Lynx Hackathon Instructions

In this project you will build an algorithm for trading a set of financial instruments, also called assets. The goal is to produce the best trading strategy as measured by an out-of-sample period. This document is meant to be read in conjunction with the code example we sent you.

Input Description

The input will consist of historical prices of the assets as well as a few additional time series that can potentially affect the traded assets.

All data are provided at a daily frequency. A price dated as 2023-02-15 is to be interpreted as the *closing* price for that day. In other words, it is only available at the end of 2023-02-15 (say 20:00). Time series data that are not asset prices obey the same logic. All prices are assumed to be quoted in the same currency.

The assets will vary in many aspects such as the magnitude the prices are quoted in and the variance of price changes. We denote asset prices as $P_{T,j}$ and positions as $X_{T,j}$, where the index T refers to a specific day and the index j refers to a specific asset.

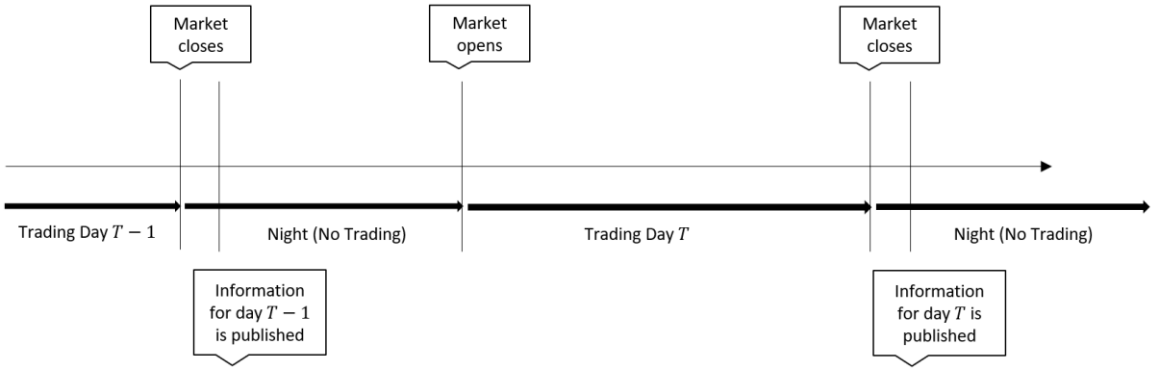
Market Mechanics and Trading Setup

In this project, we imagine trading different assets on a single exchange. The exchange is open during the day and closed during the night - the exact times are not important for this project. We place our orders prior to the market open and assume they execute at the closing price for that day. This is a conservative approach as normally you would execute an order throughout the day at various different prices.

The events on a generic day T are as follows:

1. Asset prices (and other input data) for day $T-1$ become available at the end of day $T-1$, or equivalently at the very beginning of day T.
2. Your algorithm executes while the market is closed, during the night from $T-1$, heading into trading on day T. Your orders for asset j, $\Delta X_{T,j} = X_{T,j} - X_{T-1,j}$ are then ready for execution prior to the market open.
3. Your orders execute while the market is open on day T, yielding a new position $X_{T,j}$.
4. The market closes, yielding new closing prices $P_{T,j}$ and any potential new information from other time series.
5. You calculate your day T return in asset j using *yesterday's* position as
   $R_{T,j} = X_{T-1,j} * (P_{T,j} - P_{T-1,j})$.
   Since this formula ignores transaction costs the returns calculated in this way are called gross returns.
6. Trading incurs a small transaction cost that is proportional to your order ΔX. Taking transaction costs into account, the formula for your return on day T on asset j is
   $$R_{T,j} = X_{T-1,j} \cdot \left(P_{T,j} - P_{T-1,j}\right) - \delta \cdot \left|X_{T-1,j} - X_{T-2,j}\right| \quad (1)$$
   where δ is the transaction cost coefficient. We make the simplified assumption of a constant $\delta = 0.0002$ for all traded assets. Returns calculated from this formula are called net returns.

It is important to take into account that there is a lag between *calculating* a desired position and actually *acquiring* that desired position. If this is not observed, then the algorithm is said to contain the benefit of hindsight, which is not allowed and impossible in practice. In finance this is often called a forward-looking-bias whereas in data science competitions it would be called a data leakage.



Trading Capital and Bookkeeping

In traditional cash-equity trading, you start with some capital C and invest X into different assets at the prevailing market price, leaving you with remaining capital $C - X$. As the market prices change, you will accumulate unrealized profits and losses. A key component here is the need to bookkeep how your capital changes as you buy/sell assets so that you for example don't end up with a negative capital account (i.e. a loan).

*In this project, we will **not** be engaging in cash-equity trading*. Instead, we will be engaging in *margin trading*. In this setup, we are free to buy and sell assets without actual cash transactions. Instead, we post a so called margin requirement amount to the exchange that covers the risk of our positions.

For our purposes, we can imagine that we have enough capital to meet any margin requirement, as long as we control our risk taking. This simplifies our setup and means you can focus more on developing the algorithm than doing bookkeeping. The total size of your position will be determined by how much risk you wish to take.

For more details on derivatives trading and margin requirements, you can find plenty of information online, but this is ***not*** necessary to complete this project successfully.

Short Selling

The second important aspect of derivatives trading is that it allows for short selling, that is to hold a position with negative exposure. This means that positions $X_{T,j}$ can take negative values as well as positive values. The calculation of returns on a position uses the same formula, irrespective of the sign of $X_{T,j}$.

Risk Management

The set of positions we have on any given day constitutes our *portfolio* for that day. Even though we operate under simplified assumptions as previously described, we need to take risk management into perspective.

Consider the price returns of a single asset as a time series: $\{r_1, r_2, r_3, \dots, r_T\}$. One measure of risk commonly used is the standard deviation of returns: σ_T. In finance, this is referred to as the *volatility* of an asset. Note that this measure can (and does) vary over time.

We can use our risk measure to inform position sizing in different assets. A desirable characteristic is for the total portfolio risk (measured as the standard deviation of portfolio profit and losses) to be contributed to in

equal parts from all assets. In other words, we do not have a diversified trading strategy if our portfolio allocates a large part of its risk to only a single asset. One way to achieve this is so called inverse volatility weighting. We increase/decrease positions according to the volatility of a given asset, done in a proportionate fashion.

It is also desirable for the total risk of your portfolio (measured as the standard deviation of portfolio profit and losses) to not change too drastically over time. This can be achieved by scaling all positions by the inverse of a trailing measure of total portfolio risk. There are more advanced methods of risk management and portfolio construction. The interested reader might want to look up Markowitz optimization. But we only require that you manage your risk in *some way*.

Examples for volatility scaling and portfolio risk scaling can be found in the example code.

Performance Metrics

Given the positions $X_{T,j}$ of your trading strategy it is easy to calculate the returns $R_{T,j}$ of your strategy for each asset while accounting for costs as in equation (1). The daily return of your strategy is then $R_T = \sum_j R_{T,j}$.

1. An important measure of strategy performance is called the Sharpe ratio. The Sharpe ratio measures the trade-off between your strategy return and the amount of risk you are taking. We use the following formula to calculate the Sharpe ratio from the daily strategy returns R_T:

$$SR(R_T) = \sqrt{252}\frac{Mean(R_T)}{STD(R_T)} \tag{2}$$

   Where $Mean(R_T)$ is the mean or average daily returns of your strategy and $STD(R_T)$ is the standard deviation of daily returns of your strategy. When the Sharpe Ratio is calculated from net returns, i.e. taking transaction costs into account, it is called net Sharpe Ratio, otherwise it is called gross Sharpe Ratio or Sharpe ex. slippage.

2. We will judge the performance of your strategy using a variant of the Sharpe Ratio called the Adjusted Sharpe Ratio:

$$SR_{adjusted}(R_T) = SR(R_T) \cdot \left(1 + \frac{SR(R_T)}{6} \cdot Skewness(R_T) - \frac{SR(R_T)^2}{24} \cdot Kurtosis(R_T)\right) \tag{3}$$

   Where $Skewness(R_T)$ is the skewness of daily strategy returns and $Kurtosis(R_T)$ is the kurtosis of daily strategy returns, see Maillard (2018) for details. The Adjusted Sharpe Ratio incorporates penalty terms that disincentivize negative skewness and high kurtosis. Your strategy will be evaluated based on the net Adjusted Sharpe Ratio.

3. The holding period and its inverse, the turnover, are measures of how fast your strategy trades. Holding period is defined as

$$HP = 2 \cdot \frac{\sum_{tj}|X_{t,j}|}{\sum_{tj}|X_{t,j} - X_{t-1,j}|} \tag{4}$$

   A high turnover / low holding period will cause large transaction costs. Most strategies face a trade-off between alpha decay (the tendency that gross performance deteriorates when trading slower) and transaction costs. Keep an eye on the difference between net and gross Sharpe Ratio to detect if your transaction costs become too high.

In practice, hedge funds use more complicated measures of performance which depend on the fund mandate, the traded markets and the preferences of investors. We will provide you with a Python library which you can use to calculate key figures and visualize the performance of your strategy.

Detecting data leakage

As explained earlier, it is crucial that your strategy uses only data until and including time T to calculate the desired position X_T. This is because we are simulating real trading in which you don't have information of future events. Breaking this convention is called a forward-looking-bias or a data leakage, depending on the context, and leads to automatic disqualification of a strategy.

There are several ways to detect a data leakage in your strategy. A simple method is to run the strategy on two different data sets, A_{ti} and B_{ti}. The two data sets are identical up to a certain time step T but differ for $t > T$. Positions X_t calculated by your strategy from A_{ti} and B_{ti} must be identical for $t \leq T$, otherwise you have a forward looking bias. Another way to check for a forward-looking-bias is to generate positions using a shorter history of data, and make sure that the positions are identical to those obtained when using the whole dataset. You will be required to apply a suitable test to your strategy.

During the event

During the event you will be given a data set which you can use to develop and test your own trading strategy – with our help of course. You can use any programming language you want, we typically use Python or Matlab. In order to evaluate your strategies we will hold back a part of the data, which is often called the out of sample period. Towards the end of the event, when you are done developing your strategy, we will give you access to the out of sample period. You will then be asked to run your strategy on the full data set and save the resulting positions to a csv file which we can evaluate.

External data

To ensure a level playing field participants may only use the provided data set and are not permitted to bring in additional data sources.

Further reading

Curious to learn more? There is quite a bit of material online which is freely available. A first starting point is the following LinkedIn post, as well as the more in-depth follow up article:

- The 10-line CTA | LinkedIn
- The 10-line CTA revisited | LinkedIn

And if you want to read even more, the following collection of lectures has some excellent material:

- Quantopian Lectures Saved · GitHub

Don't be shy to research and use well-established trading strategies. We reward performance, not originality.