CS 419(M): Project Report
Learning to Trade Stocks

Rohan Gupta
180010048@iitb.ac.in

Rhishabh Suneeth
200260040@iitb.ac.in

Malhar Ashwin Kulkarni
19D070032@iitb.ac.in

Utsav Desai
200100054@iitb.ac.in

Abstract. In the market, investors are always in the dilemma of which stock to buy or sell, whether they should look at the best performing stock or whether they should try new stocks to see where it would lead them to. We model the portfolio selection problem as a risk aware multi armed bandit instance and compare it with the traditional goodness measure in the context of stocks. We also provide an improvement by observing that the popular convention [1], [2] of posing portfolio selection as a bandit instance can be relaxed while trading stocks. We provide a benchmark for various algorithms in case of bernoulli arms, and an intuitive and statistically correct trading strategy, given a prediction oracle.

Keywords: Multi Armed Bandits · Stock Trading

1 Introduction

Many practical problems like advertising, algorithms used in social media, investment decisions etc. involve decision making in a sequential manner with uncertainty. In such processes, the learner is faced with the dilemma whether to acquire knowledge by trying out new actions(taking new decisions) or to use current knowledge to their immediate advantage in order to find a series of decisions (actions) that lead to an optimal result. This is commonly known as the exploration vs exploitation dilemma.

2 Preliminaries

2.1 Multi Armed Bandits

Suppose there are k slot machines and a sequence of N trials. At each trial t=1...N, the learner chooses to play one of the machines $I_t \in \{1..K\}$ and receives a reward drawn randomly from a corresponding fixed but unknown distribution $\nu_I t$ with mean $\mu_I t$.In the classic setting, the random rewards of the same machine across time are assumed to be independent and identically distributed, and the rewards of different machines are also independent. The objective of the learner is to develop a policy, an algorithm that specifies which machine to play at each trial, to maximize cumulative rewards. A popular measure for the performance of a policy is the regret after some n trials, which is defined to be:

$$\xi(n) = \max_{\{\forall i \in [1,K]\}} \sum_{t=0}^{n} (R_{i,t} - R_{I_t,t}) \tag{1}$$

However, in a stochastic model it is more intuitive to compare rewards in expectation and use pseudo-regret. Let $T_i(n)$ be the number of times machine i is played during the first n trials and let $\mu^* = max\{\mu_1, \mu_2 ... \mu_k\}$. Then,

$$\xi(n) = n\mu^* - \mathbb{E}\sum_{t=0}^{n} R_{I_t,t} = \sum_{1<=i<=K, \mu_i<\mu^*} (\mu^* - \mu_i)\mathbb{E}[T_i(n)] \tag{2}$$

Maximising the cumulative reward is thus equivalent to minimising regret. A $O(\log \log n)$ asymptotic lower bound for the growth of regret has been proven by Lai and Robbins.

2.2 UCB algorithm

In this algorithm, the action chosen at step t is given by

$$A_t = \arg\max_a \mu_t(a) + \sqrt{\frac{2 \log t}{N_t(a)}} \tag{3}$$

where $\mu_t(a)$ is the estimated expectation of the reward of arm a at time t and $N_t(a) = \sum_i = 1^t \mathbb{I}(A_i = 1)$ is the number of times this action has been performed in the past.
The expected regret of this algorithm has been shown to be logarithmic in T where T is the number of steps. The analysis of this algorithm is out of scope of this report.

2.3 Order Optimality for bernoulli arms

Note that the previous algorithm is due to an inequality for any sub-gaussian i.i.d. random variables: $P(\sum_{i=0}^{n} X_i - \mathbb{E}[X] < a) < exp(-ca^2)$. We get a new inequality by minimising the Moment Generating Function in the chernoff bounds for the bernoulli distribution: $P(\sum_{i=0}^{n} X_i - \mathbb{E}[X] < a) < exp(-d(a, \mathbb{E}[X]))$, where

$$d(p, q) = plog(\frac{p}{q}) + (1 - p)log(\frac{1 - p}{1 - q}) \tag{4}$$

is the KL divergence for bernoulli distributions. The arm/ action is now chosen as follows:

$$A_t = \underset{1<=a<=K}{\arg\max} \max\{q \in \Theta : N[a]d(\frac{S[a]}{N[a]}, q) <= log(t) + clog(log(t)) \tag{5}$$

As in Thompson Sampling(studied in class), the KL-UCB is order optimal for bernoulli arms. Note that even thompson sampling uses the fact that the beta distribution is the conjugate prior of bernoulli. We shall hence move forward with UCB as we do not know the distribution of stocks.

2.4 Portfolio Selection

The Online Portfolio Selection Problem (OPSP) 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.Hence, we can use multi-armed bandits to model this problem. 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. Modified versions of the above algorithms to include financial metrics can help solve this problem.

2.5 Stocks and Bandits

The bandit learning algorithm will choose whether or not to buy a single dollar's worth of stock and at the end of the day it sells the stock and observes the profit. We assume there are no brokers involved and hence the price seen by the algorithm is the price at which the stock is obtained. The profit per day is the reward for that round for a particular stock. We also ignore the effects of the algorithms decisions in the market.

2.6 Risk aware algorithms

Bandit algorithms can also be implemented to take risk into account. For this case, CVaR (Conditional value at risk) can be used since it can be used to assess financial risk. Thus the arm chosen is a linear

combination of the expected reward and the associated Conditional Value at Risk (CVaR). For this problem, the goal is to devise algorithms that are entirely distribution oblivious, i.e., the algorithm is not aware of any information on the reward distributions, including bounds on the moments/tails, or the suboptimality gaps across arms.

2.6.1 CVaR Concentration

If X is a random variable then Value at Risk (VaR) is defined as follows:

$$VaR_\alpha(X) = \inf\{y : P(X < y) < \alpha\} \tag{6}$$

The expected value of X which is greater than VaR_α is known as CVaR.

$$CVaR_\alpha(X) = \mathbb{E}[X|X > VaR_\alpha(X)] \tag{7}$$

The CVaR at each step is calculated using the following result:

$$\hat{c}_{n,\alpha} = \frac{1}{n(1-\alpha)} \sum_{i=1}^{n} X_i \mathbb{I}[X_i > X_{\lfloor n(1-\alpha) \rfloor}] \tag{8}$$

where I[.] is an indicator function. [2] proved that this estimator has exponentially decreasing bounds.

3 Implementation details

In this section, we briefly explain the implementation of the model. The bandit algorithms have been implemented in both risk aware and non-risk aware setting.

3.1 Dataset

The model has been applied on 10 years stock data (2003-2013) of the following fortune 500 companies:

AAPL AMZN COST GS JPM MSFT TGT WFC WMT

3.2 UCB and CVaR

Instead of optimising over (3), we choose the arm as

$$A_t = \arg \max_{a \in arms} \text{UCB}(a, t) + \gamma CVaR(a, t|A_i, i < t)$$

Here, γ controls the tradeoff between risk and reward. [1] use a similar method for portfolio selection on a subset of the original assets chosen by a minimum spanning tree. [2] use a similar hyperparameter, but in an exploration setting. They hypothesise that there must be a trade off between minimising risk and regret. However, our experiments show that minimising risk instead decreases the regret in a financial setting.

4 Results

4.1 Results for the standard algorithms with Bernoulli Arms

For arms with Bernoulli rewards, the following figures show the performance (in terms of regret) of various bandit algorithms.

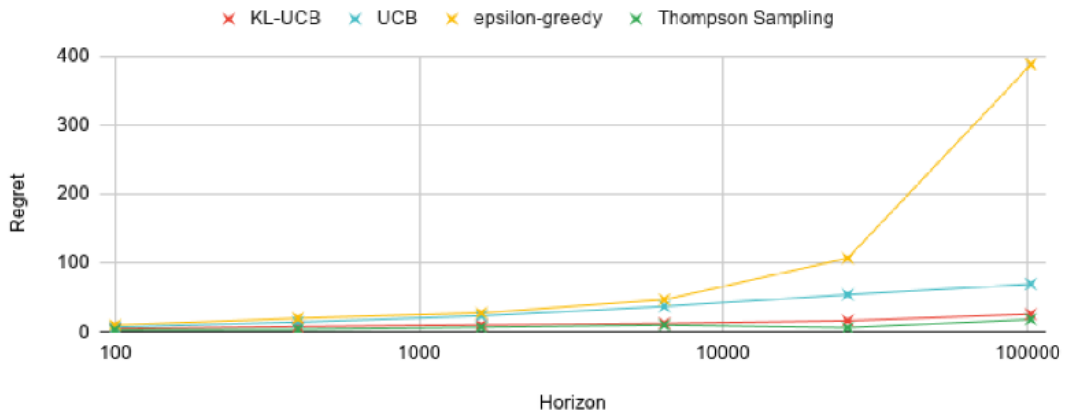
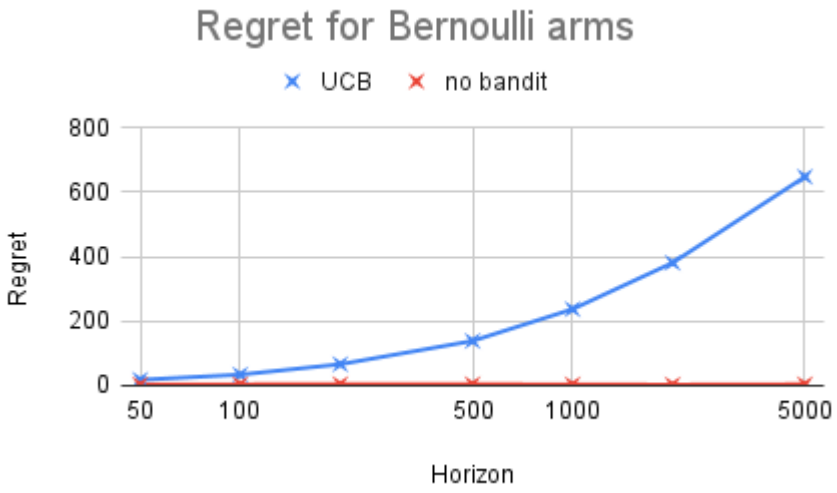


Figure 1: Regret plots on a logarithmic time scale, for 5 arms. As expected, KL-UCB and Thompson sampling are both order optimal for Bernoulli Arms. Epsilon greedy samples randomly w.p. ϵ and greedily rest of the time

Here, note that we also introduce an improvement into the UCB algorithm. We include an extra step in the UCB algorithm, such that every arm is pulled at each time instant, but only the best one is added to the regret. This way, we do not use bandits, and hence, attempt to improve the regret for Bernoulli arms. The following plot shows the results of implementing said algorithm on Bernoulli arms:



4.2 Results for standard algorithms with Stock arms

Using the aforementioned UCB algorithm, along with the improvement whereby each arm is pulled at every time instant, so as to increase the number of pulls for each arm at each time instant, we obtain the results as shown in figure (2). Here, we also compare these two algorithms (UCB with and without bandit), with the CVaR algorithm as presented in [2], (again, both with and without the bandit).

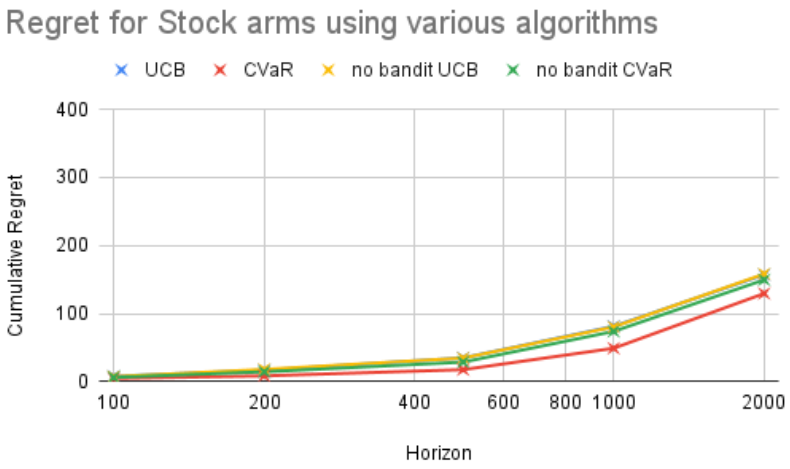
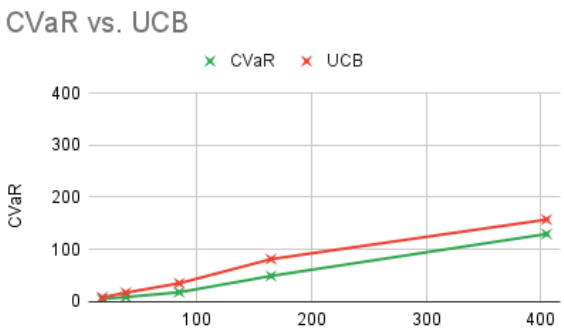


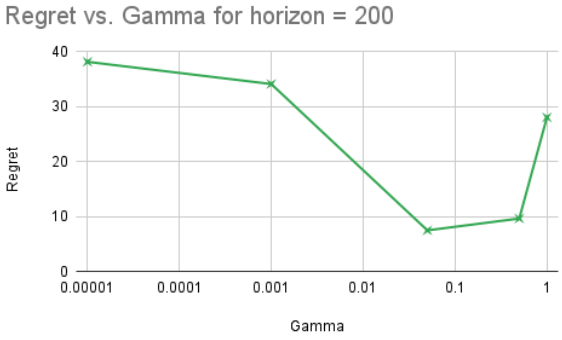
Figure 2: Regret for Stock arms using multiple algorithms

Here, the values for α and γ we use in the CVaR algorithm are 0.96 and 0.03, respectively.

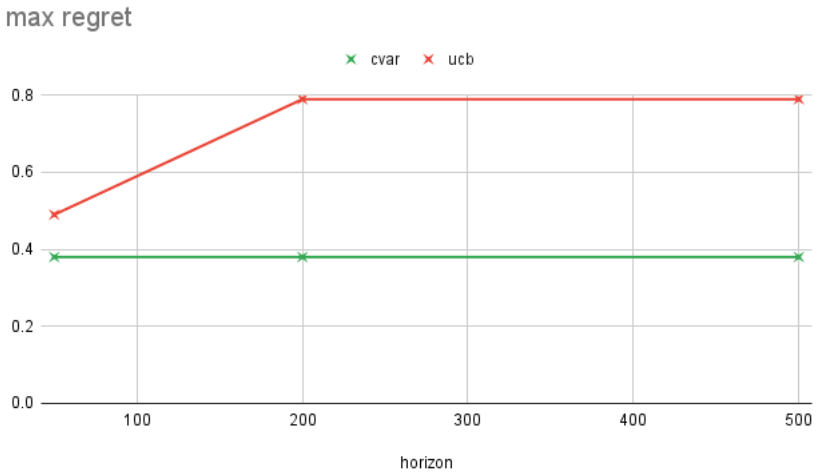
Clearly, there is not too much of a difference between the UCB algorithm and the CVaR algorithm, in terms of regret for larger values of horizon. However, implementing these algorithms with regret does give a considerable improvement in the implementation.

The comparison between the CVaR algorithm and the UCB algorithm may be seen through the plot in figure (3), which clearly shows that the UCB regret increases linearly with the CVaR Regret, at higher horizons.





This distinction between the UCB algorithm and the CVaR algorithm can also be observed through the maximum regret observed for each of these algorithms at various values of horizon. The plot for maximum regret for both the UCB and the CVaR method is shown in the graph in figure (5).



Clearly, for higher horizon values, the maximum regret for the UCB algorithm at any given seed increases and goes much higher than that of the CVaR algorithm.

Then, we also looked into the variation of regret with γ, which is just the factor by which the CVaR value is multiplied by, whilst adding to the UCB. Upon varying γ from 0 to 1, we find that there is a clear minima in regret at higher values.
However, the maxima of the curve obtained at $\gamma = 0$, still gives a lower regret than that of the UCB algorithm. The plot shown in figure (4) gives the value of regret for various values of γ, with a horizon set at 200.

5 Modified Martingale Trading Strategy

Now we provide bounds for the reward obtained in a toy case, given any probability of profit, $p > 0.5$. Say that you invest r amount of money each time you enter a trade. Let the buying and selling price per stock be given by $pps(b)$ and $pps(s)$ respectively. We assume that the opening and closing prices are i.i.d. for simplicity(a weaker assumption of profiting in/winning a trade being i.i.d. also works). Then the number of stocks is $ns = \frac{r}{pps(s)}$ and if we assume profit, its expectation is given by

$$\mathbb{E}[ns(pps(s) - pps(b))|\text{profit}] = r\mathbb{E}\left[\frac{pps(s)}{pps(b)} - 1\right] = r\gamma > 0 \tag{9}$$

Whereas assuming loss, the above expectation is at least(worst case) -r. Assume a worst case analogue of the above values as an arm, i.e., $X_i = 1+\gamma$, w.p. p and -1 otherwise. We show that the probability of net profit after n trades has the following bound[1]:

$$\mathbb{P}[\text{profit}] = \mathbb{P}[\sum_{t=1}^{n} X_t < n] \leq exp\left(-\frac{2\gamma^2(1.5-\gamma^2)n}{(2+\gamma^2)}\right) \tag{10}$$

This says that the probability of performing suffering a loss decreases exponentially with time. However, it is not good in the short run(for a trader doing less trades, for example). Consider two trades: the first one losing and the next one winning(earning profit). The expected value of total profit will be at least $r(\gamma - 1)$. This requires γ to be greater than 1, or $\mathbb{E}\left[\frac{pps(s)}{pps(b)}\right] > 2$, which is highly unrealistic. We hence need a different strategy to utilise our budget, r. Now we consider a case where we trade with budgets

$$b_0, b_0\alpha, b_0\alpha^2, ...$$

[1]due to time and space constraints, the proof is provided as an informal appendix here

until we 'win'. The nice thing about this algorithm is that given the rewards for previous runs, the expected reward for the current run remains the same[2] for a given α (formally, for α given by the solution of $(1+\gamma)\alpha^n > \sum_{i=0}^{n} \alpha^i = \text{constant}$). This is popularly know as the martingale strategy. There is a problem with this, however: the budget exponentially increases until winning. We therefore meet in between the two mentioned strategies as a compromise. We divide our initial budget, r into $b_0, b_0\alpha, b_0\alpha^2, ..., b_0\alpha^n$, and in each run, we trade until we reach n or win. This enjoys a higher probability of winning a run, however, the expectation of profit per run decreases to $\gamma b_0 < \gamma r$. The nice properties of still holds as we are effectively doing that with a higher probability(given by 1-$P(\text{losing a run}) = 1-(1-p)^n$). A final remark is that we would need p necessarily > 0.5 for to hold. However, this is not needed in the modified martingale case: we can choose n and b_0 appropriately for it[3].

6 Some more Remarks

- The risk aware bandit model surprisingly outperforms UCB and no-bandit cases. This might be because lowering risk is more informative in financial settings.
- Since the stocks are de-trended, the mean rewards are quite close zero(see Figure **??**). Hence, the sub-optimality gaps for each arm is very low. This may be one of the reasons UCB performs poorly. Note that higher order regression would also give poor results to predict stocks due to this. This provides motivation to look for sequential networks to solve the problem.
- Note that UCB and other bandit algorithms assume IID(static) nature of arms and hence may fail to capture the time varying stochasticity of stocks. Hence an algorithm like Exp 3, working under the stronger assumption of an adversary might give better results.

7 Conclusion

In this project, we compared the performance of both risk aware UCB and non-risk aware UCB with and without bandits across different horizons in terms of regret.Surprisingly, the risk aware bandit outperformed the bandit without risk. We also used hyperparameter tuning to tune the value of γ to obtain the minimum regret.

8 Future Work

This model can be improved by adding a classifier with reasonably high accuracy which predicts whether the prices are going up or down. This gives the opportunity to short stocks as well thus giving potential for higher reward. External features/parameters can be added to model this classifier thus increasing the maximum reward possible by taking advantage of variability.

References

[1] X. Huo and F. Fu. Risk-aware multi-armed bandit problem with application to portfolio selection. *Royal Society open science*, 4(11):171377, 2017.

[2] A. Kagrecha, J. Nair, and K. Jagannathan. Distribution oblivious, risk-aware algorithms for multi-armed bandits with unbounded rewards. *arXiv preprint arXiv:1906.00569*, 2019.

Note: References for other algorithms are provided in github for conciseness

[2] Hence the sequence of rewards is a martingale process($\mathbb{E}[X_n|X_m, m<n] = (1+\gamma)\alpha^n - \sum_{i=0}^{n}\alpha^i = \mathbb{E}[X_{n-1}]$, which is γb_0 since distributions are iid)

[3] again, shown here qualitatively