Project Research: Universal Portfolios & GAs

Andrea Landini

May 2023

Contents

Contents

Abstract

In this research I will briefly present the concept of the Universal Portfolio, first presented by Thomas Cover in the article of the same name, together with the performance obtained. To better understand Cover's study, I will mention constant rebalancing portfolios. In the second part of the study, I will present a genetic algorithm for the Knapsack problem, with the aim of showing how this algorithm can achieve results in portfolio management. The Knapsack Problem thus lends itself as a tool for framing the stock picking problem in question. I will use the following R libraries and compare their tools with my own in order to show the advantages and limitations of the strategies used. All the libraries presented are well known for data cleaning and data visualisation, with the only exception of 'logopt', which offers exactly the portfolio strategies presented by Cover.

1 Introduction

Suppose a gambler has an initial bankroll of X_0 and bets at each time step t a given fraction f of X_t. Should he win, he would get fX_t back and another fX_t, however a loss would shriken his bankroll fX_t. The bankroll would then evolve as

$$X_n = X_0(1+f)^S(1-f)^F$$

where S is the number of successes and F the number of failures, i.e., in particular $S+F=n$. The gambler would obtain a average growth (per bet) of

$$\left(\frac{X_n}{X_0}\right)^{1/n} = (1+f)^{S/n}(1-f)^{F/n}$$

where $S/n \to p$ the probability of success and $F/n \to 1-p$ the probability of failure. The term on the left is the growth rate per bet on average and the gambler wants to find an f that maximizes this quantity via the right-hand side. Taking the logs leads to

$$\frac{1}{n}\left(\log X_n - \log X_0\right) = \frac{S}{n}\log(1+f) + \frac{F}{n}\log(1-f),$$

or in the limit for n large we obtain:

$$E[g(f)] = p\log(1+f) + (1-p)\log(1-f) \tag{1}$$

where g is the expected growth rate per bet when betting fraction f, which is independent of n now. We can now apply this framework to investing decisions, since the payoff of each decision can differ significantly and the probability of success is not easily available. Suppose we have n assets, with random return vector $x \in R^n$, where the x_i are of the form $x_i = \frac{p_i(new)}{p_i(old)}$, i.e., relative price changes. In the spirit of Kelly's approach, we allocate fractions across these assets, so that the logarithmic growth rate is given as $\log f^\top x$.

$$f \in \Delta(n) \doteq \left\{ f \in R^n \mid \sum_i f_i = 1, f \geq 0 \right\}$$

Thus, We would have the relative price change realizations x_t in time t and allocations f_t, so that the logarithmic portfolio growth over time is given by

$$\sum_{t=1}^{T} \log f_t^T x_t$$

Equivalently the average logarithmic growth rate is given by:

$$\frac{1}{T}\sum_{t=1}^{T}\log f_t^T x_t \tag{2}$$

which is the generalization of (1), however now we spread capital along multiple assets.

1.1 Constant Rebalancing Portfolio

A Constant Rebalancing Portfolio (CRP) is one where the $f_t = f$ are constant over time with the implicit assumption that the return distribution is somewhat stationary and hence from a sequential decision perspective we maximize the expected logarithmic growth rate by picking the expectation maximizer in each step assuming i.i.d. returns. The term constant rebalancing arises from the fact that in each time step, we rebalance the portfolio to represent the constant allocation f across assets; note that this rebalancing usually incurs transaction costs.

One of the arguments for CRPs is that they can even generate a return when the log geometric mean of the random variable we are investing in is 0, e.g., a fair coin whose payout is so that we have a new bankroll of amount $1 + r$ when $X = 1$ and $1/(1 + r)$ if $X = 0$. More generally, suppose that we have a discrete one dimensional random variable $1 + X$ (this is our bet) with outcomes $1 + x_i \in R_+$with probability p_i, i.e., the x_i corresponding to the returns, so that the log geometric mean is at least 0 :

$$\sum_i p_i \log(1 + x_i) \geq 0$$

Betting a fraction f leads to the expected growth function:

$$r(f) \doteq \sum_i p_i \log(1 + f x_i)$$

Observe that $r(f)$ is strictly concave in f in the interval $f \in [0, 1]$ with $r(0) = 0$ and $r(1) \geq 0$ (here we use that the log geometric mean is at least 0). Thus by concavity (or Jensen's inequality) it follows:

$$r(f) = r(1 \cdot f + 0 \cdot (1 - f)) > f \cdot r(1) + (1 - f) \cdot r(0) \geq 0.$$

As such betting a fraction of $f = 1/2$ is always safe as long as the geometric mean is at least 1. However, for completeness, note that betting a fixed fraction of $1/2$ can be quite suboptimal

2 Universal Portfolios

Cover showed that the best CRP provides returns at least as good as buying and holding any particular stock, as his portfolios yield the average of the returns all stocks. Once the notion of Constant Rebalancing Portfolios is defined, given a set of assets, a natural question is whether one can estimate or compute the optimal allocation vector f given either distributional assumptions about the returns or actual data. The natural second order question is then, if so, whether it is possible while investing. Given relative price change vectors $x_1, \ldots, x_T$, we want to solve:

$$\max_{f \in R^n} \frac{1}{T}\sum_{t=1}^{T}\log f^T x_t$$

While this optimization problem can easily be solved with convex optimization methods, provided the price change vectors $x_1, \ldots, x_T$, this is not very helpful as the past is usually not a

great predictor for the future. Motivated by the strong connection to information theory, Cover realized that one can define Universal Portfolios, that are growth optimal for unknown returns for universal coding where an asymptotically optimal code can be constructed without knowing the source's statistics. The algorithm constructs a portfolio over time, so that when $T \to \infty$, for any sequence of relative price changes $x_1, \ldots, x_T$, then

$$\frac{1}{T}\sum_{t=1}^{T} \log f_t^T x_t \to \max_{f \in R^n} \frac{1}{T}\sum_{t=1}^{T} \log f^T x_t$$

where the f_t are dynamic allocations. However, the proposed algorithm spreads the capital across an exponential number of sequences arising from binary strings of length T (assuming the most basic case based on a binomial model), making it impractical for computational implementation as well as from an actual real-world perspective as one would probably be bankrupted by transaction costs (nonetheless, it has been developed a theoretically polynomial time implementable variant of Cover's universal portfolios).

The Universal Portfolios approximate the return of optimal fixed portfolio in the long term. Formally, with m stocks, let $\mathbf{x}_i = (x_{i,1}, \ldots, x_{i,m})^T \in R^m$ be the vector of stock return in day i. Note that on day i when we decide our portfolio, we only know the price of previous days, instead of $\mathbf{x}_i$. Here $x_{i,j}$ is the return of stock j in day i, i.e. $x_{i,j} = \frac{price\ of\ stock\ j\ on\ day\ i}{price\ of\ stock\ j\ on\ day\ i-1}$. On each day i, we use a vector $\mathbf{f}_i = (f_{i,1}, \ldots, f_{i,m})^T \in \Delta^m$ to represent the portfolio. Here Δ^m is the simplex $\left\{\mathbf{f} \in R^m \mid \sum_{j=1}^m f_j = 1\right\}$. On day i we allocate $f_{i,j}$ of our current total wealth to stock j. Thus our return on day i is $\mathbf{f}_i^T\mathbf{x}_i$, compared with day $i-1$. The total return after n days is $T_n = \prod_{i=1}^n \mathbf{f}_i^T\mathbf{x}_i$. With Universal Portfolios, we can approximate the return of optimal Constant Rebalancing Portfolio, as we defined it as a fixed wealth allocation throughout all days. Given a fixed wealth allocation $\mathbf{f} \in \Delta^m$, the return of CRP on day n is $S_n(\mathbf{f}) = \prod_{i=1}^n \mathbf{f}^T\mathbf{x}_i$. With Universal Portfolio algorithm, we can achieve

$$\frac{T_n}{S_n(\mathbf{f}^*)} \geq \frac{1}{(n+1)^{m-1}}$$

Here $\mathbf{f}^*$ is the optimal CRP. Considering the average return per day, we have

$$\left(\frac{T_n}{S_n(\mathbf{f}^*)}\right)^{\frac{1}{n}} \geq \frac{1}{(n+1)^{\frac{m-1}{n}}}$$

As $n \to \infty$, the right hand side approach 1 . Thus we get an average daily return comparable with optimal CRP. However, it has been shown that, in a market with m stocks, over n days,

$$\frac{\text{performance of UNIVERSAL}}{\text{performance of best CRP}} \geq \frac{1}{(n+1)^{m-1}}$$

By performance, we mean the return per dollar on an investment. The above ratio is a decreasing function of n. However, the average per-day ratio, $\left(1/(n+1)^{m-1}\right)^{1/n}$, increases to 1 as n increases without bound. For example, if the best CRP makes 1.5 times as much as we do each day over a period of 22 years, it is only making a factor of $1.5^{1/22} \approx 1.02$ as much as we do per year. Such implementations of Cover's algorithm are exponential in the number of stocks with worst-case run times of $\Theta\left(n^{m-1}\right)$. In some sense, Cover's algorithm divides its money evenly among all CRPs. Unfortunately, for some market sequences, the number of CRPs which perform near optimally can be as small as $1/\Theta\left(n^{m-1}\right)$.

As we can see in the example proposed by this portfolio, the Universal Portfolio is able to guarantee the average of possible portfolios. This means that the more portfolios are considered, the closer the performance of the Universal Portfolio will be to that of the best portfolio. This result is repeated for any combination from the dataset considered in the Cover paper.

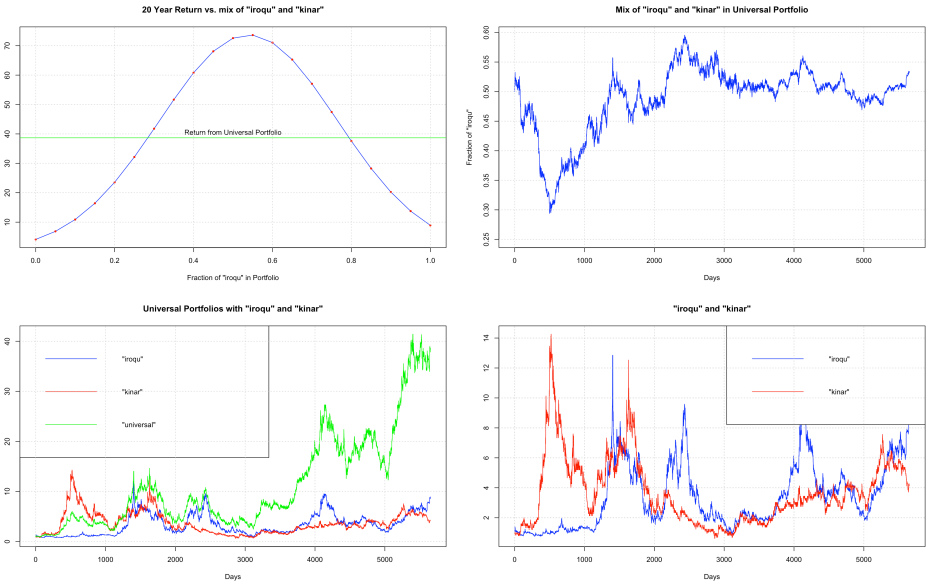

Figure 1: Results of Universal Portfolio for 2 stocks

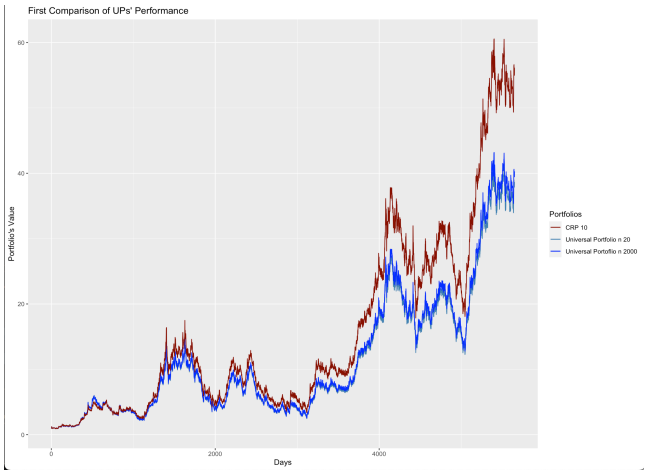

Figure 2: Comparison of Perfomances of 2 Universal Portfolios against a revisited version of the CRP

In order to highlight the limitations of such portfolios, I refined the notion of a constant rebalancing portfolio so as to consider two additional variables as inputs: the frequency with which the weights of the securities in the portfolio are rebalanced and the length of the time series within which the performance of the securities is considered. In other words, the frequency and length of portfolio rebalancing. This shows the possibility of outperforming the Universal Portfolios.

The following graphs show that these variables do not necessarily give a lower result than those observed by Cover, so one would have to find the optimal level of both variables to maximise one's performance. it is therefore necessary to frame the problem to try to solve these unknowns. Moreover, a further limitation of Cover is the limited number of shares that are drawn, and how it does not rebalance to exclude the worst ones by replacing them with better performing ones.

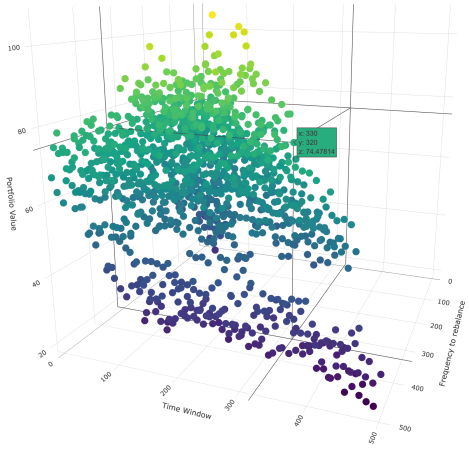

Figure 3: Different results for a 2 stocks portfolio for any given rebalancing frequency and time window

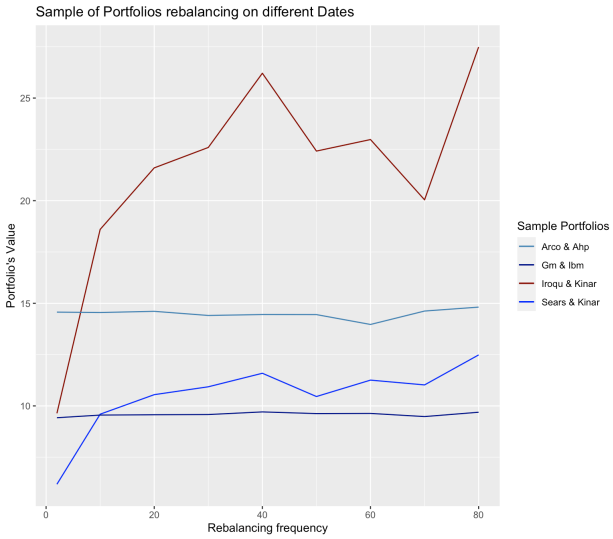

Figure 4: Different results for a 2 stocks sample portfolios for any given rebalancing frequency

3 Genetic Algortihm in Portfolio Decision

3.1 Knapsack Problem

A genetic algorithm is a heuristic optimization technique that can be applied to the Knapsack problem, which involves finding the combination of items that maximizes the total value while staying under a certain weight limit. The algorithm begins by creating an initial population of candidate solutions, which are evaluated using a fitness function that ensures the weight limit is not exceeded. The fittest individuals are selected to generate the next generation of candidate solutions using crossover and mutation operators. The process continues until a termination criterion is met, and the fittest individual in the final population is chosen as the optimal solution to the Knapsack problem.

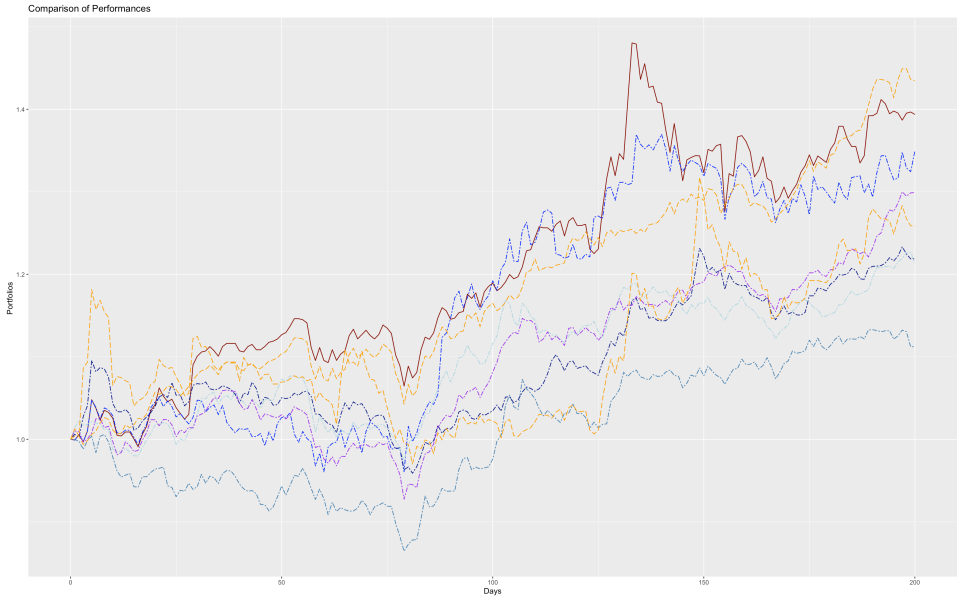

Figure 5: Performance of GA's portfolio compared to other strategies

3.2 Implementation

In this case, I applied a Genetic Algorithm to portfolio management replacing the weight of each stock with its own volatility, while the "value" of the stock is defined by the same price. Moreover, I have supported several attempts, every time changing the parameters (rebalancing frequency and temporal window within which to consider the performances of the actions). In each attempt, the portfolio built by the algorithm proved to be at least as good as the best randomly selected portfolios. This result must also take account of the fact that the GA was subject to an additional degree of freedom, as it also had to choose the actions.

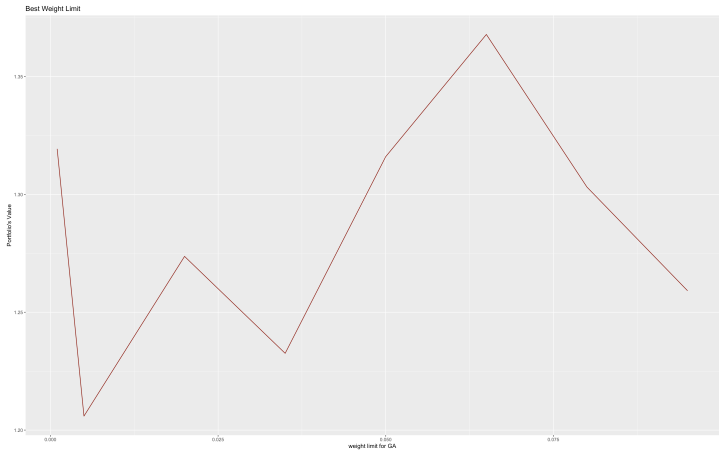

Figure 6: Performance of GA's portfolio compared to other strategies

At this point you can see how a portfolio built by a GA is able to produce a return at least comparable with the best achievable portfolios, however this does not justify how the knapsack problem is an extremely simplified interpretation. In fact, if the weight limit were raised, the GA would not hesitate to consider all 36 shares in the dataset as a single portfolio. This is due to the fact that the same weight limit is deliberately defined in a primitive way, as the sum of all the volatilities of the shares within the portfolio. However, this should be an incentive to redefine a more sophisticated version of the problem, so that the GA itself would be able to make a tighter selection and thus lead to a higher portfolio return.

References

[1] Cover, T.M.: Universal portfolios. Mathematical finance **1**(1), 1–29 (1991)

[2] Cover, T.M., Thomas, J.A.: Elements of information theory second edition solutions to problems (2006)

[3] Forrest, S.: Genetic algorithms. ACM computing surveys (CSUR) **28**(1), 77–80 (1996)

[4] Orval: Logopt: a journey in r, python, finance and open source (2012), https://optimallog.blogspot.com/2012/06/universal-portfolio-part-2.html

[5] Ross, K.W., Tsang, D.H.: The stochastic knapsack problem. IEEE Transactions on communications **37**(7), 740–747 (1989)

[6] Turner, D.A.: The knapsack problem & genetic algorithms (2020), https://youtu.be/MacVqujSXWE