Developing a Backtesting library in Python

Juan F. Imbet *Ph.D.*

Paris Dauphine University

M203 (M2)

What is backtesting?

- Backtesting is a simulation technique used to test the effectiveness of a trading strategy.
- You want to know what **would have happened** if you had used a particular strategy in the past.
- No backtest is perfect, as it relies on the assumption that what you see and would have done **ex-post** is what you would have seen (and done) **ex-ante**.
- However, it is a useful tool to test the robustness of a strategy.

Python libraries for backtesting

- Python has several libraries that can be used for backtesting, but we will develop our own.
- I have explored many of them, and identified two main issues:
  - Too much focus on technical analysis.
  - Little Portfolio Optimization.
  - Not fast enough for large datasets.
- Ideally we would like to focus more on the *what to buy* and *how much to buy* rather than on the *when to buy* (although important)

What we will do

- We will develop a simple backtesting library in Python.
- We will use a blockchain architecture to help the user reduce the probability of overfitting.

Some python libraries for backtesting

- **Backtrader**: A popular library for backtesting. It is very flexible and has a lot of features. However, it is not very fast.
- **Zipline**: Developed by Quantopian, it is a good library for backtesting, but it stopped being maintained. (Some libraries are not comptible.)

General Idea, Notation

- Time is discrete: $t = 0, 1, 2, \ldots, T$, e.g. daily, intraday, etc.
- Filtration of companies available for investment: S_t
- Filtration of information available at time t: $\mathcal{F}_t$
- Investment strategy: A function that maps the available set of companies and information to a portfolio: $\pi_t = \pi(S_t, \mathcal{F}_t)$
- Backtesting: Simulate the performance of the strategy π over a historical period.

First Module: Data, Moments, and Returns.

- The library should work with data from any asset as far as it is in the right format.
- For testing we will use data on US stocks.
- For the moment the only data in the information set corresponds to the market data of the companies.
- We will start with a narrow set of companies, and then we will expand it.
- We will use `pandas`, `numpy` and `scipy` for this module.
- Go to branch `branch1` in our repository.