Massachusetts Institute of Technology
Department of Electrical Engineering and Computer Science
6.3800 Introduction to Inference
Fall 2024

Computational Lab 1

Issued: Wednesday, September 4 **Due:** Thursday, September 19 at 11:59pm

---

Python Version: Make sure to run your code using Python Version 3.6.9 or newer.

Optional Reading: Python Scientific Lecture Notes, at `https://scipy-lectures.github.io/` is a useful Python reference. The NumPy information in Section 1.4 and the matplotlib information in Section 1.5 are likely to be particularly helpful in getting started with the computational labs.

Investing

This lab is a good (and hopefully fun) introductory lab and Python warm-up exercise. You'll implement two investment strategies and make some observations of their properties, which you'll learn how to analyze as an application of the inference tools we will develop later in the class.

Suppose you have some money and need to decide what to do with it. You can put it under your mattress, in a bank account, or invest it in some combination of bonds, stocks, or other funds. To simplify the problem, we'll just consider two choices among these possibilities: Investment A and Investment B. This is our investment "portfolio." Assume that each day[1] an investment has a return—money invested at the start of that day increases or decreases in value by the end of that day, reaching a new value. We will investigate two classic investment strategies described below.

> **Buy and Hold:** At the start of the first day, we split our initial wealth into two portions, putting the first portion into Investment A and the second into Investment B. We then let the value of the resulting *buy and hold portfolio* evolve according to the returns of these investments over time (i.e., many days) without further intervention.
>
> **Constant Rebalancing:** At the start of the first day, we invest our initial wealth as in the buy and hold strategy. But at the start of each subsequent day, we rebalance our portfolio by reallocating it between the two investments to the same proportion used to allocate our initial wealth. For example, if Investment A earns a better return than Investment B on a given day, then at the start of the next day we will sell some of Investment A and purchase some more of Investment B in order to adjust the proportion of our wealth in each investment to be the same as at the start. This strategy is referred to as a *constant rebalanced portfolio*.

[1] The investment period needs not be a single day, but for simplicity, let's restrict our attention to this case.

The buy and hold strategy is obviously simplest. And the constant rebalancing strategy is at least a bit counterintuitive since we will generally sell some of the investment that is doing better every day. But which strategy is better? Let's investigate.

Code and Data

In this computational lab, you are provided with the starter file `stocks.zip`. Download the file from the course website and extract it to your working directory. It contains

- Folder `data` with two csv files, `priceA.csv` and `priceB.csv`
- File `stocks.py`, which will serve as the starting template for your code.

Important: Do not change the function definitions in the code as they must remain consistent for automatic grading. Do not import additional modules since these modules might not exist in the autograder environment. The only parts of the code that you need to fill in are in blocks denoted by "`YOUR CODE GOES HERE`". If you would like, feel free to create additional functions as needed to help with the computation or to answer the questions in the lab, although this is not required. See the comment at the beginning of each function you complete for what the expected input and output are. You must use Python 3; we recommend version 3.6.9 or above.

Lab

First, let's compare these two investment strategies on a pair of stocks corresponding to two large (real) companies: Stock A and Stock B. In the files `priceA.csv` and `priceB.csv`, you'll find the daily stock price history of the two companies from August 1995 to August 2015, corresponding to roughly 5000 days over the 20 year period.[2]

(a) Implement both strategies on this data, starting with a \$1 investment and partitioning it equally into the two stocks (i.e., \$0.5 each). Plot your wealth over time. Specifically, in a single figure plot the wealth (i.e., total value of the portfolio) for the two strategies as a function of the number of days since the investing started. Describe what you observe, and offer an interpretation.

To further analyze and compare these two investment strategies, we'll now use simulated data for two different investments: Investment C and Investment D. Investment C is a fixed income investment, earning 5% interest every day, i.e., money invested grows by a factor $\gamma = 1.05$ every day. Investment D is more volatile: on a given day, its value changes by a factor of either $\beta = 1.4$ (i.e., grows) or $\alpha = 0.7875$ (i.e., shrinks).

[2]The data are aligned by date—the first rows of each correspond to day one, second rows of each correspond to day two, etc.

For an investment window of n days, there are then 2^n possible patterns of returns for Investment D.[3] For each of these possible patterns, you will compare the buy and hold and constant rebalancing strategies. Suppose we again start with a \$1 investment, partition it equally into Investments C and D (i.e., \$0.5 each), and invest for $n = 20$ days.

Sanity check. To help you get started generating (and ultimately checking) your code, note that at the end of the first day, the \$0.5 invested in Investment C is worth $\$0.5 \cdot \gamma = \$0.525$, and the \$0.5 invested in Investment D is worth $\$0.5 \cdot \beta = \$0.7$ or $\$0.5 \cdot \alpha = \$0.39375$, depending on whether the investment grows or shrinks. Suppose that it grows. Then at the beginning of the second day, the total wealth is $\$0.525 + \$0.7 = \$1.225$, so constant rebalancing will put $\$1.225/2 = \$0.6125$ into each of Investments C and D. At the end of the second day, the value of investment C in the constant rebalanced portfolio will be $\$0.6125 \cdot \gamma = \$0.643125$, and that of investment D will be $\$0.6125 \cdot \beta = \$0.8575$ or $\$0.6125 \cdot \alpha = \$0.48234375$, depending on whether Investment D grows or shrinks on the second day.

(b) Compute the average value of the portfolio at the end of the investment window for each strategy, where the averaging is over all the possible return patterns for Investment D. Which performs better, buy and hold or constant rebalancing?

Write your code for this part in the `compute_average_value_investments` function in `stocks.py`, returning the average value for each strategy (`average_buyandhold, average_rebalancing`).

Hint: You may find the Python function `numpy.binary_repr()` useful.

(c) For each of the return patterns, evaluate which strategy yields the bigger portfolio value at the end of the investment window. For what fraction of the patterns does buy and hold perform better?

If w_n is the value of wealth after n days, we refer to

$$r_n \triangleq \frac{1}{n} \log_2(w_n)$$

as the wealth *doubling rate* (or *growth exponent*). In this part, you will evaluate the doubling rate for the two strategies involving Investments C and D. But now, instead of considering all patterns of returns for Investment D, you will only consider patterns where there are equal numbers of up and down days over the n day window, i.e., $n/2 = 10$ days yielding return β and $n/2 = 10$ days yielding return α.

(d) Compute the doubling rate for the buy and hold and the constant rebalancing strategies above. Which one is larger?

[3] For example, for $n = 3$, the patterns would be $\alpha\alpha\alpha$, $\alpha\alpha\beta$, $\alpha\beta\alpha$, $\alpha\beta\beta$, $\beta\alpha\alpha$, $\beta\alpha\beta$, $\beta\beta\alpha$, and $\beta\beta\beta$.

Write your code for this part in the `compute_doubling_rate_investments` function in `stocks.py`, returning the doubling rate for each strategy (`doubling_rate_buyandhold`, `doubling_rate_rebalancing`).

(e) **(Optional, not graded—just for your enjoyment!)** Suppose in the buy and hold strategy, instead of putting half of your initial wealth of \$1 in Investment C, you more generally put a fraction $0 \leq \lambda \leq 1$. Is it possible to achieve a higher doubling rate by choosing $\lambda \neq 1/2$? Likewise, is it possible to achieve a higher doubling rate for the constant rebalancing strategy by choosing $\lambda \neq 1/2$?

What to upload: Upload to Gradescope your writeup as a single PDF file and your code as `stocks.py`. The writeup should include your answers and plots for parts (a)–(d). Scans/images of handwritten work are fine but please write neatly — we can't grade what we can't decipher! Plots should be part of the writeup and not uploaded as separate files.

Gradescope will run a number of tests which are visible to you. These are to make sure your code runs correctly in the Gradescope system. Please make sure you pass these tests as we will use the autograder to check your submission.