On Optimal and Objective-Specific Selection of Sequential Offers

Kwa Jie Hao, Alessandro de Sanctis, Jan Rademacher, Guillam Bagaria

3/04/2018

Introduction

In this paper, we will explore situations which can be framed as an agent making decisions in a sequential way. In many circumstances, we will think of these decisions as receiving and selecting sequential offers. These problems can be split into two main sections: single items, and multiple items. In general, we explore solutions and algorithms which tell us how to select the offer we want.

For the single item case, there are well-known optimal policies to select the best offer to what is known as the secretary problem. We look into cases where instead of trying to find the best offer, we are content to settle for an offer that is "good-enough": accepting an offer that is within the top $100 \cdot (1-\tau)\%$ of offers of the true distribution. We devise a generally nonparametric solution which, without making too many assumptions on the true distribution of offers, allows us to accept with high probability an offer within our desired boundary.

This technique, however, breaks down when we are dealing with multiple items since it requires us to know, or make some assumption on the covariance structure between the items. For the multiple item case, we turn to multiple-armed bandit techniques, and in particular, the upper confidence bound algorithm, to help us make selections on offers for multiple items. We use the portfolio selection problem as an example and use data from the S&P500 to test our algorithm. Our algorithm manages to outperform slightly our chosen benchmark of an equally-weighted portfolio.

A Quantile Solution to a Variant of the Secretary Problem

Literature

Our chosen problem is based on the secretary problem, a famous problem involving optimal stopping theory. The problem is as follows: interview n candidates for a position sequentially. After each interview, you have to decide if you want to hire the candidate. If you do not accept the candidate after the interview, you will not be able to make an offer to the same candidate in the future (the stopping sets in this problem are non-absorbing). The goal is to maximize the probability of choosing the best candidate (Ferguson, 1989). The initial problem we're trying to solve here has a slightly different objective: we are trying to maximize the probability of choosing a candidate within the top $100 \cdot (1-\tau)\%$ of the true distribution of candidates.

There is a vast literature on the secretary problem and its variants. In particular, there exists a well-known and simple solution which performs very well(Bruss, 2000). THe policy states that, if we know in advance that we will receive a total number of N offers, we simply observe the first $\frac{N}{e}$ offers (let M be the best offer among the first $\frac{N}{e}$ offers), then choose the first offer after that larger than or equal to M. This gives a $\frac{1}{e}$ probability of obtaining the optimal solution, regardless of the size of N.

There are other related problems, such as the the hiring problem, which involves accepting more than one offer / hiring more than one candidate. In such cases, other methods like maximum hiring (hiring only if the candidate is better than everyone that's been hired), hiring above the current mean or the current median (Helmi & Panholzer, 2011) can be used. However, the setting is different since we are only accepting one offer, and not multiple offers.

Our approach to the secretary problem is similar to the known solution. However, it is novel in the sense that we are using a threshold that we learn from the first n^* data points.

Motivation

In most situations, one would like to maximize returns and try to obtain the best offer in our given time period. There are certain situations, however, where instead of trying to maximize the returns, the objective instead is to accept an offer within the top $100 \cdot (1 - \tau)\%$ of the true distribution. This happens in situations where people are satisfied with "good-enough" offers, perhaps because of some uncertainty associated with the situation. One example of this concept is the case of couple pairing, or finding a spouse. While perhaps an ideal spouse exists for most people, the costs of finding this ideal spouse is prohibitive and so most people are satisfied with someone who is "good-enough". Another example is when one is selling a house. While one would like to receive as much money as possible for the house being sold, there are other factors such as uncertainty, time, administrative, or personal constraints (for instance, needing the money from the sale urgently) which means that people are often happy to accept a good-enough offer for their house. Since the value of a house is relative to the markt's valuation of houses, there is no objectively high or low offer. Thus, it makes sense to say define a customer's objective as that of accepting a value that is within the top 10% of offers for example.

In this section, we analyze this problem by formulating it as a secretary problem and then use quantile estimation which tells us to measure the quantiles for certain time periods and then accepting the first offer after this time period which is above the estimated quantile. We can test this approach on real data, such as stocks, or simulated data, and see how it performs compared to other methods.

The main working assumption made is that there is a time-stationary true distribution for offers (distribution remains constant over time). If there is reason to believe that the distribution is changing over time, the model can be modified in a Markovian way so that only the past n offers or taken into account, or reset if the change is determined to be caused by a structural break (a sudden change in distribution as opposed to a dynamic change). However, note that the way in which we will be estimating the quantiles is non-parametric and distribution-free! We assume that there *is* a true distribution but makes no explicit assumptions on the parameters or nature of that distribution.

Quantile Estimation

The quantile of a distribution is defined (Hyndman & Fan, 1996) as:

$$Q(p) = F^{-1}(p) = \inf\{x : F(x) \geq p\}, \quad 0 < p < 1$$

According to Hyndman and Fan, the authors of the "quantile" package in R, there are 9 popular definitions of estimating sample quantiles. They evaluate the qualities of each estimator according to a list of 6 desirable properties for sample quantiles. Due to the complicated terminology, and for the sake of brevity, we will not cover the definitions of these estimators in our paper. In accordance with the recommendations made by Hyndman and Fan, we choose to work with the approximately median-unbiased estimator (definition number 8), which can also be defined independently of the underlying distribution, allowing us to make non-parametric estimations. Although we won't go into the definition here, we note that this estimator makes a linear interpolation when the quantile does not cleanly divide the number of points.

Method

Let Q represent the true quantile value and $\hat{Q}_\tau$ represent the sample quantile estimator, where τ represent the quantile and $\tau \in \mathbb{N}, \tau \in (0, 1)$. Let n be the sample size. The key observation to make is that the probability that a random variable is greater than its (say) 90th quantile is 0.1 **regardless** of the distribution. So the expected number of points larger than this quantile for a given n is Binomial(n,0.1). This allows us to put some bounds on the performance of our nonparametric quantile estimators with regards to sample size.

We will be using some concentration inequalities to arrive at our result. In particular, we make a few more assumptions on the distribution that we are estimating. Firstly, we assume the existence of a moment generating function for the true distribution of the item we are studying. Secondly, we assume that the density of the distribution is positive for the region of the quantile we are estimating.

Essentially, we want to guarantee that the sample quantile we have estimated is larger than or close to the real quantile with high probability. Or, $P(\hat{Q}_\tau > Q - \epsilon)$ where ϵ is small.

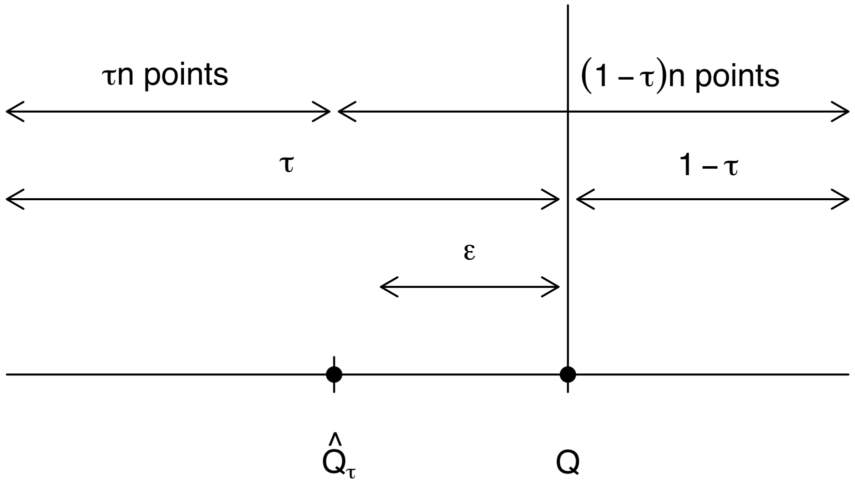


Figure 1

Note that by construction of the sample quantile, at $\hat{Q}_\tau$, there are approximately $\tau \cdot n$ points less than or equal to $\hat{Q}_\tau$ and $(1-\tau) \cdot n$ points more than the sample quantile. Therefore, if $\hat{Q}_\tau < Q - \epsilon$, this means that there are less than $(1-\tau) \cdot n$ points greater than $Q - \epsilon$ and thus more than $\tau \cdot n$ points less than $Q - \epsilon$. This can be seen clearly on the figure above. This means that we can formulate this as a binomial event.

$$P(\hat{Q}_\tau < Q - \epsilon) = P(\ Bin(n,\ P(x < Q - \epsilon)\ ) \geq \ \ \tau \cdot n)$$

Our next task is to find $P(x < Q - \epsilon)$. The true probability of a given point being larger than the true quantile Q is $1 - \tau$ and the true probability of being less than or equal ot the true quantile Q is τ. However, here we are interested in the true probability of a point being less than $Q - \epsilon$. First, denote the density of the unknown true distribution by $\psi(x)$. Then,

$$\begin{aligned} P(x < Q - \epsilon) &= \tau - \int_{Q-\epsilon}^{Q} \psi(x)\ dx \\ &\leq \tau - \epsilon\delta \end{aligned}$$

where we want $\psi(x) \geq \delta$. Here, we approximate the true density of the distribution in the region from Q to $Q - \epsilon$ by $\epsilon\delta$, a block with width ϵ and height δ, where ideally δ is the minimum density of the true distribution between Q to $Q - \epsilon$, or:

$$\delta = \min_x\ \psi(x),\ x \in (Q, Q - \epsilon)$$

At this point, we need to say something about δ and the true distribution $\psi(x)$. Although we have promised that this estimation will be nonparametric and distribution-free, to obtain values of δ which make sense, we can reference the quantiles of several popular and well-known distributions. Furthermore, notice that:

$$\lim_{\epsilon \to 0} \delta = \psi(Q)$$

Therefore, we need to look at the value of $\psi(Q)$, the density of the τ-th quantile, for several distributions. Often, we see that real world distributions have very different densities, which means that we will need to come up with more realistic values of δ. We will cover the selection of δ in the coming pages.

Since we propose that $P(x < Q - \epsilon) \leq \tau - \epsilon\delta$,

$$\begin{aligned}
P(\hat{Q}_\tau < Q - \epsilon) &= P(\ Bin(n,\ P(x < Q - \epsilon)\ ) \geq \ \ \tau \cdot n) \\
&\leq P(\ Bin(n,\ \tau - \epsilon\delta) \geq \ \ \tau \cdot n) \\
&= P(Bin(n,\ \tau - \epsilon\delta) - n(\tau - \epsilon\delta) \geq n\epsilon\delta)
\end{aligned}$$

From here, we can bound the probability using Hoeffding's inequality [CITE HERE]. This implicitly uses Chernoff bounding, which is why we assumed the existence of a moment-generating function for the true distribution. Using Hoeffding's inequality, we have:

$$\begin{aligned}
P(\hat{Q}_\tau < Q - \epsilon) &\leq e^{-2n(\epsilon\delta)^2} \\
&= \phi
\end{aligned}$$

This means that:

$$P(\hat{Q}_\tau \geq Q - \epsilon) \geq 1 - \phi$$

This means that for small ϕ, we have the sample quantile estimator to be larger than the true quantile minus a small error with high probability. Analyzing ϕ,

$$\begin{aligned}
e^{-2n^*(\epsilon\delta)^2} &= \phi \\
\text{Then } \ \epsilon &= \frac{1}{\delta}\sqrt{\frac{ln(\frac{1}{\phi})}{2n^*}} \\
\text{or } \ n^* &= \frac{ln(\frac{1}{\phi})}{2(\epsilon\delta)^2}
\end{aligned}$$

Therefore, for a given δ, and a given error level ϵ, it tells us the number of observations we need for the quantile estimator to be larger than or near the true quantile with probability $1 - \phi$. This also tells us that any offer that is larger than $\hat{Q}_{\tau,n^*}$, the sample quantile estimator for quantile τ after n^* samples, will be larger than $Q - \epsilon$ with high probability. In the context of house-selling, where we want to accept an offer within the top 10% of offers in the true distribution, it tells us that any offer that we receive after n^* offers that is larger than the estimated quantile $\hat{Q}_{0.9,n^*}$ is highly likely to be in the top 10% of offers from the true distribution. Observe also that the ln in the numerator helps to reduce the size of $\frac{1}{\phi}$, which allows us to know with greater certainty the accuracy of our estimator without paying too large of a cost.

The initial algorithm is then as follows:

1. Determine values of $\tau, \phi, \epsilon, \delta$, and the time period $t = 1, 2, ...T$ over which the algorithm will run.
2. Compute the value of the τ-th quantile when you have n^* points (simply collect the data before this point).
3. For every point that comes in after, if it is larger than or equal the estimate of the τ-th quantile, accept it. Otherwise, re-estimate the τ-th quantile with the new data point.
4. Repeat step 3 until a point larger than or equal to the estimated quantile arrives.

Analysis of Algorithm

Notice that the structure of the algorithm is roughly the same as the general solution to the secretary problem: observe the first m offers, then take the first offer larger than a benchmark derived from the first m offers. In the original solution, the benchmark was the highest offer among the first m offers, but in our version, the benchmark is the sample τ-th quantile.

The biggest challenge that we face in applying the algorithm is in the selection of the parameters ϵ and δ. Given the nonparametric nature of our estimation, Note also that the quantile τ affects the value of δ. Since we are talking about cases where we are hoping to receive an offer within the top $100 \cdot (1-\tau)\%$ of the true distribution, it implies that we deal with distributions where higher offers are more rare and thus the density is lower at higher quantiles. Therefore, the closer τ is to 1, the lower the value of the density, which in turn requires us to have a lower value of δ.

One way of choosing ϵ is to choose it as a fraction of the sample quantile. We want ϵ to be small so that the estimated quantile is close to the true quantile. However, the definition of small is relative. Therefore, this approach is reasonable since the size of ϵ is relative to the specific distribution from which the offers are coming from. For example, if you are selling a car which has a market value of \$20,000, then we might choose $\epsilon = 0.01\hat{Q}_\tau$, which might be a few hundred dollars. By pegging ϵ to the estimated quantile, we automatically adjust find the right order of magnitude for ϵ.

With this method of selecting ϵ, we have to modify our algorithm to recalculate ϵ at every step so that n^* is updated with every step. Again, once the number of iterations t exceeds n, we accept the first offer which exceeds our sample quantile.

The bigger problem that we face is the selection of δ. δ is intrinsically tied to the true distribution of the offers, which are trying not to make any major assumptions about. In most applications of this algorithm, there is usually prior information on the offers, such as offers made previously in a different time period. One can try to gain some useful information from these past values to choose δ. The rationale is that although the distribution from previous periods might be different, it is not unreasonable to assume that for the same setting, the underlying distributions are likely to be very similar.

Our solution to this is rather ad-hoc: firstly, we plot the density of the training data and compare its shape to common distributions, such as the normal distribution, the poisson, or the binomial. We choose the distribution class it seems closest to and measure the sample mean and variance of the training data to find an approximation for the value of δ. This is also a chance to use prior knowledge of the type of offer in question: for example, it is known that the distribution of stock prices is similar to the normal distribution but with fatter tails. We then simply calculate δ by measuring the sample mean and variance of the stock in question and check the density at the τ-th quantile of the normal distribution with those moments.

If the training data is large enough, we can use kernel density estimation to get a good idea of δ by estimating the density of the offers looking at past data (Chen, 2017), although this also requires some idea of the class of distribution of the data. This goes beyond the scope of this paper.

What happens if we end up choosing a bad value of δ? If the choice of δ is such that $\psi(X) < \delta$, then what that means is that instead of looking to accept an offer in the top $100 \cdot (1-\tau)\%$ with high probability, in actuality we will be calculating the sample quantile μ such that $\mu < \tau$ - the result is that our algorithm leads us to accept an offer in the top $100 \cdot (1-\mu)\%$ with high probability instead. This is not a big problem as long as μ and τ are not too far apart, which in turn relies on the choice of δ being not that much larger than the true value of δ.

Since there is some uncertainty with the choice of parameters, we can keep track of n^* and $\hat{Q}_\tau$ for multiple values of ϵ (as well as ϕ or δ if the user so desires) when implementing the algorithm.

Simulation

We conduct a simple test on this algorithm by using simulated data from both a normal distribution with mean $\mu = 25000$ and standard deviation $\sigma = 1000$, then a poisson distribution with lambda parameter $\lambda = 25000$. We will illustrate fully the steps taken with the normal distribution. Firstly, we generate a sample of 2000 data points, and use 500 for learning the shape and moments of the distribution.

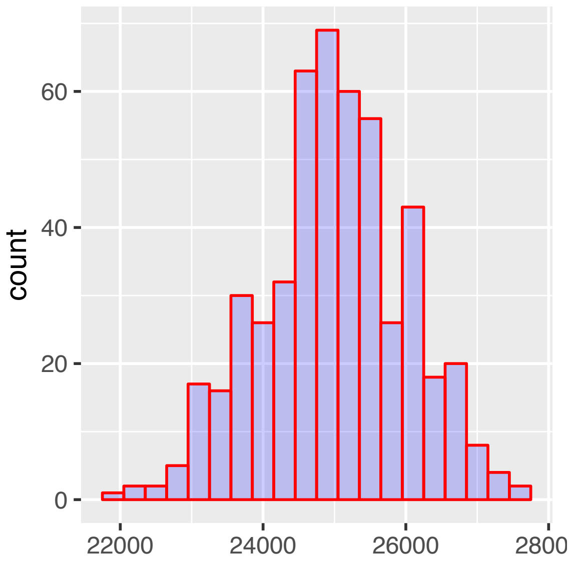


Figure 2

Then, we estimate the sample mean and variance and calculate the value of δ using a normal distribution since the histogram looks approximately normal. Running the algorithm, we can track the values of n^* and ϵ as they update.

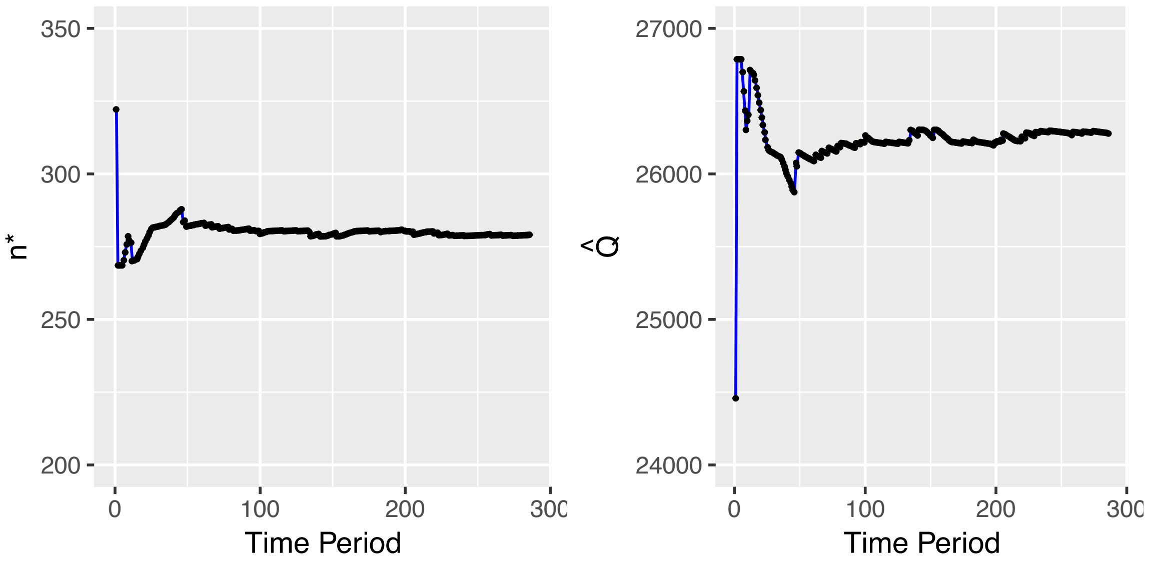


Figure 3

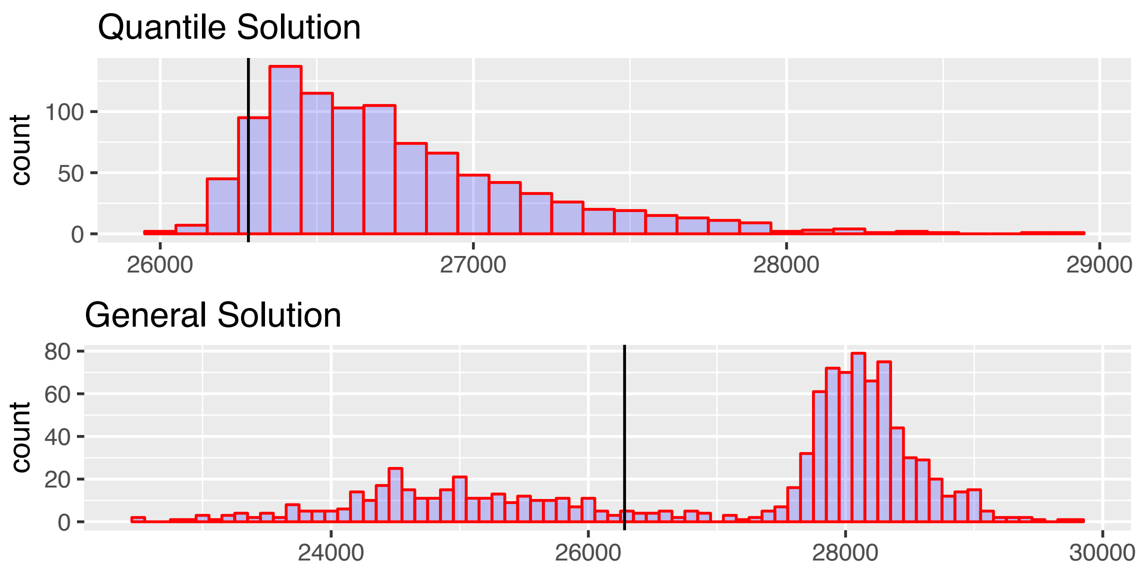


Figure 4

We find that approximately 92% of the accepted offers are larger than the value of the true quantile. We see that the fluctuation of the value of $\hat{Q}$, the estimated quantile, and thus also ϵ which is calculated as a fraction of $\hat{Q}$, quickly dies off and the value of $\hat{Q}$ settles down to fluctuate around a mean after approximately 100 time periods. There seems to be less fluctuation for the value of n^* which also remains relatively constant after 100 time periods.

We compare our accepted offers to that from the general solution to the secretary problem. Only approximately 65% of accepted offers are larger than the value of the true quantile. However, there is a higher concentration of high offers. This tells us that our algorithm does indeed perform well and serves a different function to the odds algorithm, which tries to obtain the best offer as opposed to just the top $100 \cdot (1-\tau)\%$ of offers.

We do not include figures for the tests with the simulated Poisson distribution due to a lack of space. However, they reflect similar results which can be found in the code section of this paper.

One thing to note is that when the variance of the distribution seems to be large, the value of δ becomes very small. This poses a challenge for our method since the number of observations required, n^*, might become prohibitively large. However, for many real-world applications, we can dismiss offers which are extremely low to reduce the variance. Returning to the secretary example, most poor candidates would have been removed by some sort of pre-screening process before the interview stage. Therefore, the variance in quality of candidates who make it to the interview stage is relatively small.

Extensions

When there are multiple offers to accept, or multiple secretaries to hire, we can just keep the algorithm going. This will ensure that all offers accepted are within the top $100 \cdot (1-\tau)\%$ of the distribution with high probability.

Multiple Items

Unfortunately, the philsophy behind this technique breaks down when it comes to analyzing multiple, possibly correlated, items. With multiple correlated items, some assumption would have to be made about the multi-variate covariance structure of the problem, which we do not currently assume.

When there is some constraint on the decision such that the decision on multiple items has to be made collectively, we can utilize the same technique to make decisions. For example, consider a stock index. The decision to invest in an index is equivalent to the decision to invest in all of the stocks in the index at once. Thus, regardless of the number of components involved, one need only apply the same analysis to the stock index's price and ignore the components in making a decision on whether or not to invest in the index.

To overcome the limitations of our single-item solution, we turn to multiple-armed bandit techniques which allow us to deal with the correlation between items.

Bandit problems

Bandit problems refer to the class of problems of making decisions under uncertainty, a serious challenge in machine learning. In particular, consider a framework in which an agent sequentially makes decisions on which action to take, observes the reward or penalty coming from the action taken, and aims to optimize the total reward or penalty over a period of time. When the decision is made with respect to a variety of choices, this framework goes under the name of the multi-armed bandit problem. This concept refers to the situation in a casino, when a gambler is faced with one or a variety of slot machines (the armed bandits) and he has to decide which machines to play and for how long in order to maximize his returns. This problem is very similar to when an investor has to choose an optimal portfolio and can be applied to the sequential setting in portfolio choice theory.

Framework

Hereafter, we present the framework of the stochastic multi-armed bandit problem. In this setting the rewards come from unknown distributions which are independent and identically distributed with respect to the actions previously taken.

As explained by Bubeck and Cesa-Bianchi [2012], the difficulty of the stochastic version of the problem comes from the trade-off between exploration and exploitation. Indeed, the learner has to decide whether exploiting the current knowledge about the distributions of the rewards to focus on the most profitable arms, or exploring further the other arms in order to infer with better precision which are the arms which yield the highest rewards.

For example, consider a two-armed bandit problem in which the learner already chose the two arms five times each and observed the following payoffs:

Arm A: 0, 1, 0, 1, 0

Arm B: 1, 0, 0, 0, 0

From choosing arm A the learner got two times the payoff he got by choosing arm B. But what is the possible conclusion? It may be that arm A happens to give positive rewards more often than arm B, in this case it would make sense to ignore arm B and to keep choosing arm A for the remaining time. However, it is also possible that maybe five trials are not enough to establish that arm B provides worse payoffs, and it could be useful to choose arm B few more times in order to have a better idea of its true potential.

In the general problem, consider a learner interacting with an environment where for each action he takes he will get (observe) a reward. Interactions take place in a sequential way for $t = 1, 2, \dots, n$ times according to the following steps:

- The learner chooses an action $A_t \in \mathcal{A}$, where $\mathcal{A} = \{1, 2, \dots, K\}$ denotes the set of available actions/arms at step t.
- The environment responds by giving the learner a random reward $X_t \in \mathbb{R}$ distributed according to P_{A_t}.

The goal of the learner is to maximize the some of the rewards $S_n = \sum_{t=1}^{n} X_t$ that he obtains across the whole period.

Among the available actions $A_t \in \{1, 2, \dots, K\}$, let us define the expected reward gained when action A_t is chosen as $\mu_k = \int_{\mathbb{R}} x P_k \, dx$, assuming that μ_k exists and is finite (e.g. P_k should not have heavy tails). Consider $\mu^* = \max_k \mu_k$ as the maximal reward and define $\Delta_k = \mu^* - \mu_k$ as the expected amount the learner loses by taking action k available at time t compared to taking the optimal action, also known as the immediate regret of action k. Let us count how many time an action k has been taken until round n as $T_k(n) = \sum_{t=1}^{n} 1_{A_t = k}$, including the choice of that round.

The total expected regret is defined as $R_n = n\mu^* - \mathbb{E}\left[\sum_{t=1}^{n} X_t\right]$. By definition of μ^*, this is the difference between the maximal amount the learner could have received knowing the distributions generating the rewards and what he is expecting to make.

It can be shown that the total expected regret may be rewritten as $R_n = \sum_{k=1}^{K} \Delta_k \mathbb{E}[T_k(n)]$, which shows that in order to keep the total regret small, the learner should minimize a weighted sum of expected action counts where the weights are given by the immediate regrets.

The Upper Confidence Bound Algorithm

Given the trade-off between exploration and exploitation, the solution that we are going to describe is based on the idea to simultaneously perform both exploration and exploitation.

The idea is based on the principle of optimism in the face of uncertainty, which is to take the decision over a set of actions as if the environment is as nice as is plausibly possible. In our case, even if the learner does not have a perfect information on the expected payoff of each arm, based on previously observed data, he can construct a set of plausible guesses. If the optimism turns out to be unjustified sufficiently often, then the learner should correct his estimate on the expected payoff of the arm chosen.

Let $X_1, X_2, \dots, X_n$ be the independent and identically distributed rewards associated to one arm such that $\mathbb{E}[X_i] = 0$ and define $\hat{\mu} = \sum_{t=1}^{n} X_t$. In particular, let us assume that the rewards are 1-subgaussian, that is $\mathbb{E}[exp(\lambda X)] \leq exp(\lambda^2/2)$, for any $\lambda \in \mathbb{R}$. The following holds

$$P(\hat{\mu} \geq \epsilon) \leq exp(-n\epsilon^2/2).$$

By setting $exp(-n\epsilon^2/2) = \delta$, the above inequality can be rewritten as

$$P\left(\hat{\mu} \geq \sqrt{\frac{2\log(1/\delta)}{n}}\right) \leq \delta.$$

When a decision should be taken at time t, for every arm the learned has computed a sample mean $\hat{\mu}_i(t-1)$ based on the $T_i(t-1)$ times it has already chosen arm i. Hence, a good plausible candidate for the largest estimate of the expected reward of arm i is

$$\hat{\mu}_i(t-1) + \sqrt{\frac{2\log(1/\delta)}{T_i(t-1)}},$$

where a small δ denotes a more optimistic algorithm. Hence, the algorithm will choose the arm i that maximizes the above quantity. More formally, the Upper Confidence Bound algorithm will choose $A_t \in \{1, 2, \dots, K\}$ such that

$$A_t = \begin{cases} \text{argmax}_i \left(\hat{\mu}_i(t-1) + \sqrt{\frac{2\log(1/\delta)}{T_i(t-1)}}\right), & \text{if } t > K; \\ t, \quad \text{otherwise} \end{cases}$$

where it is required to have visited every arm at least once. It is possible to see that an arm tend to be explored over the others if it has already performed well, i.e. if $\hat{\mu}_i(t-1)$ is large, or if it has not been explored often, i.e if $T_i(t-1)$ is low.

The value of $1-\delta$ is called the confidence level, and different values of it define different variations of the algorithm. It is often the case that δ is chosen to be time dependent.

Using the definition of total expected regret from the previous session, it can be proved that it is bounded from above by

$$R_n \leq \sum_{i:\Delta_i>0} \left(\Delta_i + \frac{1}{\Delta_i}(8\log(1/\delta) + 8\sqrt{\pi \log(1/\delta)} + 28) \right).$$

Multiple Bandit for Portfolio Selection

Motivation

Portfolios are an important tool in modern finance. By investing in multiple stocks at the same time, one can enjoy the high returns of stock markets (compared to investing in bonds) while reducing the overall investment risk (compared to investing into a single stock). A vast group of agents makes use of portfolios, particularly pension and mutual funds, insurance companies, and corporate risk management units.

Portfolio choice (i.e. which stocks to choose for a portfolio) is a key concept, which has been studied in great detail due to its importance for the financial industry. Much goes back to the 1950s and Harry Markowitz' work on portfolio optimization [Markowitz, 1952]. According to his theories, investors are willing to take a certain amount of risk (risk appetite) and choose the portfolio that promises highest returns for this level of risk. The highest-return portfolios for each level of risk are called *efficient portfolios* and taken together they form the *efficient frontier* - the curve along which investors should choose an optimal portfolio that corresponds to their risk appetite.

There are numerous approaches to actually choose optimal portfolios (e.g. mean-variance analysis, linear programming, dynamic programming), but in recent years especially machine learning algorithms have become more popular. Particularly, because they offer an efficient way to deal with large amounts of data and because they can be well applied to sequential data and real-time streaming (i.e. online learning) [Shen et al., 2015].

One way to implement sequencial learning is to use the concept of multi-armed bandits explained in the previous section. There are a few caveats though:

- classical bandit problems assume that the returns between different arms/machines are identically distributed for each arm, and that returns are independently distributed across arms - this is generally not the case for stock returns, as returns might be correlated across different stocks and distributions might also change over time
- solutions to the classical multi-armed bandit problem usually identify one arm/machine which is chosen to be exploited - in portfolio optimization one is usually interested in a diversified portfolio to pool risks, i.e. to choose several stocks at the same time (there have been some combinatorial multi-armed bandit approaches to portfolio selection, but everything boils down to binary choices and an equal distribution among arms - this is sub-optimal as the main focus of portfolio choice is to get the correct weighting)

Shen et al. [2015] come up with an orthogonal portfolio bandit approach to mitigate these shortcomings when directly applying the classical bandit problem. Their approach contains the following aspects:

- principal component analysis to create a set of orthogonal portfolios to choose from
- Sharpe ratio as risk-adjusted reward function in upper confidence bound to identify optimal portfolios
- splitting portfolios into an active and passive group to make a combination which has overall a low risk

In the following we describe Shen et al.'s paper in greater detail. Namely, we discuss the algorithm, its assumptions, implement it in our own dataset, and compare its performance to a benchmark of equally weighted portfolios.

Assumptions

We assume an investment environment which is

- friction less (there are no costs of doing a transaction)
- self financing (there is no withdrawal or infusion of money from outside)
- discrete-time (we only consider fixes points in time, and not what happens within)
- finite horizon (we do not consider infinite horizons)

and where investors have access to n risky assets (a risk free asset does not exist).

Data

We take 71 return series from stocks listed in the SP500 from January 2000 to January 2018, yielding 4500 observations for each series. The returns are calculated as $S_{k,i}/S_{k-1,i}$, where $S_{k,i}$ is the adjusted closing price of stock i on date k.

Algorithm

The algorithm takes as input the return series and calculates the optimal portfolio weigths $\boldsymbol{\omega} = (\omega_{k,1}, \omega_{k,2}, ..., \omega_{k,i}, ..., \omega_{k,n})$. The sum of all portfolio weights has to add up to 1. Short-selling is possible, implying that $\omega_{k,i}$ may be negative.

The optimal weights are calculated for each time in a fixed horizon m. At each time interval, the following steps are performed to calculate the portfolio weights:

1. **Calculate the expected returns $\mathbb{E}[\boldsymbol{R}_k]$ and covariance matrix of returns $\boldsymbol{\Sigma}_k$:**
   - $\mathbb{E}[\boldsymbol{R}_k] = \mathbb{E}[(R_{k,1}, ..., R_{k,i}, ..., R_{k,1})^T] = [\frac{1}{\tau} \sum_{j=k-\tau}^{k-1} R_{j,1}, ..., \frac{1}{\tau} \sum_{j=k-\tau}^{k-1} R_{j,i}, ..., \frac{1}{\tau} \sum_{j=k-\tau}^{k-1} R_{j,n}]$, where
     - $R_{k,i}$ is the return of asset i at time k, calculated as $S_{k,i}/S_{k-1,i}$, where $S_{k,i}$ is the adjusted closing price of stock i on date k
     - τ is the number of observations per series taken for learning(i.e. exploration period)
   - $\boldsymbol{\Sigma}_k$ is the covariance of the matrix of asset returns during the last τ observations, calculated using the Pearson correlation coefficient
2. **Decompose the covariance matrix of returns $\boldsymbol{\Sigma}_k$ by means of a principal component decomposition:**
   - $\boldsymbol{\Sigma}_k = \boldsymbol{H}_k \boldsymbol{\Lambda}_k \boldsymbol{H}_k^T$, where
     - $\boldsymbol{\Lambda}_k$ is the diagonal matrix with the eigenvalues $(\lambda_{k,i})$ of $\boldsymbol{\Sigma}_k$ on the diagonal and in decreasing order (note that all eigenvalues are larger than zero)
     - $\boldsymbol{H}_k$ is the orthogonal matrix with the eigenvectors $(H_{k,i})$ corresponding to the eigenvalues in $\boldsymbol{\Lambda}_k$ in the corresponding order from left to right in the columns
   - **Note that:** (a) every eigenvector corresponds to a particular portfolio, (b) all portfolios are orthogonal (by construction), (c) the portfolios are the arms of the bandit in this setting, and (d) every eigenvalue corresponds to the variance of the respective portfolio
3. **Renormalize the matrices of the principal component decomposition:**
   - $\widetilde{\boldsymbol{H}}_k = (\widetilde{H}_{k,1}, ..., \widetilde{H}_{k,i}, ..., \widetilde{H}_{k,n})$, where
     - $\widetilde{H}_{k,i} = \frac{H_{k,i}}{H_{k,i}^T \mathbf{1}}$
     - $\mathbf{1}$ is a column vector of length n and ones in every position
   - $\widetilde{\boldsymbol{\Lambda}}_k = \widetilde{\boldsymbol{H}}_k^T \boldsymbol{\Sigma}_k \widetilde{\boldsymbol{H}}_k$
   - **Note that:** this step was necesasary to meet the condition that the weights of each portfolio sum up to 1

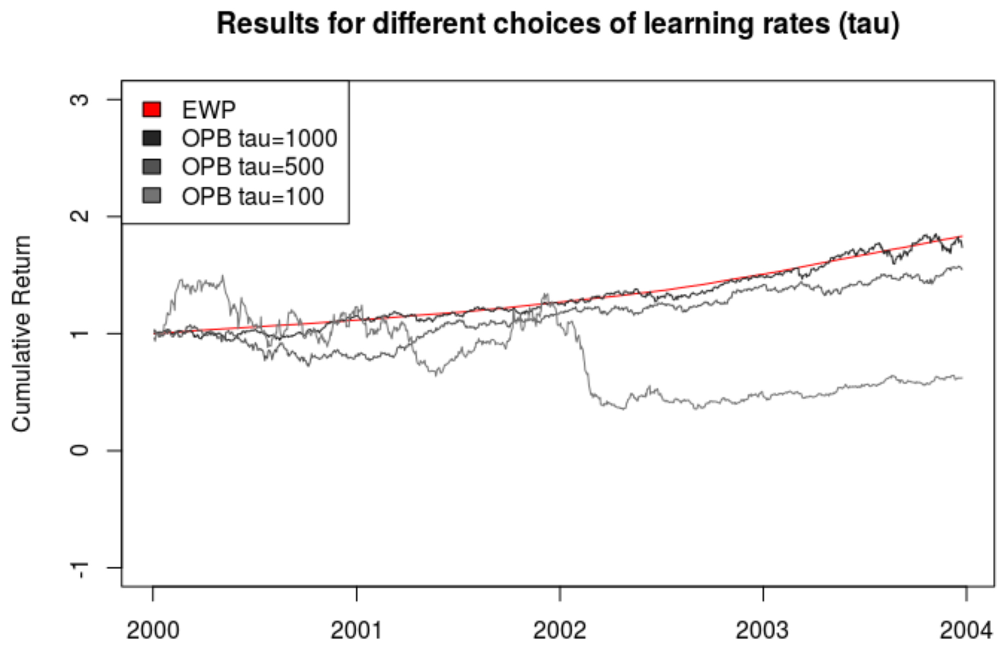
4. **Compute Sharpe ratio of the portfolios**
   - $SR_{k,i} = \frac{\mathbb{E}[\widetilde{H}_{k,i} R_{k,i}]}{\sqrt{\tilde{\lambda}_{k,i}}} = \frac{H_{k,i}\mathbb{E}[R_{k,i}]}{\sqrt{\lambda_{k,i}}}$
   - **Note that:** the Sharpe ratio is just a risk adjusted reward, where the expected portfolio return is divided by the portfolios volatility
5. **Incorporate one-sided confidence bound into reward function:**
   - Optimal portfolio: $i_k^* = argmax_{i=1,\dots,n} SR_{k,i} + \sqrt{\frac{2\log(k+\tau)}{\tau + k_i}}$, where
     - τ is the number of observations per series taken for learning(i.e. exploration period)
     - k_i is a counter accounting for how often portfolio i has been selected so far during the times of the fixed horizon m
6. **Select the optimal portfolio both in the set of passive portfolios (first l) and the set of active portfolios (last n-l):**
   - Choose l based on elbow plot of eigenvalues
   - **Note that:** this idea goes back to numerous empircial studies stating that a large part of market fluctuation can be expressed by very few eigenvectors (usually between 3-5), which are referred to as the systemic factors in the market. The next $n - l$ eigenvectors can be regarded as idiosyncratic risks which can be explored to get higher returns. A combination of both types allows for (a) following the market and receiving passive returns, and (b) adding the chance of extra returns from a small factor [Shen et al., 2015; Meucci, 2009; Bai and Ng, 2002, Grinold and Kahn, 2000].
7. **Compute the optimal mixture weight to minimize the overall variance:**
   - $\theta_k^* = argmin_{\theta_k^*} \lambda_{k,p} = \frac{\tilde{\lambda}_{k,i_k^*}}{\tilde{\lambda}_{k,i_k^*} + \tilde{\lambda}_{k,j_k^*}}$, where
     - θ_k^* is the optimal portfolio mixture weight obtained from minimizing the equation of the total variance $\lambda_{k,p} = \theta_k^2 \tilde{\lambda}_{k,j_k^*} + (1-\theta_k)^2 \tilde{\lambda}_{k,i_k^*}$
8. **Compute the optimal portfolio weight $\boldsymbol{\omega}_k$:**
   - $\boldsymbol{\omega}_k = (1 - \theta_k^*)\widetilde{H}_{k,i_k^*} + \theta_k^* \widetilde{H}_{k,j_k^*}$

Results

We assess the performance of this orthogonal portfolio bandit (OPB) algorithm by comparing it to a benchmark of an equally weighted portfolio (EWP). The comparison is made based on having 1 USD starting capital and observing its cumulative earning throughout 1000 trading days (four years). The weights of the EWP portfolio are kept the same throughout the entire time period whereas the OPB portfolio are adjusted every day according to the described algorithm.

Below we show the development of the OPB portfolio for different choices of τ and for two time horizons. The parameter indicating the amount of passive portfolios l was determined by an elbowplot and set to 1.

We observe that the OPB portfolio can perform well, if the learning rate (τ) is set large enough. From 2000-2004 it performs simlarly to the EWP bench mark, but from 2008-2012 it siginificantly outperforms the EWP benchmark.



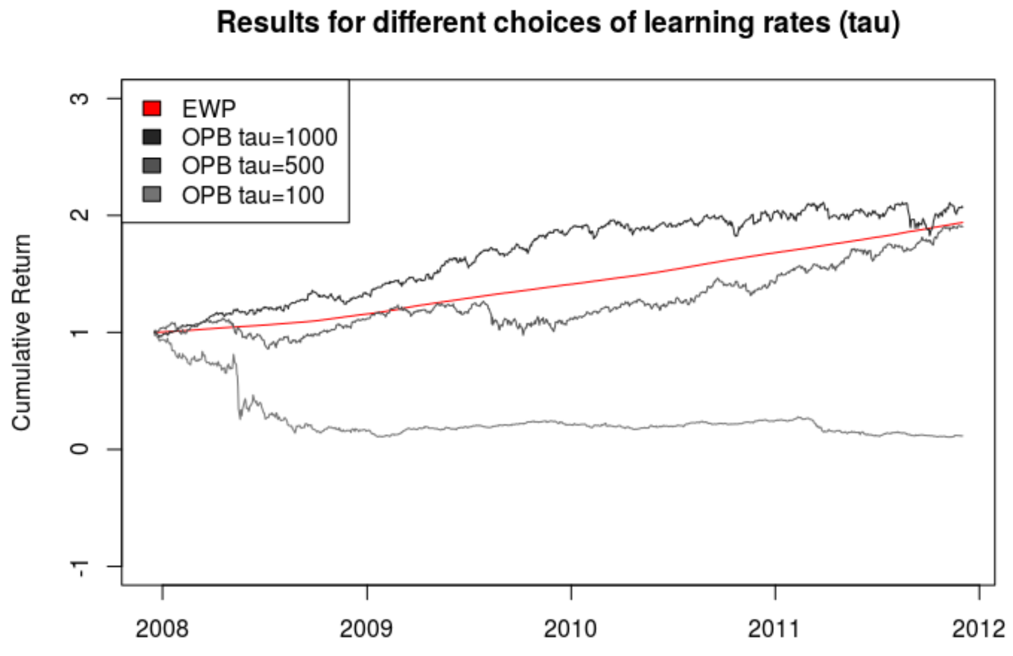


Figure 5

Conclusion

Our study finds that multi-armed bandits can be a useful way to think about portfolio selection problems. Together with other machine learning tools, they enable professionals to deal with the large amounts of financial data available in an efficient away. Moreover, they are particularly suited for deal with sequential data and online learning settings.

Our results also show that, given the right input parameters, an orthogonal portfolio bandit algorithm can perform slightly better than an equally weighted portfolio algorithm. The improvement is, however, not large enough to justify the portfolio switching/transaction cost, which might be a substantial drawback to applying this algorithm in practice.

References

Ferguson, Thomas S. Who Solved the Secretary Problem?. Statist. Sci. 4 (1989), no. 3, 282–289. doi:10.1214/ss/1177012493. https://projecteuclid.org/euclid.ss/1177012493

Bruss, F. Thomas. Sum the odds to one and stop. Ann. Probab. 28 (2000), no. 3, 1384–1391. doi: 10.1214/aop/1019160340. https://projecteuclid.org/euclid.aop/1019160340

Helmi, A., Panholzer, A.: Analysis of "hiring above the median": a "Lake Wobegon" strategy for the hiring problem. In: Proceedings of the ACM-SIAM Meeting on Analytic Algorithmics and Combinatorics (ANALCO 2012), pp. 75–83 (2012)

Hyndman, R.J. and Fan, Y. (1996) Sample Quantiles in Statistical Packages. American Statistician, 50, 361-365. http://dx.doi.org/10.2307/2684934

Chen, Y.-C. "A tutorial on kernel density estimation and recent advances." arXiv:1704.03924 (2017)

Bai, J., & Ng, S. (2002). Determining the number of factors in approximate factor models. Econometrica, 70(1), 191-221.

Bubeck, S. and Cesa-Bianchi, N. (2012). Regret analysis of stochastic and nonstochastic multi-armed bandit problems. Foundations and Trends in Machine Learning.

Grinold, R. & Kahn, R. (2000). Active portfolio management. McGraw Hill New York, NY, 2000.

Markowitz, H. (1952). Portfolio selection. The journal of finance, 7(1), 77-91.

Meucci, A. (2009). Risk and asset allocation. Springer Science & Business Media.

Shen, W., Wang, J., Jiang, Y. G., & Zha, H. (2015, July). Portfolio Choices with Orthogonal Bandit Learning. In IJCAI (Vol. 15, pp. 974-980).