Online Portfolio Selection using a new stochastic Multi-Armed Bandit Algorithm

Boby Chaitanya Villari[1*] **and Mohammed Shahid Abdulla**[2]

[1*]Doctoral Student, IT & Systems Area, Indian Institute of Management Kozhikode, Kerala, India

[2]Associate Professor, IT & Systems Area, Indian Institute of Management Kozhikode, Kerala, India

Abstract

Online Portfolio Selection Problem is a sequential decision-making problem where a decision-maker, using the information on assets available till that time period, repetitively selects a portfolio over available assets maximizing a long-term return that must be calculated at the end of each time period. Typically, multiple assets need to be 'explored' and a profit maximizing asset be 'exploited', hence this problem falls into explore-exploit class of Machine Learning based decision-making problems. Multi-Armed Bandit algorithms suit well to such explore-exploit scenarios, and there are instances in extant literature where these algorithms were applied to the same. In this work, we employ a newly-proposed Multi-Armed Bandit algorithm named effSAMWMIX to solve a naïve portfolio selection problem which is a variant of Online Portfolio Selection problem. In a naive portfolio problem, all the weight in resources for a time period is invested in a single asset. We compare the performance of effSAMWMIX vis-a-vis other Multi-Armed Bandit algorithms such as KL-UCB, Thompson sampling as also the benchmark Buy & Hold strategy. We tested the algorithms on real-world market datasets (the Fama-French FF48, FF100 and ETF139) as well as simulated datasets based on parameters drawn from real-world indices. We report our results where effSAMWMIX has achieved better cumulative wealth and Sharpe Ratio when used as a naive portfolio algorithm.

Keywords: Online Portfolio Selection Problem, Multi-Armed Bandit, Geometric Brownian Motion, effSAMWMIX, KL-UCB, Thomson Sampling, Naive Bandit Portfolio

1. Introduction

Decision making under uncertainty has always been a challenge - so much so in the case of Online Portfolio Selection Problem (OPSP). An Online Portfolio Selection Problem (OPSP) (Borodin & El-Yaniv, 2005; Dannoura & Sakurai, 1998; Fiat, 1998; Li & Hoi, 2014; Mohr & Schmidt, 2013; Schmidt, Mohr, & Kersch, 2010) is a sequential decision-making problem, where the decision-maker (Agent henceforth) must select a portfolio given a set of assets, with an aim to maximize a long-term return or reward. OPSP often encounters great deals of uncertainty due to the changing economic and political environments (Kumar & Garg, 2012; Merton, 1969). The rapid availability of price information, especially in Internet-based modern-day economies and indices, creates the need for fast portfolio selection algorithms based on the available albeit limited information. OPSP could require simultaneous optimization and the best asset (choice) identification. OPSP thus requires optimization of a suitable investment metric at every purchase decision time to identify the best choice of an asset to invest in.

In solving any Portfolio Selection Problem (PSP), the investor decides on a strategy to allocate the available (but finite) wealth among the available choice of assets. Every asset is a diverse investment opportunity and the realization of the asset allocation strategy builds a portfolio. An asset is termed risky if the prices of the asset are uncertain and such riskiness needs to be incorporated into the portfolio allocation process. The time between any two portfolio allocation decisions is called a period. If there is only one decision during the whole investment period, it is called a Single-Period PSP. A multi-period PSP requires

***Corresponding Author:**
Boby Chaitanya Villari, Indian Institute of Management Kozhikode, Kerala, India,
Email: bobycv06fpm@iimk.ac.in, Tel.: +91 8943686199

sequential decision-making over the time horizon of investment where the investor needs to stay very actively and thus is proposed as an online decision-making problem. Investor's role, during the decision making, is to optimize an objective function, which could be the Return on Investment (RoI) or Risk of losing wealth (Risk) or a combination of both (Risk & Return). Thus, portfolio decision making could also involve the management of Risk and maximize the RoI. The following section briefly discusses the OPSP and introduces a Machine Learning (ML) perspective of the OPSP problem.

Extant literature addresses OPSP primarily in two ways. The first one considers risk management to be factored into the objective function whose performance measure is quantified by Cumulative Wealth (CW). CW is the terminal wealth obtained from the portfolio at the end of a multiperiod investment horizon (Li & Hoi, 2014). The objective function considers the net risk of the asset choice decisions with respect to the CW that is sought to be maximized (e.g. maximize the Sharpe Ratio). Thus, this closed-form objective function technique operates based on this innate risk-based decision to measure the quality of its performance. Such performance measures are seen in seminal work such as (Markowitz, 1952; Mossin, 1966) and the more recent (Lisi, 2011; Rockafellar & Uryasev, 2000). These ideas are characterized by building statistical models of the asset prices in the market. The input to these statistical models requires a forecasting model in the form of an equation (DeMiguel, Martín-Utrera, & Nogales, 2015). The forecasting model, in turn, requires a calibration based on historical data of asset prices or market capitalization data (Fama & French, 1992).

The second way to address OPSP is based on utilizing the modern-day computing infrastructure along with intelligent ML techniques that include Neural Networks or RL Algorithms (Shen, Wang, Jiang, & Zha, 2015). ML algorithms are solely based on the empirical observations motivated by dynamic rise and fall of the asset prices. Algorithms (or Strategies) like Follow-the-Winner (Agarwal, Hazan, Kale, & Schapire, 2006; Li & Hoi, 2014), Follow-the-Loser (Li & Hoi, 2012) etc., are a couple of those which make use of such dynamic price changes. In brief, an ML Algorithm's approach to OPSP is to concretely explore the available information of past asset prices and based on its indigenous technique, provide a suggestion as to how the portfolio allocation be done for the next period. The algorithm typically intends to maximize the cumulative wealth at the end of the multi-period investment horizon.

In this research work, we propose and demonstrate a version of effSAMWMIX (proposed in our working paper (Villari & Abdulla, 2017)) referred henceforth as NBP-effSAMWMIX. Using this NBP-effSAMWMIX, we build a Naive Bandit Portfolio (NBP) algorithm like (Shen et al., 2015) and compare the same with NBPs that implement existing Stochastic Multi- Armed Bandits (SMAB) like UCB1 (Auer & Ortner, 2010), KL-UCB (Garivier & Cappé, 2011) and Thompson Sampling (Agrawal & Goyal, 2012; Kaufmann, Korda, & Munos, 2012; Thompson, 1933). We address the NBP versions of UCB1, KL-UCB and Thompson Sampling algorithms as NBP-UCB1, NBP-KLUCB and NBP-TS respectively. In the following sections we review SMAB algorithms (referred as SMABs) and describe how we constructed an NBP algorithm using such SMABs. To analyze the performance of these OPSP strategies, we implemented the proposed NBP-effSAMWMIX algorithm on a synthetic data set obtained by simulating stock prices using Geometric Brownian Motion (GBM) (Marathe & Ryan, 2005). We followed these with similar experiments on real-world market data (standard datasets) obtained from sources reported in literature (Bruni, Cesarone, Scozzari, & Tardella, 2016). Li & Hoi compared the performance of ML based OPSP algorithms against 'Buy & Hold' strategy which is a benchmark strategy (Li & Hoi, 2014). We also, in this work, compared the performance of above mentioned SMAB algorithms with that of 'Buy & Hold' strategy in order to validate these ML algorithms against a baseline strategy.

2. Stochastic Multi-Armed Bandits & The Naive Bandit Portfolio Algorithm

2.1 Stochastic Multi-Armed Bandit (SMAB) problem

A Multi-Armed Bandit (MAB) problem is a sequential decision-making problem which spans over iterations of decision-making horizon H, with decision made at indices t ∈ {1,2,...,H}. In each round (or iteration) t the decision maker chooses an action a at from among a set of K fixed action choices that are available and obtains a noisy reward X_t^a which is always bounded by [0,1]. In MAB terminology the words Choice, Action and Arm are used alternatively. Here X is the reward obtained by the MAB algorithm for choosing arm a in iteration t. The act of choosing an action is called pulling an arm as each available choice is an arm for a Multi-armed Bandit. The choice of which 'arm' to 'pull' is based on a goodness function of each arm, that is also updated for each arm as t proceeds towards H. The cumulative reward of a MAB algorithm is thus given by

$$C = \sum_{t=1}^{H} E\left(X_t^{a_t}\right) \qquad (1)$$

where a_t is the chosen action in iteration t. Of all the K action choices, say the choice a^* that could give the best possible reward is known (somehow, say through an oracle policy). We denote the reward obtained by pulling this arm to be X_t^* at any iteration t, thus the maximum expected reward for this oracle policy is

$$O = \sum_{t=1}^{H} E(X_t^*) \qquad (2)$$

The objective is to maximize the cumulative reward C and obtain a value as close to the maximum expected reward O as possible. Thus a MAB objective function tries to minimize the Regret ($\overline{R} = O - C$), which is the difference between the highest possible expected reward and the expected reward obtained by the algorithm (Lai & Robbins, 1985). Any MAB operates under certain assumptions on reward distributions {Xta}. For an SMAB, the rewards of each arm a ∈ {1,2...,K} accrue with a probability distribution v^a on [0,1]. Any indication of ν^a is unknown to the SMAB. The rewards $\{X_t^a\}$ from arm a are assumed to be independent and identically distributed (i.i.d) across the horizon of decision-making iterations i.e., $1 \leq t \leq H$ and are independent of reward distributions of other arms. There are a few other MAB settings like the adversarial MAB (Auer, Cesa-Bianchi, Freund, & Schapire, 1995) where the environment chooses the rewards to minimize the C of the algorithm, however we deal with SMABs where the rewards follow the i.i.d. assumption stated above. This limitation of reward being bounded between [0,1] is a requirement for applying SMAB algorithms. In real-world scenario where the stock returns could be any number between [-∞,∞], the returns are normalized to be between [0,1] without losing any information on the same.

2.2 The Naive Bandit Portfolio algorithm

A Naive Bandit Portfolio algorithm uses an SMAB as a decision-making engine. We take the term 'naïve' from (Shen, Wang, & Ma, 2014), since the algorithm does not decorrelate (or take any decision based on) any possible correlations between the K assets. The inputs to the SMAB are the time horizon H, number of available arms K (where each arm represents an asset), the time period between decisions is Δt which could be a day for daily returns or a week for weekly returns. The random variable X_t^a is the reward obtained when an asset a is chosen in decision period t and W_t is the cumulative wealth obtained until the iteration t. We explain later the precise form that the return X_t^a takes, but currently we assume that asset a has a price process $\{\rho_t^a\}_t^H = 1$ Here, ρ_t^a is the price of an asset a at iteration t. Then the gross return on asset a in iteration t is denoted by R_t^a and is obtained as $R_t^a = \frac{\rho_t^a}{\rho_{t-1}^a}$. The gross returns vector at t over the time for a portfolio with unit investment in each of the K assets can be written as $R_t = (R_t^1, R_t^2, \ldots .. R_t^K)'$. Similarly, price vector ρ_t represents the prices of all available assets at time iteration t while 1<t<H as mentioned before. In a typical scenario where prices are being simulated using GBM, ρ_0 has all elements such that $\rho_0^a = 1$.

The portfolio investment decision made is determining of weights proportionate to which investment will be made in K assets. Thus, the weights vector required at time t is represented as $\omega_t = (\omega_t^1, \omega_t^2, \ldots . \omega_t^K)'$ The ath element of tth vector, ω_t^a, represents the proportion of the available capital invested in asset a. This $\omega_t^a > 0$ is determined by the SMAB algorithm with the additional condition that ω_t sums to 1, i.e., $\sum_{a=1}^{K} \omega_t^a = 1$. The Cumulative Wealth (is indicated in equations by) W_H, which is the realized wealth at the end of time horizon H, is calculated as shown below

$$W_H = \rho_0 \prod_{t=1}^{H} \omega_t . R_t = \rho_0 \prod_{t=1}^{H} \sum_{a=1}^{K} \omega_t^a . R_t^a \qquad (3)$$

The S_t^a to be used in the SMAB algorithm is the Sharpe Ratio of past returns R_t^a. However, here a parameter τ indicating the moving-window setting is used in (Shen et al., 2014). This moving-window is the number of periods of data, prior to the current decision-making period, the algorithm should consider while calculating S_t^a. Thus, for an NBP, the true return for an asset a in iteration t is given as $X_t^a = S_t^a$ where

$$S_t^a = \frac{\mu_t^a(\tau)}{\sigma_t^a(\tau)} \tag{4}$$

The τ denotes the moving-horizon time period. Here, $\mu_t^a(\tau)$ is the average of previous τ returns $\frac{1}{\tau}\sum_{s=t-\tau}^{s=t} R_s^a$ before t. For example, when =100 , μ_102^a (100) is the mean return of past 100 periods before the period t=102. Similarly, $\sigma_t^a(\tau)$, is the standard deviation of the returns $\{R_s^a\}_{s=t-\tau}^t$. For our work, we consider τ = 120 where mean (μ) and standard deviations (σ) of the past 120 time periods is taken in to consideration for calculating S_t^a. As mentioned previously, for an SMAB algorithm to operate, an assumption that the reward $S_t^a \in [0,1]$ is necessary. To obtain S_t^a in [0, 1], we normalize the S_t^a based on data for every asset a in period t as given below.

$$\overline{S}_t^a \equiv \frac{S_t^a - min_b\left(S_t^b\right)}{max_b\left(S_t^b\right) - min_b\left(S_t^b\right)} \tag{5}$$

S_t^a is given as input to the NBP algorithm (see Algorithm 1). From the NBP Algorithm 1, notice that an SMAB is implemented to compute the weights vector for any iteration t ∈ {τ + 1,...,H}. Current work compares the performance of the proposed NBP-effSAMWMIX with analogous SMAB-based NBP-UCB1, NBP-KLUCB, and NBP-TS algorithms - where the kernels are different bandit algorithms. The functioning of the NBP versions of each of these algorithms is presented below.

2.3 NBP-UCB1 algorithm

UCB1 is a first-generation SMAB algorithm that updates both exploration and exploitation components, to store them additively in a UCB parameter (Auer, Cesa-Bianchi, & Fischer, 2002). UCB here is an acronym for Upper Confidence Bound. The following UCB1 parameter is updated in every iteration for every arm (asset)

$$U_t^a = \hat{\overline{S}}_t^a + \sqrt{\frac{2\ln(t)}{N_t(a)}} \tag{6}$$

where $\hat{\overline{S}}_t^a$ is the mean of the observed (normalized) Sharpe Ratios of arm (asset) a until iteration t, at the instances that arm a has been pulled. Also, $N_t(a)$ is the number of times arm a has been played until iteration t. To explore the rewards obtained if any arm is pulled, UCB1 allows mandates all arms are pulled at least once. Thus, the minimum value of $N_t(a)$ will be 1. In the context of this work, the Sharpe Ratio ($\overline{S}_t^a$) which is the normalized risk-adjusted returns from the asset. The asset to be invested in at iteration t is calculated by

$$a_t = argmax_{a\in 1,2,\dots,K}\left(\hat{\overline{S}}_{t-1}^a + \sqrt{\frac{2\ln(t)}{N_{t-1}(a)}}\right) \tag{7}$$

Followed by updates $N_{t+1}(a_t) := N_t(a_t) + 1$ and $N_{t+1}(a) := N_t(a)$, for $a \neq a_t$. The $N_t(a_t)$ - sample empirical mean $\hat{\overline{S}}_t^a$ is updated with the new sample $\overline{S}_t^a$. While $\hat{\overline{S}}_t^a$ represents the exploitation component, the term $\sqrt{\frac{2\ln(t)}{N_t(a)}}$ is the exploration bonus adjusting for arms that have not been tried out enough. UCB1 captures the principle of "optimism under uncertainty", with the parameters getting updated simultaneously with knowledge related to both exploration and exploitation. Note here that though $\overline{S}_t^a$ is analogous to reward in this MAB, it is not i.i.d (since there is dependence on other assets due to normalization). However, relaxing the i.i.d. assumption only entails loss of certain properties like logarithmic regret, and doesn't render a MAB unsuitable for OPSP. With other details specific to this OPSP, the NBP-UCB1 is written as Algorithm (1).

Algorithm 1: NBP-UCB1 Algorithm

```
1.  WithInputs: $K$(assets), H (Horizon), $\Delta t$, $R_t$, $\tau$
2.  For $t = \tau + 1$ to $H$, do
3.  Calculate moving − window average return $\mu_t^a(\tau) = \frac{1}{\tau}\sum_{s=t-\tau}^{t} R_s^a$
4.  Calculate standard deviation $\sigma_t^a(\tau)$ of the returns $= \{R_s^a\}_{s=t-\tau}$
5.  Calculate $S_t^a = \frac{\mu_t^a(\tau)}{\sigma_t^a(\tau)}$
6.  Calculate $\overline{S}_t^a = \frac{S_t^a - min_b(S_t^b)}{max_b(S_t^b) - min_b(S_t^b)}$ to normalize $S_t^a$
7.  end For
8.  Initialize UCB1, $\hat{\overline{S}}_{\tau+K}^a := \overline{S}_{\tau+a}^a . N_{\tau+K}(a) := 1$ for all $1 \le a \le K$
9.      For every iteration $t = \{\tau + K + 1, \tau + K + 1 \ldots H\}$
10. Choose asset $a_t = argmax_{a\in 1,2,\ldots,K}\left(\hat{\overline{S}}_{t-1}^a + \sqrt{\frac{2\ln(t)}{N_{t-1}(a)}}\right)$
11. Update $\hat{\overline{S}}_t^a := \frac{N_{t-1}(a_t).\hat{\overline{S}}_{t-1}^a + \overline{S}_t^{a_t}}{N_{t-1}(a_t)+1}$ and $N_t(a_t) := N_{t-1}(a_t) + 1$
12. For all other $a \neq a_t$, $\hat{\overline{S}}_t^a := \hat{\overline{S}}_{t-1}^a$ and $N_t(a_t) := N_{t-1}(a_t)$
13.     end For
```

2.4 NBP-KLUCB algorithm

KL-UCB is proposed in (Garivier & Cappé, 2011), wherein KL stands for Kullback-Leibler divergence, an information theoretic measure of how well-sampled an empirical mean is with respect to other empirical means. It differs from UCB1 in the exploration bonus term (analogous to $\sqrt{\frac{2\ln(t)}{N_t(a)}}$) above) which is derived by employing KL -divergence. KL-UCB is reported to possess improved regret bounds where the exploration term incorporates the distance between estimated reward distributions for the arms when calculating the UCB parameter. The NBP-KLUCB that employs KL-UCB as its decision engine as given in Algorithm (2). Notice in NBP-KLUCB that each iteration involves an optimization problem (marked as equation) and that this is solved by gradient descent algorithm using a heuristic in (Garivier & Cappé, 2011).

Algorithm 2: NBP-KLUCB Algorithm

```
1.  With Inputs: $K$(assets), H (Horizon), $\Delta t$, $R_t$, $\tau$
2.  For $t = \tau + 1$ to $H$, do
3.  Calculate moving − window average return $\mu_t^a(\tau) = \frac{1}{\tau}\sum_{s=t-\tau}^{t} R_s^a$
4.  Calculate standard deviation $\sigma_t^a(\tau)$ of the returns $= \{R_s^a\}_{s=t-\tau}$
5.  Calculate $S_t^a = \frac{\mu_t^a(\tau)}{\sigma_t^a(\tau)}$
6.  Calculate $\overline{S}_t^a = \frac{S_t^a - min_b(S_t^b)}{max_b(S_t^b) - min_b(S_t^b)}$ to normalize $S_t^a$
7.  end For
8.  Initialize UCB1, $\hat{\overline{S}}_{\tau+K}^a := \overline{S}_{\tau+a}^a . N_{\tau+K}(a) := 1$ for all $1 \le a \le K$
9.  For $t = \tau + K + 1$ to $H$, do
10. $a_t := argmax_{a\in 1,2,\ldots,K}\max\left\{q \in [0,1]: N_{t-1}(a).d.\left(\frac{\hat{\overline{S}}_t^a}{N_{t-1}(a)}, q\right) \le log(t) + c.log(log(t))\right\}$
```

```
11. Calculate value of $d$ in step 10 as $d(p,q) = p\log\left(\frac{p}{q}\right) + (1-p)\log\left(\frac{1-p}{1-q}\right)$
12. Set the value of $c$ in step 10 as $c := 0.c$ Calculateis a heuristic in (Garivier & Cappé, 2011)
13. Update $\hat{\bar{S}}_t^a := \frac{N_{t-1}(a_t).\hat{\bar{S}}_{t-1}^a + \bar{S}_t^{a_t}}{N_{t-1}(a_t)+1}$ and $N_t(a_t) := N_{t-1}(a_t) + 1$
14. For all other $a \neq a_t$, $\hat{\bar{S}}_t^a := \hat{\bar{S}}_{t-1}^a$ and $N_t(a_t) := N_{t-1}(a_t)$
15. end For
```

2.5 *NBP-TS (Thompson Sampling) algorithm*

Thompson Sampling (TS) was proposed within (Thompson, 1933) in 1933 but remained less popular compared to other MAB algorithms for the lack of proofs on the regret bounds. The empirical performance of TS is reported to be better than UCB (Gopalan, Mannor, & Mansour, 2014) and is considered to be a competent algorithm owing to its practical usability (Russo & Van Roy, 2016). The proof for logarithmic regret in the Thompson sampling SMAB, under typical conditions, has come only recently (Agrawal & Goyal, 2012; Kaufmann et al., 2012). Hence, we compared the performance of NBP-TS with NBP-effSAMWMIX in this work. The NBP-TS that employs TS as its decision engine is given in Algorithm (3).

Algorithm 3: NBP-TS Algorithm

```
1.  With Inputs: $K$(assets), $H$(Horizon), $\Delta t$, $R_t$, $\tau$, $S_t^a$
2.  For $t = \tau + 1$ to $H$, do
3.  Calculate moving − window average return $\mu_t^a(\tau) = \frac{1}{\tau}\sum_{s=t-\tau}^{t} R_s^a$
4.  Calculate standard deviation $\sigma_t^a(\tau)$ of the returns $= \{R_s^a\}_{s=t-\tau}$
5.  Calculate $S_t^a = \frac{\mu_t^a(\tau)}{\sigma_t^a(\tau)}$
6.  Calculate $\bar{S}_t^{a_t} = \frac{S_t^a - \min_b(S_t^b)}{\max_b(S_t^b) - \min_b(S_t^b)}$ to normalize $S_t^a$
7.  end For
8.  $s^a := 0, f^a := 0, \forall a$ where $s^a$ is "success counter" and $f^a$ is "failure counter"
9.  For $t = \tau + 1$ to $H$, do
10.     For $a \in \{1,2,\ldots..K\}$ do
11. Draw random variable $\theta^a$ according to distributions $Beta(s^a + 1, f^a + 1)$
12.     End For
13. $a_t := argmax_a(\theta^a)$,
14. With $\bar{S}_t^{a_t}$ as the Bernoulli parameter, obtain a random variable $r_t \in [0,1]$
15. Update $s^{a_t} := s^{a_t} + r_t$ and $f^{a_t} := f^{a_t} + r_t^c$ where $r_t^c$ is complement of $r_t$
16. End For
```

2.6 *NBP-effSAMWMIX algorithm*

The algorithm Efficient SAMWMIX (or effSAMWMIX) differs from UCB1 or KL-UCB since it avoids searching for a maximum among K values, as described in (Villari & Abdulla, 2017). Instead, it picks a 'soft maximum' using a Boltzmann Exploration structure, but with tailored step sizes as the iteration t approaches the total horizon H as indicated by $t \rightarrow H$. In each t, effSAMWMIX calculates a pull probability vector ϕ_t over the K arms and pulls one of these arms according the probability mass function in ϕ_t. This ϕ_t vector is then updated with the learning the iteration t, notably based on the term $\bar{S}_t^{a_t}$ noticed at iteration t. The best arm a^* would be such that the probability of pulling the best arm is as close to 1 as possible $\phi_t^{a^*} \rightarrow 1$, while for all other arms indicated by a, the probability of pulling (choosing) the arm $\phi_t^a \rightarrow 0$. The update equation for φ is given in equation (8)

$$\phi_{t+1}^a = (1-\gamma_t)\frac{e^{\left(\sum_{r=1}^{t}\eta_r \tilde{\bar{S}}_r^a\right)}}{\sum_{b=1}^{K} e^{\left(\sum_{r=1}^{t}\eta_r \tilde{\bar{S}}_r^b\right)}} + \frac{\gamma_t}{K} \tag{8}$$

Here, $\tilde{\bar{S}}_r^b = \frac{\bar{S}_t^b . I_{a_t = b}}{\phi_t^b}$ where $I_{a_t} = b$ is the indicator function if $a_t = b$. It is to be noted here that the difference between empirical means of the type $\hat{S}_t^b$ in the previous algorithms UCB1, KL-UCB and TS, and the quantity $\tilde{\bar{S}}_r^b$ here. The effSAMWMIX algorithm does not depend on inequalities on empirical means (e.g. Chernoff Bound, Hoeffding Bound) for its proof of logarithmic regret, whereas the other algorithms do. The learning component above is the step-size γ_t and the inverse temperature parameter η_t is given in equation (9) and (10).

$$\gamma_t = \frac{K\big(4 + (d + d_t)\big)}{t(d + d_t) - (d + d_t - 2d^2)} \tag{9}$$

$$\eta_t = \frac{1}{\left(\frac{K}{\gamma_t + 1}\right)} \log \left(\frac{1 + d\left(\frac{K}{\gamma_t} + 1\right)}{\frac{2K}{\gamma_t} - d^2} \right) \tag{10}$$

Note that d_t in equation (9) is obtained by a heuristic that must be iteratively calculated rather than obtained as input. Further, the term d is a negligibly small quantity (say 0.05but $\neq 0$, which should satisfy the condition that $d < min(\Delta^a)$ (for $a \neq a^*$), where $\Delta^a = E(X_t^*) - E(X_t^a)$ (using notation from 2.1). The NBP-effSAMWMIX that employs effSAMWMIX (Villari & Abdulla, 2017) as its decision engine is as given in Algorithm (4). In equation (9) and equation (10), d is a pre-set small value while d_t is computed at each iteration t using an iterative heuristic as given in Algorithm 4. The usage of the computed d_t obtains a tighter and more accurate bound on regret of effSAMWMIX compared to the base algorithm SAMWMIX (Abdulla & Bhatnagar, 2016).

Algorithm 4 : NBP-effSAMWMIX Algorithm

```
1.  **With Inputs**: $K$(assets), $H$(Horizon), $\Delta t$, $R_t$, $\tau$, $S_t^a$ and $d$
2.  **For** $t = \tau + 1$ to $H$, **do**
3.  Calculate moving – window average return $\mu_t^a(\tau) = \frac{1}{\tau}\sum_{s=t-\tau}^{t} R_s^a$
4.  Calculate standard deviation $\sigma_t^a(\tau)$ of the returns = $\{R_s^a\}_{s=t-\tau}$
5.  Calculate $S_t^a = \frac{\mu_t^a(\tau)}{\sigma_t^a(\tau)}$
6.  Calculate $\bar{S}_t^a = \frac{S_t^a - min_b(S_t^b)}{max_b(S_t^b) - min_b(S_t^b)}$ to normalize $S_t^a$
7.  **end For**
8.  **Initialize effSAMWMIX**:
9.  $C_0 := K + 1$ and $\sigma^2 = 2K$ ( Please note that this $\sigma^2$ has no relation to Asset Price Variance)
10.     $\eta_0 := \frac{1}{C_0}\log\left(\frac{1+C_0 d}{\sigma^2}\right)$, $Z = \frac{(4+d)K+d}{d^2}$
11. **For** $a \in \{1,2, \ldots . . K\}$**do**
12.     Initialize$\phi_{Z+K}^a \leftarrow \eta_0 \left(\frac{Z}{K}\right)\left(\frac{S_t^a}{1/K}\right)$
13. **End For**
14. **For** $t \in \{Z + K, Z + K + 1, \ldots . , Z + H\}$**do**
15. Pick the asset $a_t$which has maximum value of $\phi$ among all $K$ arms
16.  Calculate adjusted reward $\tilde{\bar{S}}_t^{a_t} = \frac{\bar{S}_t^a}{\phi_t^{a_t}}$
```

```
17. **Assume** $d_{tStep} := 0{:}01$, heuristic to obtain $d_t$ start:
18. **For** $d_{iter} = 1,2,\ldots.\frac{t-Z}{H-K}$ **do**
19.     $K_t := d + d_{iter}$ and $\gamma_t = \left(\frac{K(4+K_t)+K_t}{t.K_t^2}\right)$
20.     $C_t \leftarrow \left(\frac{K}{\gamma_t}+1\right), \sigma_t^2 \leftarrow \left(\frac{2K}{\gamma_t}-1\right)$ and $\eta_t \leftarrow \frac{1}{C_t}\log\left(\frac{1+C_tK_t}{\sigma_t^2}\right)$
21.     $P_t := \sum \phi_t^i + \exp(\sum \eta_t\, \overline{\mathrm{S}}_{\mathrm{t}}^{a_t})$
22.     **If** $\exp\left[\sum \eta_t \left(d_t \overline{\mathrm{S}}_{\mathrm{t}}^{a_t}\right)\right] > P_t$
23.         $d_t := d_{iter} - d_{tStep}$
24.         Go to Step 30
25.     **End If**
26.     $d_{iter} := d_{iter} + d_{tStep}$
27. **End For**
28. **Assign** $K_t \leftarrow d + d_t$
29. $\gamma_t = \left(\frac{K(4+K_t)+K_t}{t.K_t^2}\right), C_t = \left(\frac{K}{\gamma_t}+1\right), \sigma_t^2 = \left(\frac{2K}{\gamma_t}-1\right)$
30. $\eta_t = \frac{1}{C_t}\log\left(\frac{1+C_tK_t}{\sigma_t^2}\right)$
31. **Update** $\forall a \in \{1,2,\ldots..K\}, \phi_{t+1}^a = (1-\gamma_t)\frac{e^{\left(\sum_{r=1}^{t}\eta_r\tilde{\tilde{S}}_r^a\right)}}{\sum_{b=1}^{K} e^{\left(\sum_{r=1}^{t}\eta_r\tilde{\tilde{S}}_r^b\right)}} + \frac{\gamma_t}{K}$
32. **End For**
```

The performance of the NBP algorithms are compared against both simulations and real-world benchmark datasets as explained in the following sections.

3 Experiments

We conducted the experiments on both simulated datasets and real-world datasets. The simulated datasets are synthesized using Geometric Brownian Motion, with requisite mean μ and standard deviation parameters σ obtained from actual time series.

3.1 Stock prediction on Simulated Geometric Brownian Motion Datasets

Geometric Brownian Motion (GBM) is also known as Wiener Process in which the logarithm of a quantity that varies at random will follow a Brownian Motion (Wilmott, 2000; Wilmott, 2013). GBM is formally a mathematical modeling technique that is often used to model short-term stock price movements (Ladde & Wu, 2009). Since the stock price movement is often unpredictable the GBM's random walk model tends to predict the stock prices with reasonable accuracy (Fama, 1995). These suggestions have been validated recently to a fair extent by work such as (Reddy & Clinton, 2016). Our work utilizes the GBM technique to build a synthetic dataset to test the performance of the NBP algorithm that utilizes effSAMWMIX, UCB1, KL-UCB and TS as the SMAB engine for the NBP.

The GBM data set is generated using the daily closing prices, which are the input for the mean μ and σ used in the GBM model. If the price of stock at time 0 (indicated by t_0) is ρ_0 and a one-dimensional Brownian process X_t is available for $t > 0$, then ρ_t is calculated using GBM simulation as given in 11 (Ladde & Wu, 2009; Marathe & Ryan, 2005).

$$\rho_t = \rho_0 \exp\left[\left(\mu - \frac{\sigma^2}{2}\right)t - \sigma(X_t - X_0)\right] \qquad (11)$$

For existing time series obtained from databases, the returns at discrete time step t, for each asset a, are calculated using the following equation (12)

$$R_t^a = \frac{\rho_{t+1}^a - \rho_t^a}{\rho_t^a} \qquad (12)$$

Where ρ_t^a is the closing price of the asset on day (period) t. If H is the total number of periods for which the returns are computed, then the mean return $\hat{\mu}$ is calculated as follows

$$\hat{\mu}^a = \frac{1}{H} \sum_{t=1}^{H} R_t^a \qquad (13)$$

Also, an estimate standard deviation of all the returns σˆ is calculated as given in 14 with the Sharpe Ratio (needed in each experiment) being $S = \frac{\hat{\mu}}{\hat{\sigma}}$.

$$\hat{\sigma} = \sqrt{\frac{1}{H-1} \sum_{t=1}^{H} (R_t^a - \hat{\mu}^a)^2} \qquad (14)$$

In each experiment, from the S&P 500 Stock data set (Bruni et al., 2016), we have randomly picked K stocks from the same so that the SMABs will have K arms (assets) to choose from. In the following experiments we take K = 5 and K = 15 to generate two independent GBM datasets. Each of the stocks have the periodic closing prices from November 2004 to April 2016. Note that the purpose of using S&P 500 Stock data set is only to obtain realistic values of μ and σ so that the GBM simulated data could represent a near realistic scenario.

Our experiments report the average results over multiple runs to nullify any outlier effect of extremely favorable or unfavorable results. Using each simulated data set, we run each of the NBP algorithms for obtaining the terminal cumulative wealth. We conduct 100 such experiments where each algorithm runs and report the average of these returns.

This newly generated stock closing price data will now be the data set on which the NBP's performance is evaluated (when the NBP uses a different SMAB for decision-making process). The naming convention for the GBM simulated portfolio data set with 5 assets i.e., K = 5 is GBM05 and that with 15 assets i.e., K = 15 is GBM15. The results of the experiments on GBM05 and GBM15 datasets are given in Table 1, the horizon H employed for simulation was 632 (approximately 3 years) and τ used was 120.

Table 1: Terminal Cumulative Wealth on GBM Datasets

Cumulative Wealth (per unit $)					
Dataset Name	**Market Buy & Hold**	**NBP-UCB1**	**NBP-KLUCB**	**NBP-TS**	**NBP-effSAMWMIX**
GBM05 Dataset	1.1736	1.3789	0.8623	1.5167	1.6213
GBM15 Dataset	1.2881	1.4024	1.5597	1.7045	1.7867

NBP-effSAMWMIX performed better than when NBP-UCB1, NBP-KLUCB, NBP-TS. Also, NBP-effSAMWMIX has acquired a better CW than the Buy & Hold strategy, wherein all 51 of resource is assigned to each of the 5 stocks throughout the iterating time horizon H. Results are similarly favorable for NBP-effSAMWMIX when simulated portfolio consisted of 15 assets. The terminal cumulative wealth acquired per a unit investment for benchmark datasets is shown in Table 3.

3.2 Stock prediction on real-world benchmark datasets

We choose benchmark datasets from (Bruni et al., 2016) and (Li, Sahoo, & Hoi, 2016) where the datasets are validated for the comparative performance of portfolio selection models. These datasets are generated using real-world price values obtained from major stock markets. They are reported to contain error-free (cleaned data) of weekly return values, which are adjusted for dividends and stock splits. These publicly available datasets help in an unbiased comparison of the different NBP-SMAB portfolio selection strategies that are tested in this work. We chose these datasets to get a variety of data in terms of region, market type, the number of assets and the number of periods. For example, MSCI measures the equity market performance of global emerging markets and DJIA gives the stock market data from the USA, which is a developed economy. Of these real-world databases, DJIA,TSE and MSCI are considered for evaluation in OLPS tool box for portfolio selection which is a pioneering work on strategies for portfolio selection problem (Li et al., 2016). NASDAQ100, which is a weekly returns data, is made publicly available by (Bruni et al., 2016) for evaluation of strategies. on In Table 2, we provide the details of the datasets we considered for this work. These datasets help us comprehensively evaluate the stock prediction algorithms on variety of data i.e. data on daily returns, weekly returns, data from developed economy and on data from emerging markets.

Table 2: Summary of the benchmark datasets from real markets

Dataset	Market	Region	Time Frame	# Periods	# Assets
DJIA (Li et al., 2016)	Stock	USA	January 14, 2001 -January 14, 2003	507	30
TSE (Li et al., 2016)	Stock	CANADA	January 4 ,1994-December 31,1998	1259	88
NASDAQ100 (Bruni et al., 2016)	Stock	USA	June 2002 -April 2016	596	82
MSCI (Li et al., 2016)	Index	Global	January 14, 2001 -January 14, 2003	507	30

The abbreviations of each of these datasets is briefly clarified as follows

- DJIA: Dow Jones Industrial Average Data
- TSE: Toronto Stock Exchange Data
- NASDAQ100: Nasdaq Inc. Stock Exchange data consisting of weekly returns data for the time between June,2002 and April,2016
- MSCI: Morgan Stanley Capital International (emerging markets)

In Table 3, we report the terminal cumulative wealth achieved by each of these algorithms over the four-benchmark datasets mentioned above. NBP-effSAMWMIX has achieved the highest cumulative wealth when compared to other NBP algorithms. Except in the case of NASDAQ100 data set, NBP-effSAMWMIX has performed better than the Market Buy & Hold strategy as well. On DJIA and TSE datasets, NBP-effSAMWMIX has performed similar to albeit slightly better than NBP-TS. However, it has distinguishably better performance on MSCI data set. On the NASDAQ100 data set, the Market Buy & Hold strategy is a clear winner from the early investment periods and none of the NBP algorithms could match its performance. Except for this case, NBP-effSAMWMIX achieved the highest wealth level in all the datasets including the simulated GBM data sets.

Table 3: Terminal Cumulative Wealth on Benchmark Datasets

Cumulative Wealth (per unit $)					
Dataset Name	**Market Buy & Hold**	**NBP-UCB1**	**NBP-KLUCB**	**NBP-TS**	**NBP-effSAMWMIX**
DJIA	0.85	0.48	0.93	0.8	0.96
TSE	1.58	1.96	1.85	2.02	2.15
NASDAQ100	5.31	1.85	4.06	2.92	4.38
MSCI	0.96	1	0.92	1.06	1.24

Since multiple experiments are conducted, we also report a technique to use the Sharpe Ratio (SR) with the p-values over the investment period to infer the best-performing algorithm. The SR values indicate the risk-adjusted returns for the investment periods. The p-values are required as it is important to test whether the risk-adjusted returns are drawn from the same distribution for two-investment strategies. It is known that if the risk is higher, the peak returns could also be higher. Thus, in order to compare the if an OPSP is better than another, we tested for the similarity of SRs in distribution. To compute the p-values for the case of non-i.i.d returns, we adopted the Studentized circular block bootstrapping technique (Ledoit & Wolf, 2008; Shen et al., 2015). The essence of circular block bootstrapping technique is that it uses a number of bootstrap repetitions represented by M and a input block size b which is used to resample new blocks of data pairs from the observed pairs with replacement. Following the testing parameters reported in (Shen et al., 2015), we used M = 1000 and b = 5 . These p-values are used to further quantify the statistical significance of the difference in SR between the two comparing portfolios. The Null hypothesis (H0) is that the Shape Ratios of the portfolios in comparison have the same mean.

For the results shown in Table 4, we set the Market Buy & Hold strategy as the benchmark with 1000 bootstrap samples at 95% significance level and with block size 5.

Table 4: Results on Benchmark Datasets - Terminal Sharpe Ratios (and Corresponding P-Values

Datasets		**Market Buy & Hold**	**NBP-UCB**	**NBP-KLUCB**	**NBP-TS**	**NBP -effSAMWMIX**
DJIA	Sharpe Ratios	64.4551	61.837	55.2489	65.1978	58.1901
	p-values	1	0.4765	0.001	0.8751	0.4476
TSE	Sharpe Ratios	123.6978	96.5608	91.7642	148.3123	152.0979
	p-values	1	0.045	0.003	0.001	0.001
NASDAQ100	Sharpe Ratios	32.0086	32.2987	27.8164	38.4079	36.0387
	p-values	1	0.8931	0.015	0.001	0.001
MSCI	Sharpe Ratios	63.6745	79.4278	58.6742	71.801	73.6678
	p-values	1	0.001	0.001	0.002	0.012

Note: The p-values provided are reported in comparison with Market Buy & Hold Strategy

Further in the following paragraphs, we explain with Table 4, for a better comprehension of how NBP-effSAMWMIX exhibits a performance which is better than others and if not, is at least as good as the best performer. For example, the most basic component of Toronto Stock Exchange (TSE), which is presented below for easier reading.

Table 5: TSE Data set: Terminal Cumulative Wealth, Sharpe Ratios and p-values

Datasets		**Market Buy & Hold**	**NBP-UCB**	**NBP-KLUCB**	**NBP-TS**	**NBP -effSAMWMIX**
TSE	Cumulative Wealth (per $)	1.58	1.96	1.85	2.02	2.15
	Sharpe Ratios	123.6978	96.5608	91.7642	148.3123	152.0979
	p-values	1	0.045	0.003	0.001	0.001

The Sharpe Ratios i.e. risk-adjusted returns of NBP-effSAMWMIX are the best (152.1) among the five competing strategies. In addition, its corresponding p-value is 0.001 < 0.05. This indicates that we can reject the hypothesis that claims the mean SR value of NBP-effSAMWMIX and mean SR value of Market Buy & Hold are same. Thus, NBP-effSAMWMIX not only has a cumulative wealth of 2.15, which is the highest but also has a higher risk-adjusted return whose distribution is different from that of the Market Buy & Hold Strategy. This suggests that for TSE data, NBP-effSAMWMIX performs better than the rest of the strategies.

Table 6: MSCI Dataset-Terminal Cumulative Wealth, Sharpe Ratios and p-values

Datasets		**Market Buy & Hold**	**NBP-UCB**	**NBP-KLUCB**	**NBP-TS**	**NBP -effSAMWMIX**
MSCI	Cumulative Wealth (per $)	0.96	1	0.92	1.06	1.24
	Sharpe Ratios	63.6745	79.4278	58.6742	71.801	73.6678
	p-values	1	0.001	0.001	0.002	0.012

Consider the example of MSCI data where the risk-adjusted returns of NBP-effSAMWMIX are not the best. Here, the SR values of NBP-UCB are higher (79.43) than that of NBP-effSAMWMIX (73.67). Though the SR values of NBP-UCB and NBP-effSAMWMIX are better than Market Buy & Hold strategy, since the p-values of both these are less than 0.05, we can infer that the risk-adjusted returns of both these strategies are neither better nor worse than Market Buy & Hold strategy. Thus, it makes sense to look at the cumulative wealth value to decide on the best strategy. NBP-effSAMWMIX, which has a CW of 1.24, is the best strategy for the MSCI data set. Using similar logic, NBP-effSAMWMIX is the best performer on DJIA data set as well. On NASDAQ100 data set where CW obtained by NBP-effSAMWMIX (4.38) is less than that of Market Buy & Hold (5.31), the SR related p-values is 0.001. This indicates that the risk-adjusted returns of NBP-effSAMWMIX could not be from another distribution. Also, CW of NBP-effSAMWMIX is better than the rest of the strategies (Table 3). To comprehensively conclude the effectiveness of NBP-effSAMWMIX over competing for NBP strategies, we performed the

experiments on datasets used in (Shen et al., 2015) by obtaining the same through personal communication. The details of the datasets we obtained are given in Table 7. Note that their reference is an Equal Weighted (EW) portfolio, which is not a popular strategy for industrial practitioners due to higher turnover and additional transaction costs for frequent rebalancing of portfolios (Lynch & Balduzzi, 2000).

Buy & Hold strategy is still popular among long term investors and the datasets we considered are over a horizon of at least a couple of years (see Table 3), our previous results were presented with Market Buy & Hold strategy as the reference. We could replicate their experimental results and the results we present below are to visualize a direct comparison with those published in the literature. We follow the same format as used in Shen et.al's work (Shen et al., 2015).

Table 7: Description of Shen.et.al's datasets

Dataset	**Time Frame**	**# Periods**	**# Assets**
FF48	January 01, 1963 -December 31, 2004	498	48
FF100	January 01, 1963 -December 31, 2004	498	100
ETF139	January 01, 2008 -October 30, 2012	252	139

- Datasets are taken from (Shen et al., 2015)
- FF48, FF100: Fama & French Datasets with portfolios representing different industrial sectors
- ETF139: Exchange-traded funds (weekly data) from Yahoo! Finance

These datasets (Table 7) are publicly available but since Fama & French datasets are updated on a regular basis, the data set values get updated with time. Hence, to replicate and comprehensively compare the results from (Shen et al., 2015), we utilized the same data sets obtained from the authors. Since we could replicate the values of Equal Weighted Portfolio (EW) and NBP-UCB1 on these datasets, we did not perform the experiments on Value Weighted Portfolio (VW), Minimum-Variance Portfolio (MVP) and Online Moving Average Reversion (MAR). Instead, we used the data available from that publication and compared it against the performance of NBP-effSAMWMIX on the same data set. The results with the values of Cumulative wealth, Sharpe Ratios, and the corresponding p-values are given in the Tables 8 and 9 given below. It is interesting to note that OLMAR, an advanced OPSP algorithm that is reported to have superior performance (Li & Hoi, 2012) is outperformed by fundamental portfolio strategies like Value-Weighted portfolio (VW) and Equal Weighted portfolio (EW). Since we are comparing NBP algorithms, on the data from the following, the attention is on the results of NBP-UCB1 and NBP-effSAMWMIX. In Table 8, OBP stands for Orthogonal Bandit Portfolio, MVP represents minimum-variance portfolio, MAR represents on-line moving average reversion portfolio. While EW, VW and MVP strategies are typical baseline strategies studied in the finance literature, MAR (Li & Hoi, 2014) is an advanced online portfolio selection strategy. It is seen (in Table 8) that on these datasets as well, NBP-effSAMWMIX strategy has delivered a better cumulative wealth than NBP-UCB1.

Table 8: Cumulative Wealth obtained on Shen et.al's datasets

Portfolio Terminal Cumulative Wealth (per $)							
Dataset	**OBP**	**NBP-UCB1**	**EW**	**VW**	**MVP**	**MAR**	**NBP-effSAMWMIX**
FF48	61.75	35.23	54.77	48.06	25.1	42.34	42.22
FF100	626.04	76.91	123.92	198.32	73.73	57.74	92.62
ETF139	1.42	1.35	1.2	1.19	1.05	1.21	2.38

Also, the p-values of their Sharpe ratios indicate that the risk-adjusted returns are neither worse nor better than the EW strategy. Being unable to reject H0 is a favorable result for NBP-effSAMWMIX as it could mean that risk adjusted returns bear a distribution with a similar mean as that of EW (and that of NBP-UCB1) while the cumulative wealth is higher than that of NBP-UCB1 in all the three datasets. Especially on ETF139, NBP-effSAMWMIX outperformed the rest of the algorithms indicating the potential advantage of the same.

Table 9: Portfolio Sharpe ratios (%) with the significance level measured by p-values with respect to EW.

Dataset		Portfolio Sharpe ratios (%) with the significance level measured by p-values with respect to EW.						
		OBP	**NBP-UCB1**	**EW**	**VW**	**MVP**	**MAR**	**NBP-effSAMWMIX**
FF48	Sharpe Ratio	26.15	25.68	24.3	23.37	22.38	24.48	22.8759
	p-value	0.64	0.8	1	0.22	0.72	0.93	0.1259
FF100	Sharpe Ratio	34.89	26.09	26.97	29.76	18.01	23.6	20.612
	p-value	0.01	0.82	1	0	0.19	0.22	0.0529
ETF139	Sharpe Ratio	25.47	15.45	6.01	5.85	6.82	7.61	32.0148
	p-value	0.05	0.04	1	0.22	0.94	0.44	0.3506

4 CONCLUSION

In this work, we reported the implementation of the effSAMWMIX inside of a Naive Bandit Portfolio algorithm. In our working paper (Villari & Abdulla, 2017) we showed that effSAMWMIX has a better performance than KL-UCB and Thompson Sampling when the rewards followed a range of mathematical distributions. We intended to exploit this advantage (of effSAMWMIX) over competing SMAB algorithms reported in the literature by employing effSAMWMIX as a decision engine in a Naive Bandit Portfolio (NBP) algorithm. Along with NBP-effSAMWMIX, the NBP versions of KL-UCB and Thompson Sampling are reported for the first time. Our results include the cumulative wealth values on both simulated and benchmark real-world datasets to evaluate the empirical performance of the proposed algorithm. While competitive as an NBP engine, the performance on NASDAQ100 data set requires us to assess the data set and analyze why none of the NBP algorithms could beat the Market Buy & Hold strategy (while they could in the rest of the cases). This could be because NBP does not consider asset correlations while making the decision. To further this work, we intend to do an Orthogonalization of the portfolios in order to remove the correlation and evaluate the performance of effSAMWMIX in such a scenario (Shen et al., 2015).

REFERENCES

Abdulla, M. S., & Bhatnagar, S. 2016. Multi-armed bandits based on a variant of Simulated Annealing. *Indian Journal of Pure and Applied Mathematics, 47*(2): 195-212.

Agarwal, A., Hazan, E., Kale, S., & Schapire, R. E. 2006. Algorithms for portfolio management based on the newton method. Paper presented at the Proceedings of the 23rd international conference on Machine learning.

Agrawal, S., & Goyal, N. 2012. Analysis of Thompson Sampling for the Multi-armed Bandit Problem. Paper presented at the COLT.

Auer, P., Cesa-Bianchi, N., & Fischer, P. 2002. Finite-time analysis of the multiarmed bandit problem. *Machine learning, 47*(2-3): 235-256.

Auer, P., Cesa-Bianchi, N., Freund, Y., & Schapire, R. E. 1995. Gambling in a rigged casino: The adversarial multi-armed bandit problem. Paper presented at the Foundations of Computer Science, 1995. Proceedings., 36th Annual Symposium on.

Auer, P., & Ortner, R. 2010. UCB revisited: Improved regret bounds for the stochastic multi-armed bandit problem. *Periodica Mathematica Hungarica, 61*(1-2): 55-65.

Borodin, A., & El-Yaniv, R. 2005. Online computation and competitive analysis: cambridge university press.

Bruni, R., Cesarone, F., Scozzari, A., & Tardella, F. 2016. Real-world datasets for portfolio selection and solutions of some stochastic dominance portfolio models. Data in Brief, 8: 858-862.

Dannoura, E., & Sakurai, K. 1998. An improvement on El-Yaniv-Fiat-Karp-Turpin's money-making bi-directional trading strategy. Information processing letters, 66(1): 27-33.

DeMiguel, V., Martín-Utrera, A., & Nogales, F. J. 2015. Parameter uncertainty in multiperiod portfolio optimization with transaction costs. *Journal of Financial and Quantitative Analysis, 50*(06): 1443-1471.

Fama, E. F. 1995. Random walks in stock market prices. *Financial analysts journal, 51*(1): 75-80.

Fama, E. F., & French, K. R. 1992. The cross-section of expected stock returns. *The Journal of Finance, 47*(2): 427-465.

Fiat, A. 1998. Online Algorithms: The State of the Art (Lecture Notes in Computer Science).

Garivier, A., & Cappé, O. 2011. The KL-UCB Algorithm for Bounded Stochastic Bandits and Beyond. Paper presented at the COLT.

Gopalan, A., Mannor, S., & Mansour, Y. 2014. Thompson sampling for complex online problems. Paper presented at the International Conference on Machine Learning.

Kaufmann, E., Korda, N., & Munos, R. 2012. Thompson sampling: An asymptotically optimal finite-time analysis. Paper presented at the International Conference on Algorithmic Learning Theory.

Kumar, S., & Garg, D. 2012. Online Financial Algorithms Competitive Analysis. arXiv preprint arXiv:1209.6489.

Ladde, G., & Wu, L. 2009. Development of modified geometric Brownian motion models by using stock price data and basic statistics. *Nonlinear Analysis: Theory, Methods & Applications, 71*(12): e1203-e1208.

Lai, T. L., & Robbins, H. 1985. Asymptotically efficient adaptive allocation rules. *Advances in applied mathematics, 6*(1): 4-22.

Ledoit, O., & Wolf, M. 2008. Robust performance hypothesis testing with the Sharpe ratio. *Journal of Empirical Finance, 15*(5): 850-859.

Li, B., & Hoi, S. C. 2012. On-line portfolio selection with moving average reversion. arXiv preprint arXiv:1206.4626.

Li, B., & Hoi, S. C. 2014. Online portfolio selection: A survey. ACM Computing Surveys (CSUR), 46(3): 35.

Li, B., Sahoo, D., & Hoi, S. C. 2016. OLPS: a toolbox for on-line portfolio selection. *Journal of Machine Learning Research, 17*(35): 1-5.

Lisi, F. 2011. Dicing with the market: randomized procedures for evaluation of mutual funds. *Quantitative Finance, 11*(2): 163-172.

Lynch, A. W., & Balduzzi, P. 2000. Predictability and transaction costs: The impact on rebalancing rules and behavior. *The Journal of Finance, 55*(5): 2285-2309.

Marathe, R. R., & Ryan, S. M. 2005. On the validity of the geometric Brownian motion assumption. *The Engineering Economist, 50*(2): 159-192.

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

Merton, R. C. 1969. Lifetime portfolio selection under uncertainty: The continuous-time case. The review of Economics and Statistics: 247-257.

Mohr, E., & Schmidt, G. 2013. How much is it worth to know the future in online conversion problems? *Discrete Applied Mathematics, 161*(10): 1546-1555.

Mossin, J. 1966. Equilibrium in a capital asset market. Econometrica: *Journal of the econometric society*: 768-783.

Reddy, K., & Clinton, V. 2016. Simulating Stock Prices Using Geometric Brownian Motion: Evidence from Australian Companies. Australasian Accounting, *Business and Finance Journal, 10*(3): 23-47.

Rockafellar, R. T., & Uryasev, S. 2000. Optimization of conditional value-at-risk. *Journal of risk, 2*: 21-42.

Russo, D., & Van Roy, B. 2016. An information-theoretic analysis of Thompson sampling. *The Journal of Machine Learning Research, 17*(1): 2442-2471.

Schmidt, G., Mohr, E., & Kersch, M. 2010. Experimental analysis of an online trading algorithm. Electronic Notes in Discrete Mathematics, 36: 519-526.

Shen, W., Wang, J., Jiang, Y.-G., & Zha, H. 2015. Portfolio choices with orthogonal bandit learning. Paper presented at the Proceedings of the 24th International Conference on Artificial Intelligence.

Shen, W., Wang, J., & Ma, S. 2014. Doubly Regularized

Portfolio with Risk Minimization. Paper presented at the AAAI.

Thompson, W. R. 1933. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika, 25*(3/4): 285-294.

Villari, B. C., & Abdulla, M. S. 2017. effSAMWMIX: An efficient Stochastic Multi-Armed Bandit Algorithm based on a Simulated Annealing with Multiplicative Weights.

Wilmott, P. 2000. Quantitative Finance (vols. I and II): Wiley, Chichester, West Sussex, England.

Wilmott, P. 2013. Paul Wilmott on quantitative finance: John Wiley & Sons.

Boby Chaitanya Villari is pursuing Doctoral studies in the field of IT and Systems Area, in Indian Institute of Management Kozhikode,India. He holds a Bachelor degree in Electrical & Electronics Engineering and has a 6 year work experience in VLSI domain. His doctoral thesis is about devising, applying Machine Learning models for deployment in real-world business scenarios. He is fascinated in understanding and conducting research in the fields of IT-Strategy, Deep Reinforcement learning for real-time business applications.

Mohammed Shahid Abdulla is an Associate Professor at Indian Institute of Management Kozhikode,India in the IT and Systems Area, where he has been since 2011. He obtained his PhD in Computer Science and Automation from IISc Bengaluru (2008), ME in Signal Processing from the same institution (2003), and a BE in Computer Engineering from Mangalore University (2001). His research work is in the area of Machine Learning and Simulation-based Optimization.