
RL (ours)	Jan 2021-Mar 2023	SP500 index	Reinforcement learning, technical indicators	0.906
Nijmeh and Zurbruegg (2018) [25]	2001-2016	Nordic stock markets	20-day MA and a 200-day MA	0.66
Assefa and Mamuye (2020) [26]	2008-2018	20 US stocks	RSI and MACD indicators	0.72
Nijmeh and Zurbruegg (2018) [27]	2015	HSI stocks	RSI and MACD indicators	1.02

Table 8: Comparison of Momentum trading

Our results show that both strategies achieved higher Sharpe ratios compared to the traditional portfolio optimization strategies reported in the literature. Specifically, the Binary Classification-based achieved a Sharpe ratio of 1.182, while the Reinforcement learning-based strategy achieved a Sharpe ratio of 0.906. However, our models and data are different from the ones studied in the literature, which can affect the results. Therefore, further research is needed to validate our findings and

explore the effectiveness of these strategies in different market conditions and with different asset classes. But overall, the results are satisfactory and in line with our expectation that machine learning can improve trading performance.

Potential Improvement:

1) Potential Improvement of clustering-based pair trading

While the aforementioned trading strategies utilize 66 pairs, it is plausible that not all of them prove to be profitable. To address this, adjustments to the AROD parameter, such as Kalman Filter and other statistical indicators may enhance performance and eliminate pairs with negative returns (Appendix C). Additionally, our trading is based on the logic of "trading in a pair of two". However, it is possible that we can discover more opportunities using pairs of three or above.

2) Potential Improvement of portfolio optimization

a) Try more RNN models

There are more RNN models that can be used in stock market prediction. For example, we can use Convolutional LSTM, hierarchical attention networks, or attention-based Transformer-based models to capture both spatial and temporal patterns in the input data, or to capture temporal patterns in the input data.

b) Utilize reinforcement learning

It can be used to optimize the portfolio allocation based on the current market state and the investor's preferences. This approach enables the portfolio to adapt in real time to changing market conditions, thereby improving portfolio performance and risk management.

3) Potential Improvement of reinforcement learning

There are different more advanced reinforcement learning approaches. For example, deep reinforcement learning uses neural networks to approximate the value or policy functions in RL problems with large state spaces. Hierarchical reinforcement learning decomposes a problem into subtasks and learns a policy for each subtask. Or we can use Multi-agent reinforcement learning (MARL) to extend RL to situations where multiple agents interact in a shared environment.

Conclusion

In conclusion, we have investigated a variety of non-machine learning and machine learning-based trading strategies for financial markets. Our non-machine learning strategies included defensive portfolio allocation and statistical arbitrage trading. On the other hand, our machine learning-based strategies included clustering-based trading, RNN-based trading, reinforcement learning-based momentum trading, and multiple-classier momentum trading.

Our study found that machine learning-based trading strategies outperformed both benchmark market performance and non-machine learning models. We also compared the performance of our strategies with some results from other relevant literature. Among all of the strategies we tested, K-mean pair trading performed the best, achieving a Sharpe ratio of 1.651. This result suggests that K-mean pair trading can be a promising method for generating profitable trading signals in financial markets.

In addition to our research findings, we also developed a real-time algorithmic trading system for live trading using Interactive Brokers. Our system integrated the machine learning-based strategies we tested in this study and allowed us to execute trades based on the signals generated by these strategies. This system demonstrated the feasibility of using machine learning in real-time trading environments and highlighted the potential for developing practical applications based on the findings of our study.

Overall, our research suggests that machine learning can be an effective tool for generating profitable trading signals in financial markets. With the development of real-time algorithmic trading systems, we hope that our research will contribute to the advancement of practical applications of machine learning in the finance industry.

References

[1] O. Grotte, "What percentage of trading is algorithmic? (algo trading market statistics: Growth, trends, and forecasts)," *Quantified Strategies For Traders*, 07-Jul-2022. [Online]. Available: https://www.quantifiedstrategies.com/what-percentage-of-trading-is-algorithmic/

[2] O. Grotte, "Artificial Intelligence Applied to Stock Market Trading: A Review" *IEEE Access*, 11-Jan-2021. [Online]. Available: https://ieeexplore.ieee.org/document/9350582

[3] Aaron Debrincat, "Employing Machine Learning for Pairs Selection"[Online]. Available:https://hudsonthames.org/employing-machine-learning-for-trading-pairs-selection/

[4] A. Puder, S. Markwitz, F. Gudermann and K. Geihs, "AI-based Trading in Open Distributed Environments" *IEEE Access*, 11-Jan-2021. [Online]. Available: https://link.springer.com/chapter/10.1007/978-0-387-34882-7_12

[5] Selva Prabhakaran "Augmented Dickey Fuller Test (ADF Test) – Must Read Guide" [Online]. https://www.machinelearningplus.com/time-series/augmented-dickey-fuller-test/

[6] Mirwais Waisi "Advantages and Disadvantages of AIbased Trading and Investing Versus Traditional Methods" [Online]. Available:https://www.theseus.fi/bitstream/handle/10024/347449/Waisi_Mirwais.pdf

[7] Glory Shah, "An Improved DBSCAN Algorithm for High Dimensional Datasets: An improvement in terms of number of clusters an in general increasing the accuracy of algorithm", 2012, pp. 1-55.

[8] Mihael Ankerst, Markus M. Breunig , Hans-Peter Kriegel, "OPTICS: Ordering Points to Identify the Clustering Structure"[Online]. Available:https://www.researchgate.net/publication/221214752_OPTICS_Ordering_Points_to_Identify_the_Clustering_Structure

[9] Andrew Brim, "Deep Reinforcement Learning Pairs Trading", 2019, pp. 3-7. [Online].Available:https://digitalcommons.usu.edu/cgi/viewcontent.cgi?article=2447&context=gradreports

[10] I.T. Jolliffe, "Principal Component Analysis", 2018, pp.46-102.

[11] Oleksandr Proskurin, "Using Absolute rules of Disqualification To Detect Mean-Reverting Spreads", 2019, pp. 3-8.

[12] Alessio Emanuele Biondo, Dirk Helbing, Alessandro Pluchino, "Are Random Trading Strategies More Successful than Technical Ones?" [Online]. Available:https://www.researchgate.net/publication/235937406_Are_Random_Trading_Strategies_More_Successful_than_Technical_Ones

[13] Evan Gatev, William N. Goetzmann, K. Geert Rouwenhorst, "Pairs Trading: Performance of a Relative-Value Arbitrage Rule", 2016, pp. 67- 126.

[14] Aitor Fiz "PAIRS TRADING: AN EMPIRICAL STUDY" [Online]. Available:https://www.uv.es/bfc/TFM2014/001-014.pdf

[15] Moraes Sarmento, S., & Horta, N. A Machine Learning based Pairs Trading Investment Strategy. 1st ed. 2021

[16] Prof. Daniel P. Palomar. " *Pairs Trading*."
Available:https://palomar.home.ece.ust.hk/MAFS5310_lectures/slides_pairs_trading.pdf
[17] Gatev, E., Goetzmann, W. N., & Rouwenhorst, K. G. (2006). Pairs trading: Performance of a relative-value arbitrage rule. Review of Financial Studies, 19(3), 797-827.
[18] Avellaneda, M., & Lee, J. H. (2010). Statistical arbitrage in the US equities market. Quantitative Finance, 10(7), 761-782.
[19] Cakici, N., Fabozzi, F. J., & Tan, S. (2011). Size, value, and momentum in emerging market stock returns. Journal of Portfolio Management, 37(2), 109-123.
[20] Bonomo, M., & Garcia, R. (2013). The Brazilian term structure of interest rates: theoretical structure and empirical evidence. Revista Brasileira de Economia, 67(3), 251-268.
[21] Markowitz, H. (1952). Portfolio selection. The Journal of Finance, 7(1), 77-91.
[22] Sharpe, W. F. (1964). Capital asset prices: A theory of market equilibrium under conditions of risk. The Journal of Finance, 19(3), 425-442.
[23] Fama, E. F., & French, K. R. (1992). The cross-section of expected stock returns. The Journal of Finance, 47(2), 427-465.
[24] Black, F., & Litterman, R. (1992). Global portfolio optimization. Financial analysts journal, 48(5), 28-43.
[25]Nijmeh, A., & Zurbruegg, R. (2018). A profitability analysis of moving average technical trading rules in the Nordic stock markets. International Journal of Economics, Commerce and Management, 6(7), 1-19.
[26]Assefa, S. M., & Mamuye, F. B. (2020). Combining MACD and RSI indicators for trading strategy in stocks. International Journal of Economics, Commerce and Management, 8(4), 67-84.
[27]Luo, C., & Chen, H. (2017). Designing a trading strategy based on the relative strength index and moving average convergence divergence indicator. Journal of Applied Mathematics, 2017, 1-9.
[28] Team, T. I. (2023, April 4). *Economic cycle: Definition and 4 stages of the business cycle.* Investopedia. Retrieved April 20, 2023, from https://www.investopedia.com/terms/e/economic-cycle.asp#:~:text=An%20economic%20cycle%20is%20the,stage%20of%20the%20economic%20cycle.
[29] Team, T. I. (2023, March 22). *What is stagflation, what causes it, and why is it bad?* Investopedia. Retrieved April 20, 2023, from https://www.investopedia.com/terms/s/stagflation.asp
[30] *10-year Treasury constant maturity minus 2-year Treasury constant maturity*. FRED. (2023, April 19). Retrieved April 20, 2023, from https://fred.stlouisfed.org/series/T10Y2Y
[31] *10-year breakeven inflation rate*. FRED. (2023, April 19). Retrieved April 20, 2023, from https://fred.stlouisfed.org/series/T10YIE
[32] *Market yield on U.S. Treasury securities at 20-year constant maturity, quoted on an investment basis*. FRED. (2023, April 19). Retrieved April 20, 2023, from https://fred.stlouisfed.org/series/DGS20

Appendix A: Meeting Minutes

Minutes of the 1st Project Meeting

Date: May 20, 2022

Time: 10:00 am

Place: Zoom meeting

Present: Pak Hei, Check Hei, Siu Hei, Thomas, Prof. Golin

Absent: None

Recorder: Pak Hei

1. Approval of minutes
   This was the first formal group meeting, so there were no minutes to approve.

2. Report on progress
   - 2.1 All team members have read the instructions of the Final Year Project online and have done research on the topic.
   - 2.2 All team members have done research on computational finance and basic elements of quantitative trading.
   - 2.3 All team members have a fundamental knowledge of how the trading market works.
3. Discussion items
   - 3.1 Professor Golin recommended that we first try to define what asset classes, and machine learning models we should use, due to crawling the data we need for the project.
   - 3.2 Prof. Golin also reminded us that if HFC trading strategies are used, trading speed, network speed, and trading arch structure should be put emphasis, as we may need to compete with other market makers to find arbitrage opportunities.

4. Goals for the upcoming month
   - 4.1 All members will begin to have a brief understanding of machine learning and research different trading strategies that are related to machine learning.
   - 4.2 All members will come up with the asset class programming language(s) we would use
   - 4.3 Thomas will contact the library for the Bloomberg terminal and the CSE department to see how much budget/resources we could get for the server/hardware support.
5. Meeting adjournment and next meeting
   The meeting was adjourned at 11:00 a.m.
   The next meeting will be at 11:00 p.m. on Jul 15th in the Zoom meeting.

Minutes of the 2nd Project Meeting

Date: Jul 10, 2022

Time: 10:00 p.m.

Place: Zoom meeting

Present: Pak Hei, Check Hei, Siu Hei, Thomas

Absent: Prof. Golin

Recorder: Pak Hei

1. Approval of minutes

   The minutes of the last meeting were approved without amendment.

2. Report on progress

   2.1 All members have compared the pros and cons of different programming languages

3. Discussion items

   3.1 Pak Hei suggested using C#/C++ as the primary language to work on the trading system as it is one of the fastest languages and C++ is the language that all members know due to the COMP2012. Also, Python and R languages can be used for statistical and machine learning research.

   3.2 Thomas knows most of the popular languages and has no preference for the programming language, but particularly highlighted that he was proficient in network architecture and database language and design.

   3.3 Check Hei suggested using Python as it is one of the most prevalent general-purpose programming languages used today, and there are already plenty of libraries related to Machine Learning such as sklearn and Pytorch.

   3.4 Siu Hei also has no preference for the choice of the programming language but preferably suggested using Python as the primary language as it is easy to compute, and understand, and its dynamic structure.

4. Brief Conclusion of the meeting:

   4.1 All members agreed to use Python as the primary programming language for the research and ML modeling.

   4.2 All members agreed to only have one to two trading strategies in our final real-time trading due to the complexity of the project.

   4.3 Thomas could be responsible for data crawling and database development.

5. Goals for the coming month

   5.1 All members will come up with one to two trading strategies with the respective machine-learning model.

   5.2 Thomas will contact Prof. Golin to see the support we can get from the CSE department.

6. Meeting adjournment and next meeting

   The meeting was adjourned at 11:00 pm.

   The next meeting will be at 10:00 pm on Aug 14 via Zoom meeting

Minutes of the 3rd Project Meeting

Date: August 14, 2022
Time: 10:00 p.m.
Place: Zoom meeting
Present: Pak Hei, Check Hei, Siu Hei, Thomas
Absent: Prof. Golin
Recorder: Pak Hei

1. Approval of minutes
   The minutes of the last meeting were approved without amendment.

2. Report on progress
   2.1 All members have done the research and proposed at least one trading strategy with the respective references and algorithms.
   2.2 Thomas got the response from the CSE department that only around HKD1000-2000 could be claimed for a group. But if most of the groups do not request budget support from the CSE department, the approximate budget of HKD 3000 could be claimed. Prof. Golin suggested organizing what we need for the project and calculating the cost. Thomas suggested that Cloud services from campus could possibly be borrowed for our project in order to save the cost and time of the development.
   2.3 All members have gathered the elements of the projects and the timeline of each task.
3. Discussion items
   3.1 Pak Hei suggested implementing two models based on pair trading. The first one is the Deep Q- learning model with the deep learning method. The RL framework particularly utilizes a DQN to run a set exchanging technique on cointegrated stock sets. The DQN interatomic with an RL environment by taking the activities too long, brief, or entering no position, on the spread of the stock match. The DQN produces a Q-function, which learns the exchanging parameters to maximize the benefit of the sets exchanging methodology. This sets exchanging methodology will execute based on N exchanging parameters which speak to the spread, with a plausibility of M values or more for each parameter. As the DQN takes activities within the environment and gets remunerated for each activity, it optimizes a Q-function which yields the leading activity for any given state within the space.
   3.2 Check Hei suggested implementing Monte Carlo. The underlying concept is to use randomness to solve problems that might be deterministic in principle. These flows of probability distributions can always be interpreted as the distributions of the random states of a Markov process whose transition probabilities depend on the distributions of the current random states. Check Hei also suggested a non-ML-based approach with Martingale bidding.
   3.3 Siu Hei was also in line with Pak Hei with the idea of pair trading. Siu Hei has tried the pair trading implementation with the non-ML approach to testing the primary risk and loss with the help of a heat map and basic calculation of convergence and divergence using Python.

3.4 Thomas suggested implementing a Neural Network High-Frequency Limit Order Book. The idea is to predict the price of the stock using a limit order book with a feature extractor in 30 mins to train the data and then 10 seconds of test data from the historical order book data in the rolling windows. Trainers such as Random Forest Classifier, Extra Trees Classifier, AdaBoost Classifier, and Gradient Boosting Classifier may be developed. There may be around five folds of model construction and cross-validation. The results will be presented as either 0 or 1 in the metric predictor and then passed to the trader to trade the stock and then calculate the P/L. The Program Efficiency Bound, Efficiency Bandwidth, and Risk control will be evaluated in the process.

3.5 Pak Hei also suggested functions like the Karman filter, risk control mechanism, decision rule, trading systems, unit testing, penetration testing, and UI/UX may also be included in the project.

3.6 Thomas suggested trying to work on different models during the testing stage and choosing the best one/few to implement in real-time trading.

3.7 All members have listed some Python libraries that can be used such as Pytorch, Sklearn, and Tensorflow.

4. Brief Conclusion of the meeting

4.1 All members will start to work on the proposal

4.2 Pair Trading and HFC limit Order book trading strategies will likely be implemented.

4.3 Features such as backtesting, forecast testing, and real-time trading will be included. Other programming languages such as C#/C++ may be used in the real-time trading stage.

5. Goals for the coming month

5.1 All members will work on the Proposal.

5.2 Check Hei and Siu Hei will work on the Introduction of the proposal.

5.3 Thomas will work on the Hardware and Resources required for the proposal.

5.4 Pak Hei will design the workflow work on the work distribution, GANTT chart, meeting minutes, and references of the proposal.

5.5 Thomas will try to ask the library to see if we can use the Bloomberg terminal to get the required data. If the data could be taken from the Bloomberg terminal, Thomas would get the data required as soon as possible, including candle chart data and the limit order book data.

6. Meeting adjournment and next meeting

The meeting was adjourned at 11:00 p.m.

The next meeting will be at 10:00 p.m. on Aug 28 via Zoom.

Minutes of the 4th Project Meeting

Date: 28 Aug 2022
Time: 10:00 p.m.
Place: Zoom meeting
Present: Pak Hei, Check Hei, Siu Hei, Thomas
Absent: Prof. Golin
Recorder: Pak Hei

1. Approval of minutes
   The minutes of the last meeting were approved without amendment.

2. Report on progress
   2.1 The proposal is revised and ready for submission.
   2.2 The workflow and the work distribution are discussed and revised.
   2.3 All team members have sufficient knowledge of trading strategies and ML models.
   2.4 All team members will start the work according to the work distribution.

3. Discussion items
   3.1 The section for the "Details of Machine Learning Model and Trading strategies" should also be included for a clearer explanation.

4. Goals for the coming month
   4.1 The proposal is sent to Prof. Golin and the language tutor for review before submission.
   4.2 Thomas will start to work on the data crawling and finish before 10 September.
   4.3 Check Hei will work on the API connection.
   4.4 Siu Hei will start to work on the non-ML-based trading strategies after the historical bar data is available.
   4.5 Pak Hei will start to work on the ML-based pair trading strategies after the historical bar data is available.

5. Meeting adjournment and next meeting
   The meeting was adjourned at 11:00 p.m...
   The next meeting will be at 10:00 p.m. on Sep 5th via Zoom.

Minutes of the 5th Project Meeting

Date: 15 Oct 2022
Time: 10:00 p.m.
Place: Zoom meeting
Present: Pak Hei, Check Hei, Siu Hei, Thomas
Absent: Prof. Golin
Recorder: Pak Hei

1. Approval of minutes
   The minutes of the last meeting were approved without amendment.

2. Report on progress
   2.1 The monthly report is revised and ready for submission.
   2.2 The workflow and the work distribution are discussed and revised.
   2.3 All team members will start the work according to the work distribution.

3. Discussion items
   3.1 Report the progress from late August to Oct

4. Goals for the coming month
   4.1 The monthly progress is sent to Prof. Golin for review before submission.
   4.2 Thomas will start to work on the system development and will finish by December
   4.3 Check Hei will work on the Multi-layer perceptron binary classification.
   4.4 Siu Hei will start to work on the non-ML-based trading strategies after the historical bar data is available.
   4.5 Pak Hei will start to work on the ML-based pair trading strategies after the historical bar data is available.

5. Meeting adjournment and next meeting
   The meeting was adjourned at 11:00 p.m...

The next meeting will be at 10:00 p.m. on Dec 15th via Zoom.

Minutes of the 6th Project Meeting

Date: 15 Dec 2022

Time: 10:00 p.m.

Place: Zoom meeting

Present: Pak Hei, Check Hei, Siu Hei, Thomas

Absent: Prof. Golin

Recorder: Pak Hei

1. Approval of minutes
   The minutes of the last meeting were approved without amendment.

2. Report on progress
   2.1 The monthly report is revised and ready for submission.
   2.2 Thomas has completed the UI/UX structure.
   2.3 Siu Hei has completed the non-ML trading strategies.
   2.4 Check Hei has completed the Multi-layer perceptron binary classification.
   2.5 Pak Hei has completed the K-means pair trading strategies.

3. Discussion items
   3.1 Report the progress from late Oct to Dec

4. Goals for the coming month
   4.1 The monthly progress is sent to Prof. Golin for review before submission.
   4.2 Thomas will continue to work on the system development.
   4.3 Siu Hei will start to work on the defensive portfolio strategies.
   4.4 Pak Hei will start to work on the LSTM portfolio optimization strategies.
5. Meeting adjournment and next meeting
   The meeting was adjourned at 11:00 p.m...

The next meeting will be at 10:00 p.m. on Feb 5th via Zoom.

Minutes of the 7th Project Meeting

Date: 5 Feb 2023

Time: 10:00 p.m.

Place: Zoom meeting

Present: Pak Hei, Check Hei, Siu Hei, Thomas, Golin

Recorder: Pak Hei

1. Approval of minutes

   The minutes of the last meeting were approved without amendment.

2. Report on progress

   2.1 Thomas has completed the UI/UX structure and the trading system with database, API connection, and MQ.

   2.2 Siu Hei has completed the non-ML trading and defensive portfolio trading strategies.

   2.3 Check Hei has completed the Multi-layer perceptron binary classification.

   2.4 Pak Hei has completed the K-means pair trading and LSTM portfolio optimization strategies.

3. Discussion items

   3.1 Report the progress from late Oct to Jan and clarify the idea to Professor Golin.

4. Goals for the coming month

   4.1 The Progress Report will be finalized and to be sent to Prof. Golin and the language tutor for review before submission.

   4.2 Thomas will continue to work on the system development.

   4.3 Check Hei will continue to work according to the advice of Golin.

   4.4 Siu Hei will continue to work on the defensive portfolio strategies.

   4.5 Pak Hei will start to work on the Deep Q-Learning model.

5. Meeting adjournment and next meeting

   The meeting was adjourned at 11:00 p.m...

The next meeting will be at 10:00 p.m. on Jan 25th via Zoom.

Minutes of the 8th Project Meeting

Date: 12 Feb 2023

Time: 10:00 p.m.

Place: Zoom meeting

Present: Pak Hei, Check Hei, Siu Hei, Thomas, Language tutor Noor

Absent: Prof. Golin

Recorder: Pak Hei

1. Approval of minutes

   The minutes of the last meeting were approved without amendment.

2. Discussion items

   2.1 The language and the proposal structure.

3. Goals for the coming month

   3.1 The Progress Report will be finalized and to be sent to Prof. Golin for review and submission to the CSE system.

   3.2 Everyone will start to write and revise the progress report according to the advice of the language tutor.

   3.3 Thomas will be responsible for the Overview, system structure, and overall structure

   3.4 Siu Hei, Check Hei, and Pak Hei will write the ideas for the trading strategies on the report.

   3.5 Pak Hei will also write the appendix and meetings.

4. Meeting adjournment and next meeting

   The meeting was adjourned at 11:00 p.m...

The next meeting will be at 10:00 p.m. on 15th March via Zoom.

Appendix B: Project Planning

Distribution of Work

Task	Pak Hei	Cheuk Hei	Siu Hei	Thomas
Literature Review	●	●	●	●
Research on Machine Learning Models and Trading Strategies	●	●	●	●
Proposal	●	●	●	●
Hardware Support	○	○	○	●
Server Development	○	○	○	●
Cloud System Design	○	○	○	●
Data Crawling	●	●	●	●
Non-ML pair trading	○	○	●	○
Defensive Portfolio Allocation	○	○	●	○
ML pair trading — Clustering-based Pairs Trading	●	○	○	○
ML Reinforcement Learning	●	○	○	○
ML- Deep Q Learning	●	○	○	○
ML- Markotiwz Portfolio Optimization with LSTM	●	○	○	○
Multiple Model-based Binary Classification	○	●	○	○
Build the database	○	○	○	●
API Connection	○	○	○	●
Develop the trading system (UI/UX)	○	○	○	●
Preparation of Meeting minutes	●	●	●	●
Write the reports	●	●	●	●
Prepare for the presentation	●	●	●	●
Design the project poster	●	●	●	●

● Leader ○ Assistant

GANTT Chart

Task	July	Aug	Sep	Oct	Nov	Dec	Jan	Feb	Mar	Apr
Literature Review										
Research on Machine Learning Models and Trading Strategies										
Server Development										
Cloud System Design										
Data Crawling										
Build the database										
Non-ML pair trading										
ML- Deep Q Learning RL										
ML pair trading — K-means										
ML- Markotiwz Portfolio Optimization with LSTM										
Multiple Model-based Binary Classification										
API Connection										
Develop the trading system										
Test the API connection										
Test the Trading System										
Test the UI/UX of the trading System										
Perform integration testing										
Write the proposal										
Write the progress report										
Write the final report										
Prepare for the presentation										
Design the project poster										

Appendix C: Project Planning

Keywords

Machine-Learning

Referred to as predictive analytics, machine learning is not only inquiry-based learning but also a part of artificial intelligence that leverages data to enhance performance regarding a distinctive set of tasks. Mathematical optimization delivers approaches to analyzing data, exploiting truths in data through Machine learning. Training data is required for models to make an intellectual prediction without the need of coding explicitly. Since conventional algorithms could hardly resolve technical issues in certain areas - medicine, computer vision, face recognition, and finance; ML-based learning is thereby used. In fact, machine learning could even mimic the working of a biological brain with the use of data and neural networks.

Interactive Brokers

Interactive Brokers is regarded as one of the best MAS-regulated brokers. A unique range of financial instruments is offered via their proprietary trading platforms. It is a trading system that is used by professional traders and widely adopted by institutional traders.

- Client Portal: An easy-to-use online trading platform.
- Trader Workstation (TWS): A desktop platform created for traders to develop strategies across different asset classes.
- API and source codes are ready on the official website for implementation.

Choose of Broker

Our algorithm will be mainly developed with Interactive Brokers. It is not only one of the most historical brokers but also an all-pervasive professional trading system that provides all forms of tools to achieve successful trades.

Stationary Testing

1. Augmented Dickey-Fuller Test

The Augmented Dickey-Fuller test (ADF Test) is a common statistical test used to test whether a given time series is stationary or not. The null hypothesis of the test is that the time series can be represented by a unit root and that it is not stationary or has some time-dependent structures. The result can be interpreted by using the p-value from the test[5]:

1. p-value > 0.05: Fail to reject the null hypothesis (H0), the data has a unit root and is non-stationary.
2. p-value <= 0.05: Reject the null hypothesis (H0), the data does not have a unit root a Hurst Exponent, and is stationary.

2. Hurst Exponent

The Hurst exponent can be used to evaluate the stationarity of a time series and the mean-reverting property of the time series.

1. 0 < H < 0.5, indicated time series exhibit mean-reversion property
2. H = 0.5, indicated time series is a geometric random walk
3. 1 > H > 0.5, a persistent trending series. The closer the value is to 1, the stronger the trend.

3. Cointegration

Cointegration is a statistical technique that tests for the presence of a long-run relationship between two or more non-stationary time series. In other words, it measures the extent to which these series move together over time, taking into account their tendency to deviate from their long-term averages. Mathematically, cointegration can be represented as $Yt = \alpha + \beta Xt + \varepsilon t$, where Yt and Xt are non-stationary series, α and β are constants, and εt is a stationary error term. A common application of cointegration is in pairs trading, where two assets with a cointegrated relationship are traded to exploit any deviations from their long-term equilibrium.

4.Mean-reversion

Mean-reversion is a financial concept that refers to the tendency of asset prices or other economic variables to move back towards their long-term average over time. It assumes that deviations from this average are temporary and will eventually be corrected. Mathematically, mean-reversion can be represented as $Xt = \mu + \varepsilon t$ -

β(Yt-μ), where Xt and Yt are the two variables being analyzed, μ is the long-term average, εt is a random shock, and β is the speed of reversion to the mean. Mean-reversion has practical applications in trading strategies, such as buying assets that have recently underperformed and shorting those that have recently outperformed, in the hope of profiting from their return to the mean.

More About Models and Theory

Non-ML approach - Monte Carlo methods

The Monte Carlo method is not only a broad class of algorithms that obtain numerical results by relying on repeated random sampling but also a lightweight approach. As randomness could solve problems that might be deterministic in principle, physical and mathematical problems are using this method very often. Optimization, numerical integration, and generating draws from a probability distribution are three of these areas that depend on Monte Carlo methods the most.

Pair Trade - Pair Selection

Cointegration tests identify scenarios where two or more non-stationary time series are integrated in a way that they cannot deviate from equilibrium in the long term. The tests can identify the degree of sensitivity of two variables to the same average price over a specified period.

The cointegration approach entails selecting pairs for which the two constituents are cointegrated. If two price series, Y(t) and X(t) are found to be cointegrated, then by definition the series resulting from the linear combination $S(t) = Y(t) - \beta X(t)$, where β is the cointegration factor, must be stationary. Naturally, defining the spread series this way is particularly convenient, since under these conditions it is expected to be mean-reverting.

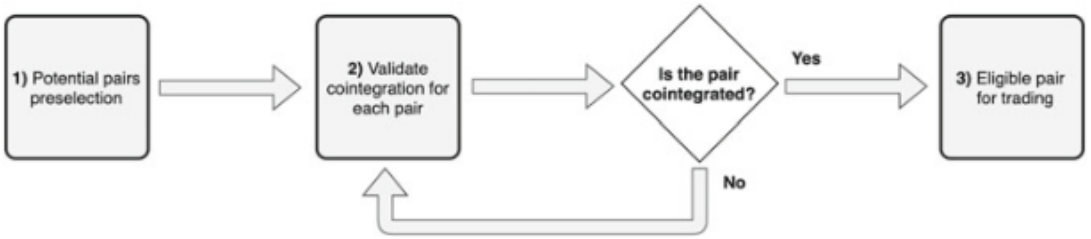


Figure 41: Pair selection using the cointegration approach

Total profit: profit of each trade × number of trades

1. profit of each trade is s_0
2. the number of trades is related to the zero crossings, which can be analyzed theoretically as well as empirically

Pair Trade - Other Approaches for pair selection

Ranking the pairs based on the spread's Hurst exponent [6]. Researchers obtain superior results when compared with the classical approaches based on cointegration and correlation, especially with a large number of trade securities in the portfolio.

Pair Trade - Algorithm

This model can be described more formally as follows [14].

i. Calculate the spread ($S_t = Y_t - X_t$) mean, μ_s, and standard deviation, σ_s during the pair's formation period.

ii. Define the model thresholds: the threshold that triggers a long position, α_L, the threshold that triggers a short position, α_S, and the exit threshold α_{exit} which defines the level at which a position should be exited.

iii. Monitor the evolution of the spread, S_t, and control if any threshold is crossed.

iv. In case α_L is crossed, go long the spread by buying Y and selling X . If α_S is triggered, short the spread by selling Y and buying X . Exit position when α_{exit} is triggered and a position is being held.

Pair Trade - Optimization

1. Parametric approach

This approach is based on the assumption that the spread follows a standard Normal distribution.

The probability the spread deviates above the mean by s_o or more is $1 - \Phi(s_o)$.

where $\Phi(.)$ is the c.d.f of the standard Normal distribution.

For a path with T days, the number of tradable events is $T(1 - \Phi(s_o))$.

For each trade, the profit is s_o and then the total profit is $s_o T(1 - \Phi(s_o))$.

The optimal threshold refers to the threshold that gives the maximum expected profit.

$$*s_o = \arg\max_{s_o} \{s_o\, T(1 - \Phi(s_o))\}.$$

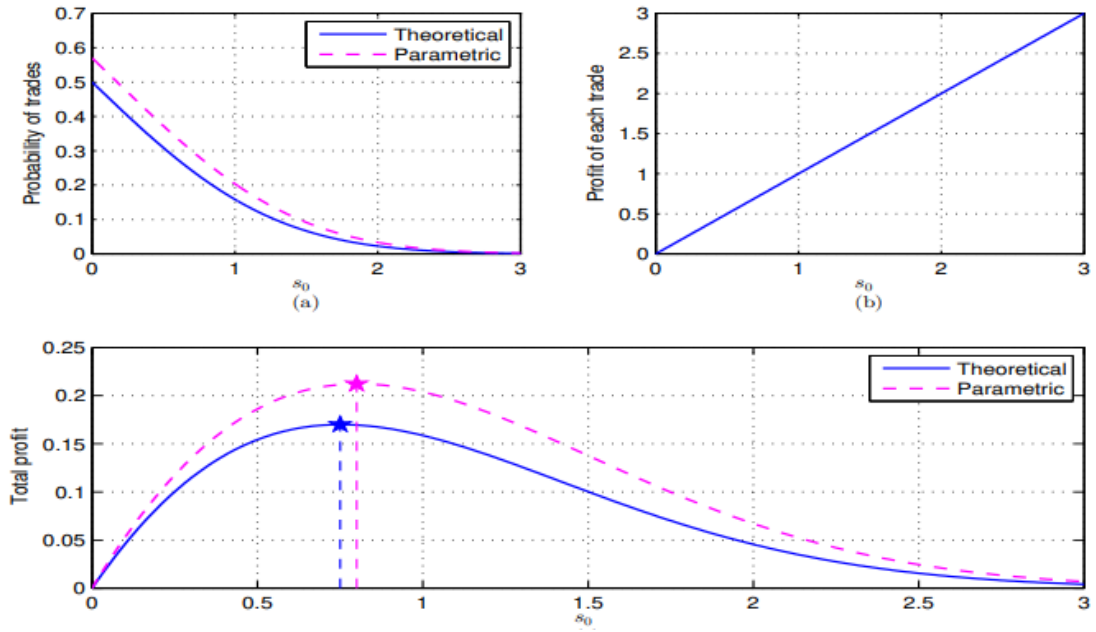


Figure 42: Parametric approach graphic representation

2. Non-parametric approach

Suppose the observed sample-path has length T: $z_1, z_2, \ldots, z_t$. We consider J threshold values as so $\in \{s_{o1}, s_{o2}, \ldots, s_{oj}\}$ and the empirical trading frequency for the threshold s_{oj} is

$$\bar{f}_j = \frac{\sum_{t=1}^{T} \mathbf{1}_{\{z_t > s_{0j}\}}}{T}.$$

The empirical values fj may not be smoothed enough and the resulting profit function may not be accurate enough.

Smooth the trading frequency function by regularization:

$$\underset{\mathbf{f}}{\text{minimize}} \quad \sum_{j=1}^{J} (\bar{f}_j - f_j)^2 + \lambda \sum_{j=1}^{J-1} (f_j - f_{j+1})^2$$

The problem can be rewritten as

$$\underset{\mathbf{f}}{\text{minimize}} \quad \|\bar{\mathbf{f}} - \mathbf{f}\|_2^2 + \lambda \|\mathbf{D}\mathbf{f}\|_2^2$$

The derivative of the objective w.r.t. f to zero yields the optimal solution

$$\mathbf{f}^\star = (\mathbf{I} + \lambda \mathbf{D}^T \mathbf{D})^{-1} \bar{\mathbf{f}}.$$

The optimal threshold is the one that maximizes the total profit.

$$s_0^\star = \underset{s_{0j}\in\{s_{01},s_{02},\ldots,s_{0J}\}}{\arg\max} \{s_{0j} f_j\}.$$

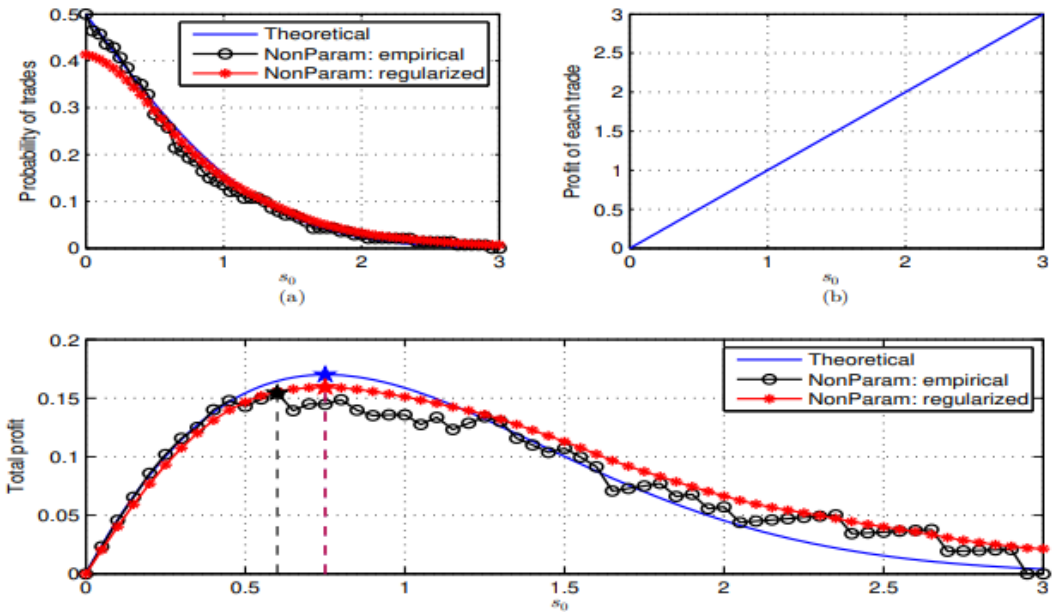


Figure 43: optimal threshold of the Non-Parametric approach

Deep Q-Neural Network Model

This Reinforcement Learning system specifically utilizes a DQN to run a pairs trading strategy on cointegrated stock pairs. The DQN interacts with an RL environment by taking the actions to long, short, or enter no position, on the spread of the stock pair. The DQN produces a Q-function, which learns the trading parameters to maximize the profit of the pair's trading strategy. This pairs trading strategy will execute based on N trading parameters which represent the spread, with a possibility of M values or more for each parameter. As the DQN takes actions in the environment and receives rewards for each action, it optimizes a Q-function which outputs the best action for any given state in the space [9].

Dimension Reduction — Principal component analysis (PCA)

PCA is a statistical procedure that uses an orthogonal transformation to convert a set of perceptions of conceivably connected factors into a set of directly uncorrelated factors, the foremost components[10]. The transformation is characterized such that the primary central component accounts for as much changeability within the data as possible. Each succeeding component, in turn, has the most noteworthy change

conceivable beneath the limitation that it must be orthogonal to the going-before components.

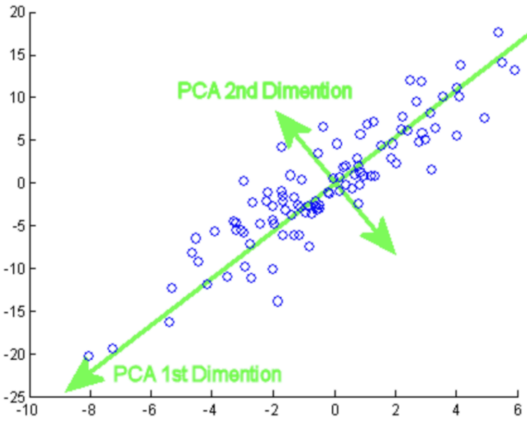


Figure 44: PCA graphical representation

Absolute Rules of Disqualification (ARODs)

After producing the clusters of resources, it is still essential to characterize a set of conditions for selecting the sets to trade. Distance, cointegration, and correlation are the most common approaches. The cointegration approach was usually preferred. Then the Engle-Granger test could be applied. Only the Cointegrated pairs will be passed to the next steps. Then, due to the pair spread, the validation step is implemented to provide more confidence based on the mean-reversion nature. Hurst exponent could be applied to measure the amount of memory in a particular time series. If the Hust Coefficient is not greater than half, it indicates that the process is leaning toward the mean reversion and then we can pass to the next step. With different ARODs applied to the strategies, some uncertain parts can be sorted out and only leave the pairs with high expectations of the real trading environment [11].