Problem Chosen	2022 MCM/ICM Summary Sheet	Team Control Number
C		2210085

Quantitative Trading Strategy Based on Ensemble Learning and Mathematical Programming

Summary

Predicting future data from historical asset price data is the first step to making a good trading decision. Then the optimal decision is made based on the predicted data using appropriate planning equations. The study of this problem will allow investors to make optimal trading strategies and maximize their investment returns using only historical data.

For problem one, it is a prediction-decision problem. Since we can only use historical asset price data, making accurate future price forecasts is crucial. We combine two prediction models that dynamically utilize historical price data up to each trading day to estimate the trend over the coming period. When the prediction period is short, the integrated learner increases the weight of the ARIMA model to capture the long-term trend. When the prediction period lengthens, the integrated learner increases the weight of the LSTM model to predict longer-term data changes better. We use a multi-objective mathematical programming equation as a decision model that integrates investor returns and risk preferences to maximize returns, minimize risk, and balance positions. We took the high transaction fees of bitcoin and gold into consideration. The future data gives the optimal solution for different trading frequencies. The final total asset value on 9/10/2021 with a 7-day trading period is $4012931.83.

For problem 2, since we used the prediction model in problem 1, it necessarily has the problem of fitting bias. In the mathematical programming model, the solution obtained by the model itself has reached the local optimum. So we analyze that only the accuracy of the forecasting model needs to be accurate enough to prove that our strategy is the best. We draw the loss curve of the prediction model and use the two statistics indicators MSE, R^2, which reflect the goodness of fit of the model, to represent the difference between the predicted value and the actual value.

For Problem 3, the transaction cost rate is an essential part of the mathematical planning model of Problem 1. To analyze the sensitivity of this investment strategy regarding transaction costs, we fit the model several times using different transaction cost rates and finally obtain a line graph of total return versus transaction cost rate to reflect its sensitivity.

Finally, combining the prediction model and the decision model, we give the trading records with a trading cycle of 7 days, as shown later. The model we designed can output accurate forecasting data on the one hand, and flexibly change the value-at-risk according to the different investment styles of different investors on the other hand. Thus, we can achieve a balanced position under different scenarios and achieve a balance between risk and return.

Key words: LSTM, ARIMA, Ensemble Learning, Multi-objective Mathematical Programming

Content

1 Introduction 2
1.1 Background 2
1.2 Restatement of the Problem 2
2 Problem Analysis 3
3 The Model Assumption 4
4 Symbol Description 4
5 Model Establishment and Solutions 5
5.1 Data curation and analysis 5
5.2 Trading model establishment 6
5.2.1 Prediction Model(LTSM-ARIMA) 6
5.2.2 Multi-objective programming decisionmaking 10
5.3 Model Accuracy Verification 13
5.4 Sensitivity Analysis 15
5.5 Trading Memorandum 16
6 Strength and Weakness 18
6.1 Strengths 18
6.2 Weaknesses 18
7. Reference 19
Appendix 20

1 Introduction

1.1 Background

Quantitative finance is a new form of finance resulting from the combination of the diversification of financial products, the multi-layered financial product trading market, and the programmatic trading methods of financial products. Its history is not long. Since the rapid development of hedge fund markets in the United States and Europe in the 1990s, various quantitative finance methods have emerged one after another. The premise of quantitative finance is the diversification and derivation of financial products. It is being said that the US and European financial markets carried out financial innovations in the 1980s, forming a spot market with stocks, fixed income bonds, and foreign exchange as the main targets; and then derived stocks, bonds, and foreign exchange through the price fluctuations of the spot. Futures market; after further development, there will be a bullish or bearish options market based on stocks, bonds, and foreign exchange and different types of stock index futures and options trading markets, which provide the premise and foundation for the development of quantitative finance. The development of computer technology has made high-frequency trading, program trading, and algorithmic trading emerge in an endless stream. Various hedge funds and speculators use microsecond speed differences and different transaction price differences to carry out cross-market arbitrage, cross-product arbitrage, and cross-time. Arbitrage constitutes the main content of quantitative financial transactions in modern financial markets. The choice of model is essential in quantitative finance. Choosing a suitable model to make investment decisions using existing data is a problem we need to solve.

1.2 Restatement of the Problem

According to our analysis, this question is a quantitative finance topic with available price data for gold and bitcoin from 9/11/2016 to 9/10/2021. Based on the above background and attached material, we need to solve the following problems.

- develop a model that predicts future price movements based only on the price data for the day and the day before, and then gives the best daily trading strategy. Using the designed model and strategy, calculate how much the initial $1,000 investment will eventually be worth on September 10, 2021
- provide evidence that the prediction-decision model is optimal
- analyze the sensitivity of the investment strategy concerning transaction costs and determine how transaction costs affect the strategy and results
- describe the model, strategy, and results to the traders through a monument.

2 Problem Analysis

2.1 Analysis of Problem 1

According to the relevant data of question 1, we know the price data of gold and bitcoin in the past five years. This question requires us to give the best daily trading strategy based only on the price data of the day。 And then use the designed model and strategy to give the final value of the asset held on September 10, 2021. We analyze the nature of this problem and think it is a prediction-decision problem. Combined with the subject background and known data, we believe that the use of neural network prediction and a multi-objective mathematical programming portfolio investment model is the most suitable. However, it should note that this question requires us to stand in the perspective of the time, and we can only use historical data to make trading decisions every day. Since five years of global data cannot use, making reasonable predictions about future asset values is also a problem we need to solve.

In response to the above problems, our idea is: first, for the problem of some missing values in the data, use the closing price of the previous day for the missing values to interpolate; then we preprocess the data, and then use LSTM to build a prediction model, through the data before each trading day. The historical data predicts the underlying price for the next several days, and obtains the predicted value trend of gold and Bitcoin from 2016 to 2021; finally, a multi-objective mathematical programming model is constructed to consider long-term returns comprehensively, short-term returns, and value at risk to find the optimal investment plan.

2.2 Analysis of Problem 2

According to the results of question 1, we know the predicted price data and daily trading strategies of gold and bitcoin in the past five years. The problem we need to solve in this question is: to provide evidence that the model gives the best strategy. Since we used a predictive model in question 1, there must be a problem of fitting bias. Therefore, we analyze that, as long as the accuracy of the prediction model is accurate enough, it proves that our strategy is the best. For the mathematical programming model, the solution obtained by itself has reached the local optimum.

In response to the above problems, our idea is: first, make a loss map of the prediction model, and use the loss map to reflect the quality of the prediction model; then we use four statistics that reflect the goodness of fit of the model to represent the predicted value and the actual value. Gap: Finally, we get MSE, R2, a total of two statistics to represent the accuracy of the prediction model.

2.3 Analysis of Problem 3

According to the results of question 1, we know the price forecast data and daily trading strategies of gold and bitcoin in the past five years. In designing mathematical programming models, transaction expense rates are also a large part of our model. The problem we need to solve in this question is: to analyze the sensitivity of the investment strategy to transaction costs and to judge how transaction costs affect the strategy and results accordingly.

In response to the above problems, our idea is to perform multiple model fittings with different transaction fee rates and obtain a scatter diagram of total revenue and transaction fee rate to reflect its sensitivity.

3 The Model Assumption

1. Assume that gold and bitcoin forecasts are only affected by price factors.
2. Assume that the gold trading market is closed only on weekends, and the previous day's closing price is used for price data on legal holidays.
3. Assume that the gold and bitcoin trading markets are perfect markets.
4. Assume that gold trading and bitcoin trading only consider price factors.
5. The forecast for gold and bitcoin is based on price data from September 11, 2016, to September 10, 2021, only.
6. When the amount of data is large enough, and it assumes that the prices of gold and bitcoin obey normal distribution.
7. Assume there are no additional circumstances such as economic crisis, taxation, etc.
8. Assume that when you trade on day N, you know the price of gold/bitcoin up to day N. Then buy at this price, and calculate the return from day N+1. The gold and bitcoin transaction rates settles on the same day.
9. Assume that cash possession demand deposit interest rate is 0%.

4 Symbol Description

Symbol	Definition
W_t	Initial daily asset value
R_{it}	Real asset price
x_{it}	Investment proportion
P_{it}	Predicted asset price
r_{it}	Rate of tomorrow return
L_{iT}	Rate of return after T days
α_i	Transaction fee rate
ϖ_t	The weights of the two prediction models
T	Transaction cycles
f_t	Forget gate
w_f	The weight term about forgetting
b_f	Bias term
$\widetilde{C}_t$	The candidate information for memory cell
h_t	The output of the hidden layer

5 Model establishment and solution

5.1 Data curation and analysis

According to the data and related information given by the appendix, we found missing values in the price data of gold in the past five years. We used the closing price of the day before the missing value to interpolate, as shown in Table 1.

Table 1 Disposal of Empty Values

Date	Price
2016-12-23	1131.35
2016-12-30	1145.9
2017-12-22	1264.55
2017-12-29	1291
2018-12-24	1258.15
2018-12-31	1279
2019-12-24	1482.1
2019-12-31	1514.75
2020-12-24	1875
2020-12-31	1887.6

For some important variables, we compute them at first:

Initial daily asset value：

$$W_t = \sum_{i=1}^{m} R_{it} \bullet x_{i,t-1} \bullet W_{t-1}, i = 1,2,3, m = 3 \tag{1}$$

Rate of tomorrow return:

$$r_{it} = \frac{P_{it} - P_{i,t-1}}{P_{i,t-1}}, i = 1,2,3 \tag{2}$$

Rate of return after T days:

$$L_{iT} = \frac{P_{iT} - P_{i,t-1}}{P_{i,t-1}}, i = 1,2,3 \tag{3}$$

Connections between T and t:

$$T = t + n, n = 2,6,13,29 \tag{4}$$

Different assets’ transaction fee rate:

$$\begin{cases} \alpha_1 = 0, cash \\ \alpha_2 = 1\%, gold \\ \alpha_3 = 2\%, BTC \end{cases} \tag{5}$$

5.2 Trading model establishment

5.2.1 Prediction Model (LSTM-ARIMA)

According to the requirements of the topic, we make decisions based on historical data, so we think it is necessary to use a predictive model to estimate the future prices of gold and Bitcoin based on historical data.

LSTM is a recurrent neural network (RNN) suitable for extracting temporal features from time series to learn long-term time series dependencies. We use a long short-term memory network (LSTM) model to predict prices in this problem.

1. Preprocessing the experimental data

The missing gold price data for non-trading days are filled with the previous trading day's price. Different prices have different dimensions and dimension units. In order to eliminate the influence of dimensions between prices, we carry out data standardization.

2. Establishing the LSTM neural network

A recurrent layer is constructed after the convolutional layer. In view of the deficiency that RNN cannot solve the long-term dependency problem, Long-Short Term Memory network is chosen as the recurrent layer in this paper. LSTM is characterized by replacing ordinarily hidden nodes with memory modules to ensure that the gradient will not disappear or explode after lots of time steps and can maintain long-term information.

Following equations (6) - (11) describe operations of three control gates in the LSTM unit, where σ is the sigmoid function. f_t in Equation. (6) is the forget gate, which determines what information the model will ignore, w_f is the weight term about forgetting, and b_f is a bias term. Equation.(7) and Equation.(8) are for the input gate, which updates the candidate information for memory cell $\widetilde{C_t}$. Activation function *tanh* is used to help regulate the values flowing through the network. The *tanh* function always limits the value between -1 and 1. Equation.(9) and Equation.(10) are for the output gate, o_t is obtained from the output gate and h_t is the output of the hidden layer. Equation.(11), the cell state C_t has been updated. Here o_t still uses a sigmoid function to indicate what to output, and C_t is multiplied by o_t after scaling by *tanh*. This is the output of this timestep. [1]

$$f_t=\sigma(W_f \cdot [h_{t-1},x_t]+b_f) \tag{6}$$

$$i_t=\sigma(W_i \cdot [h_{t-1},x_t]+b_i) \tag{7}$$

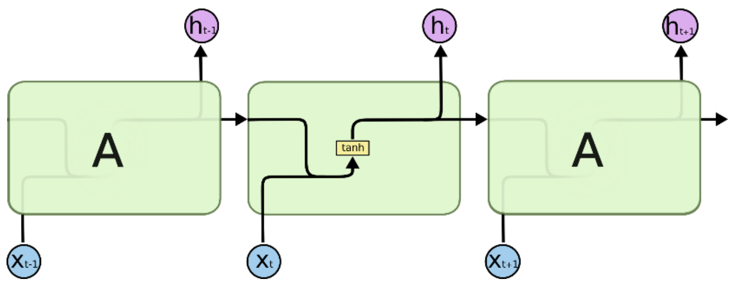
$$\widetilde{C_t}=tanh(W_c \cdot [h_{t-1},x_t]+b_c) \tag{8}$$

$$h_t=o_t*\tanh(C_t) \tag{9}$$

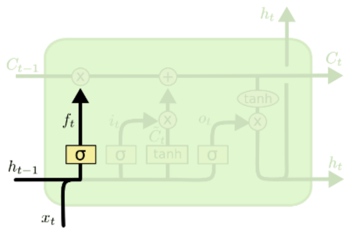
$$o_t=\sigma(W_o \cdot [h_{t-1},x_t]+b_o) \tag{10}$$

$$C_t=f_t*C_{t-1}+i_t*C_t \tag{11}$$

The specific flow of the LSTM application i in the following diagram

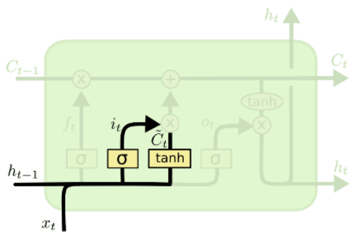


Schematic diagram of the complete forecasting process: X_t represents the input price data for each day h_t represents the output forecast price data for each subsequent day.



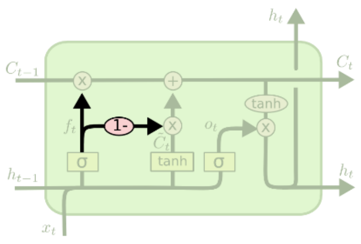
$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \ + \ b_f\right)$$

Inputs include: the day's price data predicted from the previous day h_{t-1} and the actual price data of the day X_t. And decide to discard price information that deviates from the actual through σ.

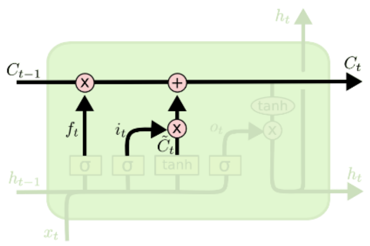


$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \ + \ b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \ + \ b_C)$$

Determine the updated next day price feeds $\widetilde{C_t}$.

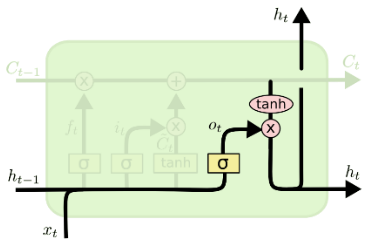


$$C_t = f_t * C_{t-1} + (1 - f_t) * \tilde{C}_t$$



$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Update the overall algorithmic learning memory of the price trend condition C_t and pass it to the next day's forecast decision



$$o_t = \sigma\left(W_o \left[h_{t-1}, x_t\right] + b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

Output forecast information h_t and pass it on to the next day's price data forecast.

3. Training and optimizing LSTM Prediction Models

The data used in the LSTM neural network should be the time-series features of the extracted time-series data. [2] Set the rolling window length to 3 days. The training set uses daily gold settlement prices for 182 days from September 11, 2016, to March 11, 2017, and daily bitcoin settlement prices, from September 1, 2016, to September 11, 2017, in total 364 days of data. An iterative prediction is performed by adding actual price data every day. Then use the data given in the question until September 10, 2021, for comparison and verification.

We also set up a Keras-based two-layer LSTM network with 50 and 100 neurons, respectively, and a drop layer after each layer to do the regularization for the data, which effectively prevents overfitting. The specific code will be shown in Appendix 1

4. Final predicted results

The price forecast charts for gold and bitcoin from 2016-9-11 to 2021-9-10 are as follows fig1,fig2:

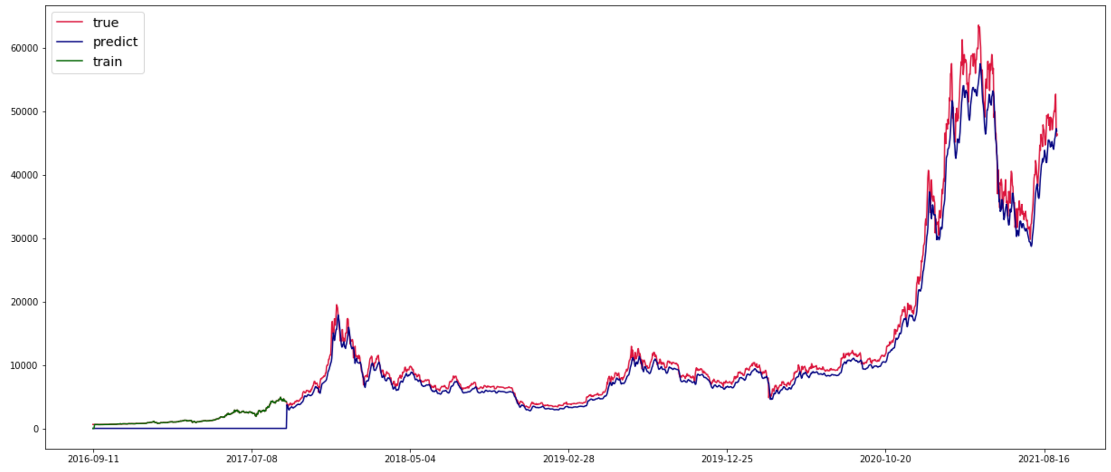


Fig1. Bitcoin Price Forecast Graph Using LSTM

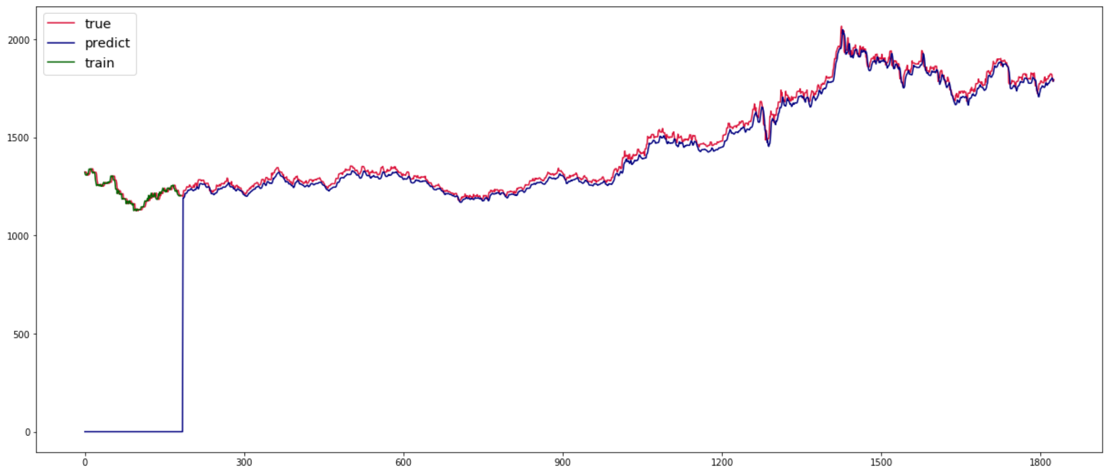


Fig2. Gold Price Forecast Graph Using LSTM

5. Ensemble Learning

LSTM has less learning and fewer number of iterations when the amount of data is small, and the prediction accuracy is not enough. Therefore, we use the ARIMA model for forecasting the price data of the first 182 days or 364 days. ARIMA model performs well in forecasting few numbers of small-scale data with the following mathematical equation:

$$y_t' = \alpha_0 + \sum_{i=1}^{p} \alpha_i y_{t-i}' + \varepsilon_i + \sum_{i=1}^{q} \beta_i \varepsilon_{i-1} \quad \text{while} \quad y_t' = \Delta^d y_t = (1-L)^d y_i \tag{12}$$

It can also be written as

$$(1-\sum_{i=1}^{p} \alpha_i L^i)(1-L)^d y_i = \alpha_0 + (1+\sum_{i=1}^{q} \beta_i L^i)\varepsilon_i \tag{13}$$

$(1-\sum_{i=1}^{p} \alpha_i L^i)$ is AR(p) (autoregressive) model, which can be applied to predict economic phenomena related to its own previous period. [3]

$(1+\sum_{i=1}^{q} \beta_i L^i)\varepsilon_i$ is MA(q) (moving average) model. It describes the time series process y as a linear combination of a series of uncorrelated random variables.

The graph of the prediction results is obtained as follows, fig3,fig4

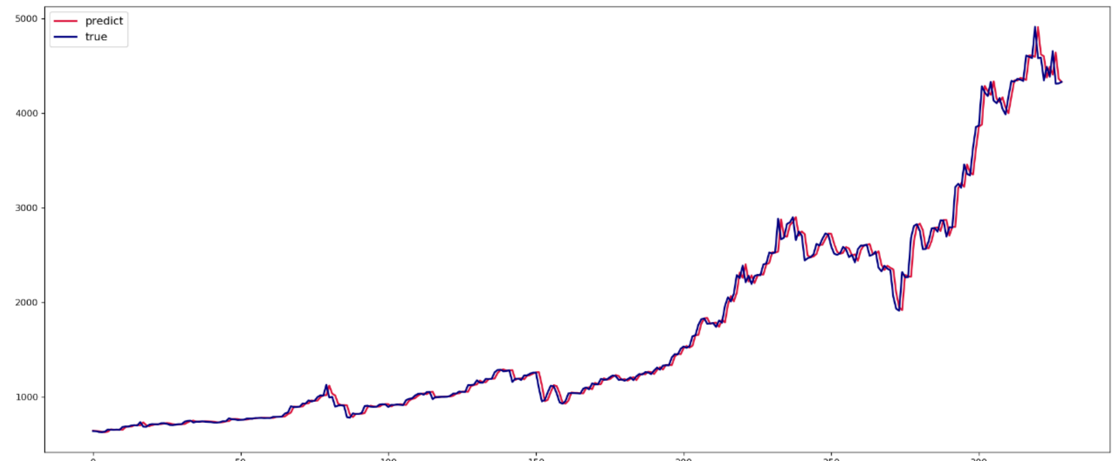


Fig3. Bitcoin Price Forecast Graph Using ARIMA

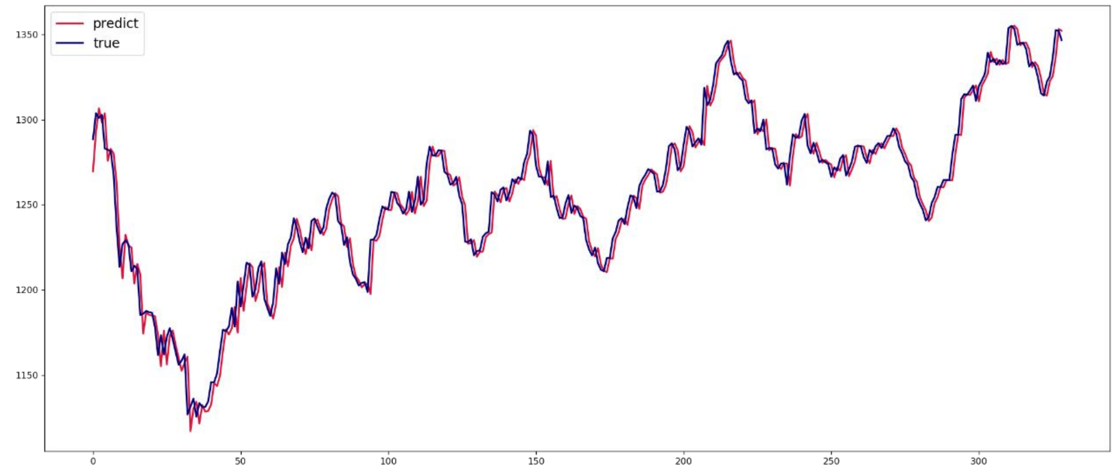


Fig4. Gold Price Forecast Graph Using ARIMA

Lastly, we used the idea of integrated learning to combine the two forecasting models to obtain the following equation.

$$Predict\ Prices = \varpi_t LSTM + (1-\varpi_t) ARIMA \tag{14}$$

In which ϖ_t represents the weights of the two prediction models and varies with time.

To summarize, we used the above equations to predict gold and bitcoin price fluctuations using data from the previous day, from a historical perspective, and without using five years' global data.

5.2.2 Multi-objective programming decision-making

According to the title's known information and prediction model, we can know the price prediction data and actual price data of gold and bitcoin in the past five years. Based on the above conditions, we can establish a linear programming model：

1. Identify decision variables

In this question, the investment ratio x_{it} is the decision variable of the multi-objective mathematical programming model

2. Determine the objective function

In this question, we need to solve the optimal investment decision problem. We comprehensively consider the three factors: long-term income, short-term income, and risk. We use tomorrow returns represent short-term returns rev_{it} :

$$rev_{it} = \sum_{i=1}^{m} W_t \bullet x_{it} \bullet r_{it}, i = 1,2,3, m = 3 \tag{15}$$

Use tomorrow costs represent short-term cost $\cos t_{it}$:

$$\sum_{i=1}^{m} \left| W_t \bullet x_{it} - W_{t-1} \bullet x_{t-1} \right| \bullet \alpha_i, i = 1,2,3, m = 3 \tag{16}$$

Use returns after T days to represent the long-term returns rev_{iT} :

$$rev_{iT} = \sum_{i=1}^{m} W_T \bullet x_{iT} \bullet L_{iT}, i = 1,2,3, m = 3 \tag{17}$$

Use cost after T days to represent the long-term cost $\cos t_{iT}$:

$$\sum_{i=1}^{m} \left| W_T \bullet x_{iT} - W_{t-1} x_{i,t-1} \right| \bullet \alpha_i, i = 1,2,3, m = 3 \tag{18}$$

Then we use VaR as a measurement of risk:

$$\sum_{i=1}^{m} x_i \bullet W_t \bullet (Z \bullet \sigma_{it} + \mu_{it}), i = 1,2,3, m = 3 \tag{19}$$

So the objective function is as follows:

$$\begin{cases} \max \sum_{i=1}^{m} rev_{it} - \cos t_{it} \\ \max \sum_{i=1}^{m} rev_{iT} - \cos t_{iT}, i = 1,2,3, m = 3 \\ \min VaR \end{cases} \tag{20}$$

3. Determine Constraints

Constraint 1: Gold does not trade on weekends. The holdings and prices of gold on weekends should be the same as at the close on Friday:

$$W_t \bullet x_{2,t} = W_{t-1} x_{2,t-1}, t = 6k, 7k \tag{21}$$

Constraint 2: The sum of the investment ratio of each asset does not exceed 100%:

$$\sum_{i=1}^{m} x_{it} = 1, i = 1,2,3, m = 3 \tag{22}$$

Constraint 3: The investment ratio of each asset is not less than 0:

$$x_{it} \geq 0, i = 1, 2, 3 \tag{23}$$

In order to solve this multi-objective programming equation, we know from the question that long-term returns are the most important and value-at-risk is the least important. Therefore, the equations are assigned weights of 0.5, 0.3, and 0.2, respectively. In summary, the final objective function is determined as:

There is stochasticity and lag in the forecast, with trading on a one-day basis introducing higher uncertainty in our portfolio returns. So we selected 3-day, 7-day, 14-day, and 30-day trading periods as examples and obtained the total asset value curve as follows, fig5.

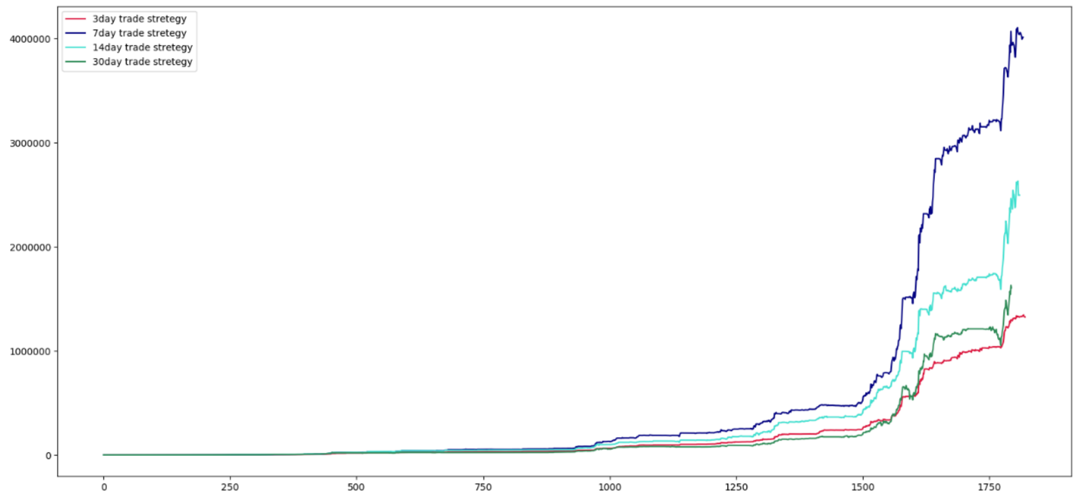


Fig5 Long-term Return Curves Using Multi-objective Programming

The 7-day trading strategy has the best-combined return performance with a fixed risk appetite. Thus, we have chosen it as a case and presented a sample of its portfolio investment position allocation and accumulated returns for some dates in the entire model as follows, table 2

Table 2 7-Day Trading Strategy Results (Sample)

Date	USD_Position	GOLD_Position	BTC_Position	Accumulated returns($)
2016-9-24	89.86%	0.00%	10.14%	999.56
2016-12-11	1.00%	0.00%	99.00%	1205.47
2017-2-3	94.51%	5.49%	0.00%	1653.97
2017-3-11	79.92%	20.08%	0.00%	1817.31
2017-5-13	59.06%	11.03%	29.91%	2724.88
2017-7-14	19.48%	61.46%	19.06%	3829.87
2018-4-9	19.03%	20.42%	60.56%	32141.86
2018-10-12	19.07%	50.98%	29.95%	54087.48
2018-12-24	14.98%	85.01%	0.01%	57817.44
2019-5-1	19.11%	10.00%	70.90%	82882.70
2019-7-14	59.69%	30.29%	10.02%	164308.10
2020-5-12	19.34%	50.31%	30.34%	395480.98
2020-9-29	63.86%	11.85%	24.29%	477187.44
2021-4-5	50.22%	29.38%	20.39%	2941902.35
2021-8-13	60.84%	18.50%	20.65%	3962751.25

We obtain a graph of the portfolio asset position allocation results under different strategies, as follows fig6.

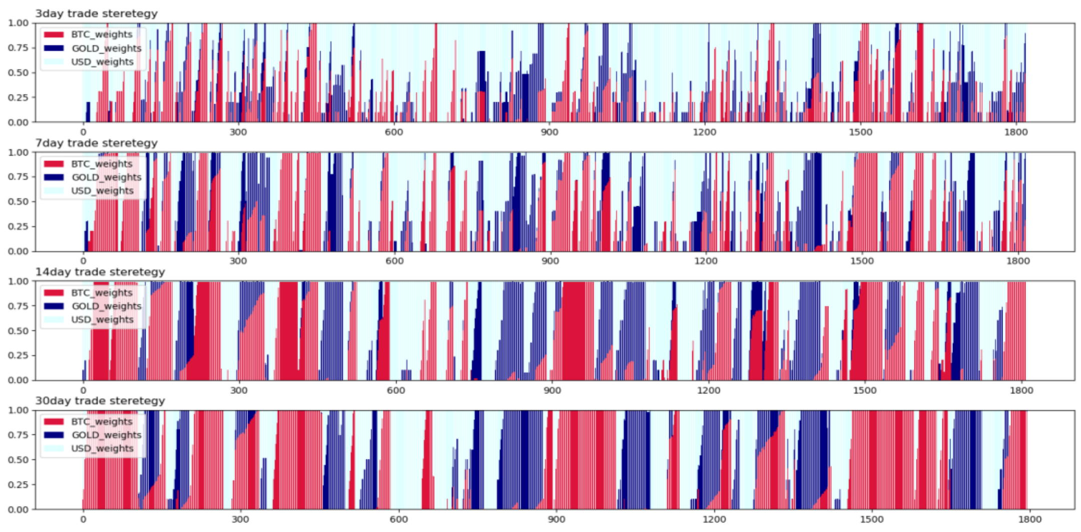


Fig 6 Portfolio Asset Position Allocation Results

Eventually the final total asset value on 9/10/2021 with a 7-day trading period is $4012931.83.

5.3 Model Accuracy Verification

We will provide evidence that our model gives the best strategy. Since we use a predictive model in problem 1, this necessarily has the problem of fit bias. As for the mathematical programming model, the solution obtained by the model itself has reached the local optimum. Therefore, we analyze that only the prediction model needs to be accurate enough to prove that our strategy is the best.

First, we use the loss function to reflect the performance of the LSTM prediction model. The loss function is calculated to adjust the model parameters and reduce the optimization error until the loss function value decreases to the target level. [4] We are graphing the loss curve (fig 7) to evidence that the LSTM neural network in problem one was functioning well, with a good declining trend and no random fluctuation trend. The LSTM achieved a good-fit, and the model performed well.

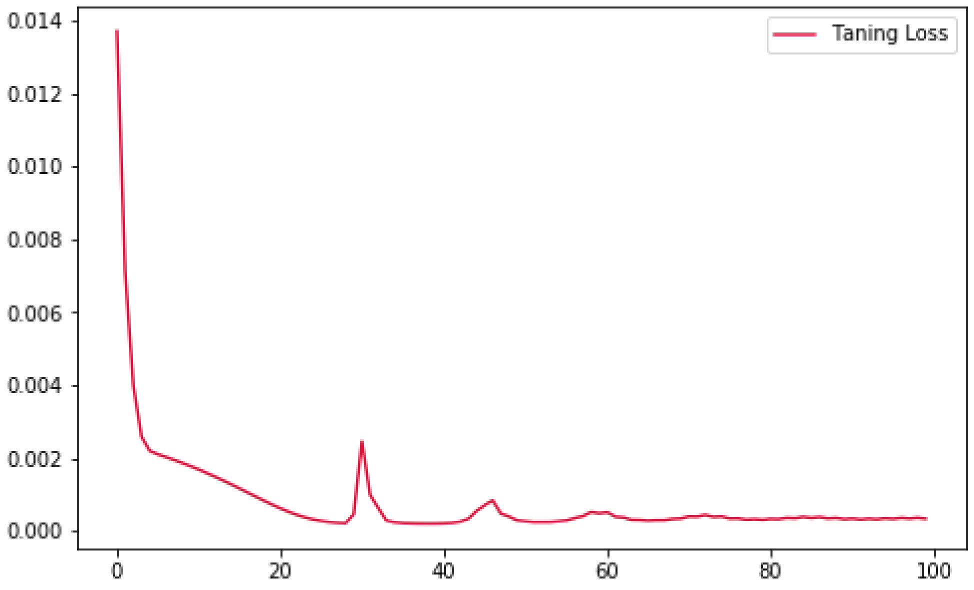


Fig7 Loss Curve From LSTM Model

In the ARIMA model, we plotted the autocorrelation coefficient (ACF) and partial autocorrelation coefficient (PACF) with truncated tails falling within the 95% [5]confidence interval as follows, fig 8.

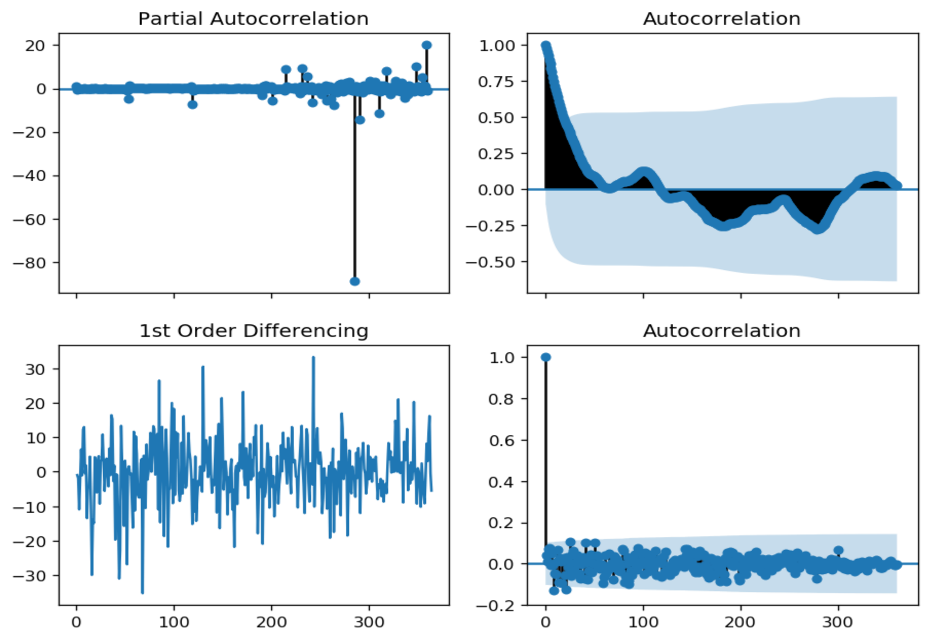


Fig 8 ACF and PACF From ARIMA Model

Then we test the model residuals teste. The residuals of our ARIMA model have a normal distribution with a mean of 0 and a constant variance.

Then we use two statistical indicator reflecting the goodness of fit of the model to indicate the difference between the predicted value and the actual value; finally, we apply a combined statistic of MSE, R^2 to demonstrate the accuracy of the prediction model, as follows table 3, table 4.

Table 3 Statistical Indicator of BTC

BTC	MSE	R^2
ARIMA	1146427	0.98367
LSTM	907281.7	0.99441

Table 4 Statistical Indicator of Gold

GOLD	MSE	R^2
ARIMA	201109	0.996754
LSTM	261244	0.992154

The MSE of the two prediction models are relatively small and the R-Square is extremely close to 1, indicating the goodness-of-fit of our prediction model is excellent.

5.4 Sensitivity Analysis

In designing the mathematical planning model, the transaction cost rate is also a part of our model. For sensitivity analysis, we use different transaction cost rates to fit the model several times and finally get a line graph of total revenue versus transaction cost rate to reflect its sensitivity. As follows, fig 9, in which the Y-axis represents the logarithmically processed total asset value, and the X-axis represents the bitcoin transaction rate.

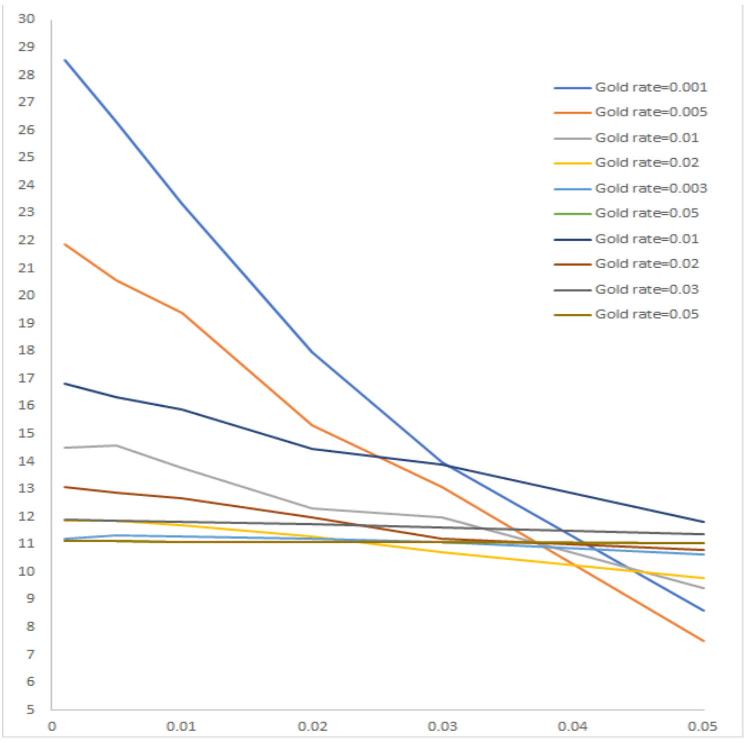


Fig 9 Sensitivity Analysis Chart

The chart shows that when gold trading rates are relatively low, an increase in bitcoin trading rates will significantly decrease total asset value. When gold trading rates are relatively high, an increase in bitcoin trading rates has little impact on the total asset value. Similarly, the reverse is also true. So we believe that total asset value is sensitive to transaction rates.

5.5 Trading Memorandum

Based on the above, we write up the following memorandum:

Our prediction model dynamically uses historical asset price data before each trading day to estimate future price movements over time. To ensure the robustness and generalization of the model, we integrate two forecasting models, and the integrated learner dynamically adjusts the weights between the two algorithms. The ARIMA model has more weight to capture the long-term trend when the prediction period is short. When the prediction period lengthens, the integrated learner increases the weight of the LSTM model to predict the future price trend better. To ensure the robustness and generalization of the model, we integrate two forecasting models, and the integrated learner dynamically adjusts the weights between the two algorithms. When the prediction period is short, the ARIMA model has more weights to capture the long-term trend. When the prediction period lengthens, the integrated learner increases the weight of the LSTM model to predict much longer-term data changes better. As shown in the previous sections, our model predicts accurate outcomes even with an extended forecast period.

Our decision model considers a combination of investor returns and risk preferences with the goal of maximizing returns, minimizing risk, and balancing positions. The future data received from the prediction model is used to solve the optimal trading solution using a multi-objective mathematical programming model, considering the fact that gold is not open on weekends. At the same time, we give the optimal solution for different trading frequencies considering the higher trading fees of bitcoin and gold, and the trader is free to change the trading frequency.

Combining the prediction model and the decision model, we give the transaction track records with a trading period of 7 days, as illustrated in fig 10.

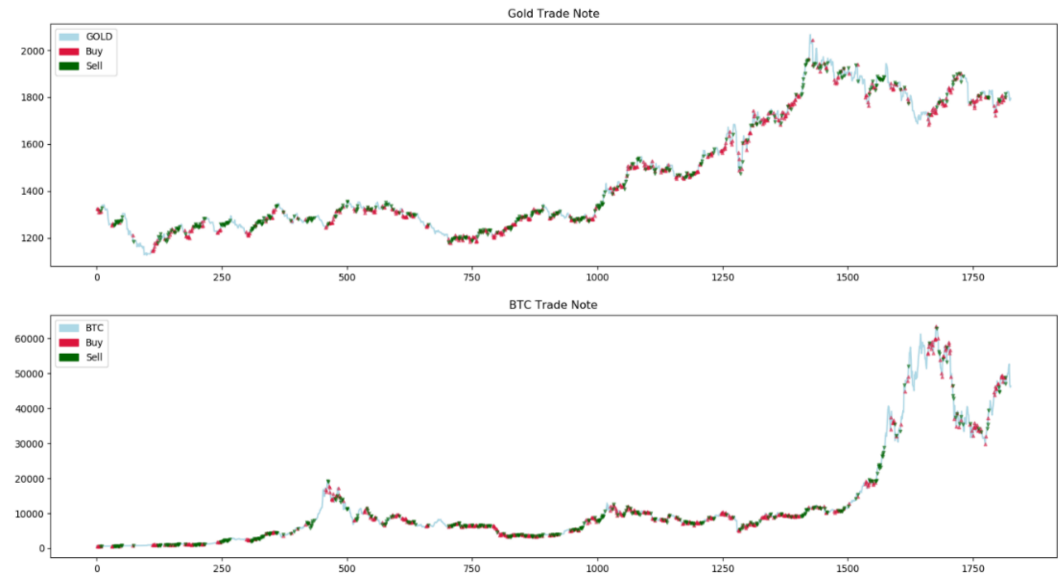


Fig 10 Transaction Track Records From 7-day Strategy

Our model can output accurate forecast data on the one hand. And flexibly adjust the value-at-risk according to the different investment styles of different investors on the other hand. Thus, we can achieve a balanced position in different conditions and achieve a balance between risk and return.

Our decision model integrates investor returns and risk preferences to maximize returns, minimize risk, and balance positions. We use a multi-objective mathematical programming model to solve for the optimal trading solution using future data obtained from the forecasting model, taking into account the fact that gold is not open on weekends. Also, we give the optimal solution for different trading frequencies considering the high transaction fees of bitcoin and gold. And the trader can freely change the trading frequency.

Combining the prediction model and the decision model, we give the transaction track records with a trading period of 7 days, as illustrated in

Our model can output accurate forecast data on the one hand, and flexibly adjust the value-at-risk according to the different investment styles of different investors on the other hand. Thus, we can achieve a balanced position in different conditions and achieve a balance between risk and return.

Our model also considers the impact of different transaction rates on trading strategies and can achieve optimal decisions under different transaction rates. However, it should be concerned that when the asset trading rates vary considerably, it may result in unstable returns and sharp fluctuations in asset prices.

6 Strength and Weakness

6.1 Strength

- We applied the idea of ensemble learning then integrated ARIMA and LSTM models in purpose to make the prediction model with better generalization ability and robustness.
- The LSTM model with higher weights in ensemble learning has significant advantages in time series modeling, which can selectively remember or forget short-term data while having a long-term memory function.
- The LSTM model has a high migratory degree and can maintain the prediction when the data features change.
- The LSTM model needs big data for support. Therefore, we use the ARIMA model for replacement in the pre-forecasting.
- The multi-objective mathematical programming equation we use for the portfolio investment decision problem is highly generalized and can solve many similar cost-revenue problems.
- The multi-objective mathematical programming model we use integrates three factors: long-term return, short-term return, and risk, which makes the investment decision more scientific and reasonable.
- We solve the problem of possible inaccurate fluctuations in the prediction model and the high fees that prevent high-frequency trading by increasing the buy-sell time interval.
- We tested the model's credibility and simulated the possible impact of gross return variation caused by changes in transaction fees.

6.2 Weakness

- The model is very computational and time-consuming, and each training epoch takes a long time
- The prediction model still has some time lags.
- We do not consider asset liquidity, and if possible, a quantitative measurement of asset liquidity could be integrated into the model.

7 References

[1]Ruthotto Lars, Haber EldadDeep neural networks motivated by partial differential equations. CoRR, abs/1804.04272, (2018)

[2]Aviv Yossi. The effect of collaborative forecasting on supply chain performance. Management Sci. (2001) 47(10):1326–1343

[3]Bu Hyoung Lee, William W. S. Wei, The use of temporally aggregated data in modeling and testing a variance change in a time series, Communications in Statistics - Simulation and Computation, 10.1080/03610918.2021.1928197, (1-18), (2021).

[4]Dikaios Tserkezos, The Effects of Temporal Aggregation and Random Sampling on the Power of the Augmented Dickey Fuller Stationarity Test: A Monte Carlo Study, Money, Trade and Finance, 10.1007/978-3-030-73219-6, (223-233), (2021).

[5]Wai-Sum Chan, On temporal aggregation of some nonlinear time-series models, Econometrics and Statistics, 10.1016/j.ecosta.2020.03.008, 21, (38-49), (2022).

Appendix

The following are the actual python execution scripts and programs used in this modeling process

```
    Prediction model(part):
def load_data(df, sequence_length=10, split=0.8):
    data_all = np.array(df).astype(float)
    scaler = MinMaxScaler()
    data_all = scaler.fit_transform(data_all)
    data = []
    for i in range(len(data_all) - sequence_length - 1):
        data.append(data_all[i: i + sequence_length + 1])
    reshaped_data = np.array(data).astype('float64')
    x = reshaped_data[:, :-1]
    y = reshaped_data[:, -1]
    split_boundary = int(reshaped_data.shape[0] * split)
    train_x = x[: split_boundary]
    test_x = x[split_boundary:]
    train_y = y[: split_boundary]
    test_y = y[split_boundary:]

    return train_x, train_y, test_x, test_y, scaler

def get_data(df, sequence_length=10, split=40):
    data_all = np.array(df).astype(float)
    scaler = MinMaxScaler()
    data_all = scaler.fit_transform(data_all)
    data = []
    for i in range(len(data_all) - sequence_length - 1):
        data.append(data_all[i: i + sequence_length + 1])
    reshaped_data = np.array(data).astype('float64')
    x = reshaped_data[:, :-1]
    y = reshaped_data[:, -1]
    split_boundary = split
    train_x = x[: split_boundary]
    test_x = x[split_boundary:]
    train_y = y[: split_boundary]
    test_y = y[split_boundary:]
    return train_x, train_y, test_x, test_y, scaler

def build_model():
    model = Sequential()
    model.add(LSTM(50, input_dim=1, return_sequences=True))
    print(model.layers)
    model.add(LSTM(100, return_sequences=False))
    model.add(Dense(1))
    model.add(Activation('linear'))
    model.compile(loss='mse', optimizer='rmsprop')
    return model

def train_model(train_x, train_y, test_x, test_y):
    model = build_model()
    try:
        history = model.fit(train_x, train_y, batch_size=256, epochs=100,
validation_split=0.01)
        predict = model.predict(test_x)
        predict = np.reshape(predict, (predict.size,))
    except KeyboardInterrupt:
        print(predict)
        print(test_y)
    fig1 = plt.figure(figsize=(16, 10))
    plt.plot(history.history['loss'], label='Training Loss',
color='crimson')

if __name__ == '__main__':
    data = gold_data
    #data = gold_data
    train_x, train_y, test_x, test_y, scaler = load_data(data,
sequence_length=3, split=0.1)
    train_x = np.reshape(train_x, (train_x.shape[0], train_x.shape[1], 1))
    test_x = np.reshape(test_x, (test_x.shape[0], test_x.shape[1], 1))
    predict_y, test_y = train_model(train_x, train_y, test_x, test_y)
    predict_y = scaler.inverse_transform([[i] for i in predict_y])
    test_y = scaler.inverse_transform(test_y)

    from sklearn.metrics import mean_squared_error
    X = gold_data
    #X = gold_data
    size = int(len(X) * 0.1)
    train, test = X[0:size].copy(), X[size:len(X)].copy()
    history = [x for x in train]
    predictions = list()
    for t in range(len(test)):
        model = ARIMA(history, order=(0, 1, 1))
        model_fit = model.fit()
        output = model_fit.forecast()
        yhat = output[0]
        predictions.append(yhat)
        obs = test[size + t]
        history.append(obs)
        # print('predicted=%f, expected=%f' % (yhat, obs))
    error = mean_squared_error(test, predictions)
    print('Test MSE: %.3f' % error)
```

Decision model:(part)

```
    for n in range(len(assets_rates) - tradeday):
        port.profit(assets_rates.iloc[n, :])
        rate_next_three_days = assets_rates_pred.iloc[n + 1:n + 1 +
tradeday, :].values
        # rate_next_three_days = assets_rates.iloc[n + 1:n +
4, :].values
        cons = (
            {'type': 'ineq', 'fun': lambda x: port.weights[0] - x[0] *
(1 + port.transaction_GOLD) - x[1] * (1 + port.transaction_BTC) -
0.01},
            {'type': 'ineq', 'fun': lambda x: port.weights[0] - x[1] *
(1 + port.transaction_BTC)},
            {'type': 'ineq', 'fun': lambda x: port.weights[0] - x[0] *
(1 + port.transaction_GOLD)},
            {'type': 'ineq', 'fun': lambda x: port.weights[1] + x[0]},
            {'type': 'ineq', 'fun': lambda x: port.weights[2] + x[1]},
            # {'type': 'ineq','fun': lambda x: -(port.weights[2] +
x[1])},
            #{'type': 'ineq', 'fun': lambda x: - x[0] + 0.1},
            #{'type': 'ineq', 'fun': lambda x: - x[1] + 0.1},
        )
        res = opt.minimize(fun, np.array([0.0, 0.0]), constraints=cons,
method='COBYLA',
                                    options={'gtol': 1e-6, 'disp':
False})
        if assets_rates.index[n].weekday() > 4:
            res.x[0] = 0.0
        # disturbance = np.random.normal(0, 0.05)
        # res.x[0] = res.x[0] * (1 + disturbance)
        # res.x[1] = res.x[1] * (1 + disturbance)
        port.change_GOLD_weights(res.x[0])
        port.change_BTC_weights(res.x[1])
        GOLD_transactions.append(res.x[0] * port.returns)
        BTC_transactions.append(res.x[1] * port.returns)
        port.settle()
```