COP290: Design Practices
Assignment-1: Trading Simulator And Analyzer

Date Updated: 07th Feb 2024

Subtask 3 [Deadline: 13th Feb 2024 (20 Marks)]

In this subtask, we finally implement some trading strategies, backtest them on past data, and get some basic insights into the world of algorithmic trading.

This document will be divided into multiple sections. The first section will give you some background and glossary for the terms we will be using throughout this document. Second section will be about some constraints and assumptions we make throughout the assignment. Further, there are some general instructions for the subtask. The subsequent sections will ask you to implement different strategies, and will include specific instructions for evaluation.

Background

Some resources that will help you get going -

- https://www.investopedia.com/terms/p/position.asp
- For glossary, refer this link. You need not go through all the terms - only look up for terms you find in this assignment doc.
- https://margcompusoft.com/m/square-off-in-share-market/

Any other background material required will be shared over Piazza.

Constraints and Assumptions

Following constraints and assumptions need to be followed throughout the assignment -

- To simplify our analysis, we will only be considering closing price for all stocks, unless stated otherwise. Thus, the term "price" will mean "close price".
- Since we should not hold any positions at the end of the trading period, any positions held should be squared-off at the closing price of end_date (If the end_date is a non-trading day, you should close the trade at the price of last trading day before end_date).

Common Instructions

- This subtask needs to be done in C++. The data you need to work upon will be generated in Python, as you have done in Subtask 1. Write the data into a suitable file format (Mention reason for choosing the file format in your report). The data needs to be read in C++ and all strategies should be implemented in C++. Your code should be clean and modular (there are some marks reserved for this).
- For every strategy that you implement, you'll need to write data to two different CSV files. Further in this document, we'll call them the daily CashFlow (cash-in-hand, could be negative) and order-statistics files. The basic layout will remain same across all the strategies.
- For daily_cashflow.csv, you should store date, and the cashflow presently. For order_statistics.csv, you should store date, order_direction (BUY or SELL), quantity (This will mostly be 1 for our analysis), and price at which the trade was made. You ONLY write to order_statistics.csv on days when a trade is made - you don't write a row for each day.
- For "EACH" strategy, you should also create a file called "final_pnl.txt". The only thing you need to write here is your final profit/loss (along with sign) after squaring off your position.
- We will release baseline results for every strategy later. You could use them to check your code.
- For strategies where we have date as parameter, it will always be represented as a string, in the format "DD/MM/YYYY".
- Any intermediates files created during execution should be deleted before ending the execution. There should only be 2 (or 3) CSV files and 1 txt file generated after the execution. This is to avoid any interference across test cases.
- To represent decimals, you should use "double" datatype in C++

1 Momentum Based Strategies

Let's move on to the first (class of) strategy that we'll implement. These are called momentum-based strategies. Here, we'll try to observe some pattern in the price series, and use the trend to place orders for the token.

1.1 Basic Strategy

The first most-basic strategy that we'll implement will observe the price series for past n days. Particularly, the algorithm we'll write is -

- If the price has been monotonically increasing for the last n days, we will buy 1 share today.
- If the price has been monotonically decreasing for the last n days, we will sell 1 share today.

Constraints and Assumptions

- We assume that we can short-sell a stock and can have a short position.
- The max position we can take is +x, and the min position we can have is -x. Thus, our position always stays in [-x, +x]
- Since we need the data of past n days even for start_date, you need to write appropriate data using Python to the data file.
- To understand more clearly, refer to this Piazza followup
- $n > 0$

Intuition

The simple intuition behind this strategy is as follows - If the price has been increasing for the last n days, we expect the price to increase further, and thus, we buy the share (hoping to sell it in the future at a higher price). Similar reasoning holds for selling the stock.

Parameters to the Strategy

- n - The number of past days to consider
- x - The maximum position allowed

Implementation

To run this strategy, we will run the command

```
make strategy=BASIC symbol=SBIN n=5 x=2 start_date="b" end_date="a"
```

The configurable parameters in this are - symbol, n, x, start_date and end_date. Running this command should create 2 csv files - daily_cashflow.csv and order_statistics.csv.

You May Want to Try

What if we only allow the position to be [0, +x]. (We no longer assume that we can take a short position). You can try this strategy and report anything interesting that you find. This code for this part will not be evaluated, though.

1.2 Trend-based Strategy, n-Day Moving Average (DMA)

Once you have implemented the basic strategy, we can make a simple extension to the same. Now, we relax the monotonic assumption, and instead work with average price of last n days. Define the strategy as follows -

- Calculate the mean price of past n days. Call this DMA. Also calculate the standard deviation.
- If the current price is greater than DMA by $\geq$ p standard deviations (sd calculated over same period), buy 1 share. Otherwise, if the current price is less than DMA by $\geq$ p s.d., sell 1.

Constraints and Assumptions

- We assume that we can short-sell a stock and can have a short position.
- The max position we can take is +x, and the min position we can have is -x. Thus, our position always stays in [-x, +x].

Intuition

When the current closing price crosses above the n-day moving average (DMA), it may be considered a buy signal. This suggests that the stock's price is trending upwards. Similar reasoning holds for sell signal. Further, we include p s.d. change to be confident about our trend.

Parameters to the Strategy

- n - The number of past days to consider for average
- x - The maximum position allowed
- p - The standard deviation threshold

Implementation

To run this strategy, we will run the command

```
make strategy=DMA symbol=SBIN n=50 x=3 p=2 start_date="a" end_date="b"
```

The configurable parameters in this are - symbol, n, x, p, start_date and end_date. Running this command should create 2 csv files - daily_cashflow.csv and order_statistics.csv.

1.3 Improving DMA

We would want to improve the plain-DMA strategy. We will do this by incorporating two new features to our basic algorithm -

- **Implement stop-loss:** If we are not able to close positions for quite some time, we close it forcefully to avoid losing more money. Thus, given a parameter, max_hold_days, we close our positions after max_hold_days, if we were not able to close otherwise.
- **Smoothing Factor:** Instead of calculating average of price of last n days directly, we will instead use the following algorithm to calculate adaptive Moving Average (AMA) -
  - First, calculate the efficiency ratio (ER) using

    $$\texttt{ER} = \frac{\text{Change in Price over n days}}{\text{Sum of Absolute Price Change over n days}}$$

    The denominator = Sum of $|\texttt{P}_i - \texttt{P}_{i-1}|$ for n days (including today)
    If denominator is 0, skip.
  - Calculate Smoothing Factor (SF) as -

    $$\texttt{SF}_t = \texttt{SF}_{t-1} + c_1 \times \left(\frac{\frac{2\times ER}{1+c_2} - 1}{\frac{2\times ER}{1+c_2} + 1} - \texttt{SF}_{t-1}\right)$$

    We will use $SF_0 = 0.5$. SF_0 is the smoothing factor on start_date.
  - Finally, calculate

    $$\texttt{AMA}_t = \texttt{AMA}_{t-1} + \texttt{SF}_t \times (\texttt{P}_t - \texttt{AMA}_{t-1})$$

    where $\texttt{P}_t$ is the price on day t. Take $\texttt{AMA}_0 = \texttt{P}_0$
  - Buy 1 share when the price is greater than $\texttt{AMA}_t$ by $\geq$ p %. Similarly, sell when the price is less than $\texttt{AMA}_t$ by $\geq$ p%.

Constraints, Assumptions and Intuitions similar to DMA. If you currently have a position > 1, and your strategy indicates that you should sell a share, you should sell the one that you had bought earliest. Similar for the other side of position.

Parameters to the Strategy

- x - The maximum position allowed
- p - Percent change needed
- n - Number of past days needed to calculate ER
- max_hold_days - Maximum number of days we can hold position
- c_1
- c_2

Implementation

To run this strategy, we will run the command

```
make strategy="DMA++" symbol=SBIN x=4 p=5 n=14 max_hold_days=28 c1=2 c2=0.2
start_date="a" end_date="b"
```

The configurable parameters in this are - symbol, x, p, n, max_hold_days, c1, c2, start_date and end_date. Running this command should create 2 csv files - daily_cashflow.csv and order_statistics.csv.

1.4 Using Indicators

To perform trend analysis, we will use indicators on the price series.

1.4.1 MACD

- Calculate the Exponential Weighted Mean (EWM) of prices for the past 12 days (Includes current price).

$$\texttt{EWM}_t = \alpha \times (\texttt{P}_t - \texttt{EWM}_{t-1}) + \texttt{EWM}_{t-1}$$

  where $\alpha = \frac{2}{12+1} = \frac{2}{13}$. Call this Short_EWM. Take $\texttt{EWM}_0 = \texttt{P}_0$
- Calculate the EWM of prices for past 26 days. Call this Long_EWM.
- Define `MACD` = `Short_EWM` - `Long_EWM`
- Calculate Signal Line as `Signal` = EWM of MACD for past 9 days.
- If `MACD` > `Signal`, buy 1 share. If `MACD` < `Signal`, sell 1 share.

Constraints and Assumptions are same as MDA.

Implementation

To run this strategy, we will run the command

```
make strategy=MACD symbol=SBIN x=3 start_date="a" end_date="b"
```

The configurable parameters are x, start_date and end_date. Running this command should create 2 csv files - daily_cashflow.csv and order_statistics.csv

1.4.2 Relative Strength Index (RSI)

RSI is one of the most used indicators for trend analysis. The algorithm we use will be as follows -

- Define average gain over last n days as the mean of Gain over the last n days. (Gain is defined as $\texttt{max}(\texttt{P}_i - \texttt{P}_{i-1}, 0)$). Similarly define average loss.
- Calculate Relative Strength (RS) as

$$\texttt{RS} = \frac{\text{Avg Gain}}{\text{Avg Loss}}$$

- Finally,

$$\texttt{RSI} = 100 - \frac{100}{1 + \text{RS}}$$

- If RSI crosses below `oversold_threshold`, we generate a buy signal. When RSI crosses above `overbought_threshold`, we generate a sell signal.

Constraints and Assumptions similar to DMA. One additional constraint here is that $\texttt{overbought\_threshold} \geq \texttt{oversold\_threshold}$.

Tip: Try drawing graphs with stock prices and RSI to visually get intuition for the RSI-based strategy.

Implementation

To run this strategy, we will run the command

```
make strategy=RSI symbol=SBIN x=3 n=14 oversold_threshold=30 overbought_threshold=70
start_date="a" end_date="b"
```

The configurable parameters are x, n, oversold_threshold, overbought_threshold, start_date and end_date. Running this command should create 2 csv files - daily_cashflow.csv and order_statistics.csv.

1.4.3 ADX

- Calculate True Range ($\texttt{TR}_t$) as maximum of $\texttt{High}_t - \texttt{Low}_t, \texttt{High} - \texttt{Prev. Close}, \texttt{Low} - \texttt{Prev. Close}$
- Define `DM+` as $\max(0, \texttt{High}_t - \texttt{High}_{t-1})$. Similarly define `DM-` using Low price instead of High.
- Define `ATR` as EWM of `TR` over past n days. (use formulas similar to above, and $\texttt{ATR}_0 = \texttt{TR}_0$
- Define `DI+` as EWM of $\frac{DM+}{ATR}$ over past n days. Similarly define `DI-`.
- Define `DX` as $\frac{"DI+" - "DI-"}{"DI+" + "DI-"} \times 100$, and `ADX` as the EWM of `DX` over the past n days.
- Finally, if `ADX` is greater than `adx_threshold`, generate a buy signal. If `ADX` is less than `adx_threshold`, generate a sell signal.

Constraints and Assumptions are similar to DMA.

Implementation

To run this strategy, we will run the command

```
make strategy=ADX symbol=SBIN x=3 n=14 adx_threshold=25 start_date="a" end_date="b"
```

The configurable parameters are symbol, x, n, adx_threshold, start_date, end_date. Running this command should create 2 csv files - daily_cashflow.csv and order_statistics.csv.

2 Linear Regression

Let's next implement a Machine Learning Algorithm. We will use Linear Regression to predict the Price of Stock at day t.

Particularly, we will use the following Linear Regression Equation -

$$\texttt{Close}_t = \beta_0 + \beta_1 \texttt{Close}_{t-1} + \beta_2 \texttt{Open}_{t-1} + \beta_3 \texttt{VWAP}_{t-1} + \beta_4 \texttt{Low}_{t-1} + \beta_5 \texttt{High}_{t-1} + \beta_6 \texttt{(No of Trades)}_{t-1} + \beta_7 \texttt{Open}_t$$

You can learn more about linear regression here.

- You will learn the parameters using historical data. The data you will use will be from `train_start_date` to `train_end_date`.
- Use the parameters and the equation to predict the close price for the current day.
- If the predicted price is greater than the actual price by $\geq$ p%, we buy the stock.
- If the predicted price is less than the actual price by $\geq$ p%, we sell the stock.

Intuition

If the predicted price is higher than the actual price, we expect the actual price to come to predicted price. Thus we buy the stock - in the hope to sell it later at a higher price.

Constraints and Assumptions

- `train_end_date` < `start_date`
- We assume that we can short-sell a stock and can have a short position.
- The max position we can take is +x, and the min position we can have is -x. Thus, our position always stays in [-x, +x].
- For results to be consistent and autograded, you should only use Normal Form Equations of Linear Regression. You are free to experiment with Gradient Descent, but your final results should only use Normal Equations.

Parameters to the Strategy

- `x` - The max position
- `train_start_date`, `train_end_date`
- `p` - The percent different required

Implementation

To run this strategy, we will run the command

```
make strategy="LINEAR_REGRESSION" symbol=SBIN x=3 p=2 train_start_date="a"
train_end_date="b" start_date="c" end_date="d"
```

The configurable parameters are - symbol, x, p, train_start_date, train_end_date, start_date, end_date. Running this command should create 2 csv files - daily_cashflow.csv and order_statistics.csv.

You May Want to Try

You can try many different features as input variables to the linear regression model. Feature engineering is a very crucial aspect of Linear Regression Models, and will help you achieve better results.

As you might have realized, training the coefficients takes some time. Thus, we decided to train the coefficients only once on a given interval, and then use those on the "complete" interval. You might want to look into online linear regression algorithms.

3 Best of All

We define a "best-of-all" strategy as the strategy which gives best (overall) PnL (could be negative) among all the strategies discussed so far. (For this strategy, you could not intermix strategies, each strategy will behave independently). To do so, we would not want to wait for each strategy to start and finish execution. Instead, we will run all the strategies in parallel. You should use pthreads or OpenMP for the same.

Constraints and Assumptions

For this stategy, you may assume the following -

- For basic, we will always use n=7. For DMA, n=50. For DMA++, RSI and ADX, n=14.
- Maximum position in all cases will be restricted to x=5.
- adx_threshold=25, oversold_threshold=30, overbought_threshold=70, max_hold_days=28, c1=2, c2=0.2.
- For DMA, p=2, while for DMA++, p=5.
- For linear regression, use one-year past data. For example, if start_data is 2nd April 2023 and end_data is 29th May 2023, then you should use train_start_date as 2nd April 2022 and train_end_date as 29th May 2022. For this strategy, we will guarantee that start_date and end_date lie in same year.

Implementation

To run this strategy, we will run the command

```
make strategy="BEST_OF_ALL" symbol=SBIN start_date="a" end_date="b"
```

The configurable parameters are - symbol, start_date and end_date. Running this command should create 2 csv files - daily_cashflow.csv and order_statistics.csv.

4 Mean-Reverting Pairs Trading Strategy

In this strategy, we move on to implement a strategy which is based on a pair of stocks, rather than a single stock. To make things simple, we will assume that we have already tested for correlation/cointegration between the stocks, and the pair is ready to use for the strategy.

- Given a pair of stocks, define spread between their prices as $\texttt{Spread}_t = \texttt{P}_{S_{1,t}} - \texttt{P}_{S_{2,t}}$

- Given a lookback_period of n days, calculate the mean (called rolling mean) and standard deviation (called rolling std dev) of the spread of past n days. (Include current day as well)
- Define z-score as

$$\texttt{z\_score} = \frac{\text{Spread - Rolling Mean}}{\text{Rolling Std Dev}}$$

- Given z-score threshold as `threshold`, if `z_score` $>$ `threshold`, generate a sell signal. Sell signal for the spread means that we sell S_1 and buy S_2. Similarly, if `z_score` $< -$`threshold`, generate a buy signal.

Intuition

In Pairs Trading Strategy, instead of betting on the price of a single stock, we instead bet on the spread between the price of a pair of stocks. Thus, even if both the prices are increasing or decreasing, we only care about the spread between the two.

When the z-score is high, we expect the spread to return to its mean, and hence, we short the spread (sell the spread). Similarly, when z-score is low, we long the spread.

Constraints and Assumptions

- We assume that we can short sell a stock and can have a short position in any of the stocks.
- The max position "in the spread" we can take is +x, and the min position is -x. Thus, our position for each stock always stays in [-x, +x].

Parameters to the Strategy

- `x` - The maximum position allowed
- `n` - The lookback period
- `threshold` - The z-score threshold

Implementation

To run this strategy, we will run the command

```
make strategy=PAIRS symbol1=SBIN symbol2=ADANIENT x=5 n=20 threshold=2
start_date="a" end_date="b"
```

The configurable parameters are - symbol1, symbol2, x, n, threshold, start_date and end_date. Running this command should create 3 csv files - daily_cashflow.csv, order_statistics_1.csv, and order_statistics_2.csv.

You May Want to Try

To simply things, we assumed that the given pair is already correlated/cointegrated. Read more about cointegration, and how you use that to determine if the given pair can be used for this strategy, and indeed exhibits mean-reverting behavior. (You can do this easily in Python). This part will not be graded.

There are many parameters in this strategy that you can play around with. For example, rather than closing positions on the opposite side of threshold, you may also close position at zero z-score level. You could also use the same mean and std dev to close your position, that you had used to open it (rather than using new rolling mean and std dev) - try reasoning why this would be better.

4.1 Stop-Loss in Pairs Trading Strategy

Often, we are stuck with unwanted positions in pairs trading strategy, and we want to clear them off to take newer positions and earn profits (Recall that we have a constraint on max position we can take). Thus, we include stop-loss based constraint in our strategy, We have already seen time based stop-loss in DMA. Here we will be looking at Loss based Stop-Loss Strategy.

- Since we had taken position when the z-score crossed the threshold (in either direction), we expected the z-score to return to the mean. Instead, if the z-score has moved in the unexpected direction, we may want to close our positions.
- Given `stop_loss_threshold`, we will close our (single) position when the z-score crosses the `stop_loss_threshold`. Note that we will use the same mean and standard deviation to calculate z-score at which we had opened the position.

Intuition, Constraints and Assumptions are same as Pairs Trading Strategy.

An additional Parameter here is `stop_loss_threshold`.

Implementation

To run this strategy, we will run the command

```
make strategy=PAIRS symbol1=SBIN symbol2=ADANIENT x=5 n=20 threshold=2
stop_loss_threshold=4 start_date="a" end_date="b"
```

The configurable parameters are - symbol1, symbol2, x, n, threshold, stop_loss_threshold, start_date and end_date. Running this command should create 3 csv files - daily_cashflow.csv, order_statistics_1.csv, and order_statistics_2.csv.

5 Competitive Part (Bonus - 3 Marks)

Want to share some new insights in the data? You are free to experiment and play around. Try out new and different strategies, use any data that you want, and derive insights and results. You should submit the code for bonus part separately (in a different Moodle Link), along with a report.

This report will be evaluated based on quality (and not quantity). It could even be a 1-page report, but if your insights and understanding is really valuable, you can score some bonus marks! (Won't be easy to gain these marks, though. Only submit if you think you really have something good to showcase). At the discretion of the Instructor, you may even be called for a Viva to discuss your results and implementation.

Report via README.md

Along with your code, you should also submit a report. This report needs to be in .md format, and should be named README.md.

Your report should include any insights observed or optimizations performed. You could include graphs for patterns observed in different strategies.

Further Notes

This document aimed to give you an insight into the world of quantitative finance. With this, you now have some basic knowledge about algorithmic trading. You can now learn and implement new strategies. As a suggestion, you could also integrate the strategies that you have written with the website you have built in Subtask 2, and display PnL and position graphs, and derive insights into different trading strategies. You can write this project on your CV for internships :)

Some other suggestions -

- Draw graphs for various indicators and observe any trends.

- Integrate different file formats instead of only CSV.
- Intermix different strategies/indicators to generate stronger signals.

References

- https://www.investopedia.com/terms/m/macd.asp
- https://www.investopedia.com/terms/a/adx.asp
- https://www.investopedia.com/terms/r/rsi.asp
- https://www.investopedia.com/terms/m/mlr.asp
- https://scholarworks.uni.edu/cgi/viewcontent.cgi?article=1016&context=jucie
- https://en.wikipedia.org/wiki/Linear_regression