Multi-Armed Bandit: A Recommendation for Neural Network Learning Algorithms

T. Caputo, P. Lenharth, R. Ludington

May 3, 2018

Abstract

The multi-armed bandit problem has recently gained popularity as a model for studying the tradeoff between exploration and exploitation in reinforcement learning. While many algorithms are understood well theoretically, empirical results suggest that relatively naive algorithms outperform more theoretically sound ones. This paper provides such evidence for a few popular algorithms. The short-term and long-term behaviors of several popular Multi-Arm Bandit solutions were also examined. This examination is motivated by the wide variety of uses for reinforcement learning. Applications sometimes call for an algorithm which learns quickly, but sacrifices optimal long term performance. Two important observations can be made from our results. First, our paper reinforces the results of previous papers which found that the ϵ-greedy and Softmax algorithms outperform algorithms such as Thompson Sampling [1][2]. Secondly, the inherent tradeoffs with learning algorithms lead to varying performance in the short term, within 200 iterations and long term, at 1000 iterations. Optimal algorithm choice varies greatly across the parameters tested. Epsilon-Greedy and Softmax are often the optimal algorithms.

1 Introduction

The multi-armed bandit problem was introduced by Robbins in 1952 [2] and has gained significant attention in machine learning applications. The name for the model comes from the one-armed bandit which is a colloquial name for a slot machine. The problem poses a situation where a gambler walks into a casino and sits down at a row of slot machines. Each one produces a random payout according to some distribution which is unknown to the gambler. Because the distributions are unknown, the gambler must learn about the distributions by experimenting. As the gambler begins to pull arms and receive payouts, they face the inherent tradeoff. They must either try to exploit their existing knowledge and pull arms that have previously paid out the most to earn more in the short term, or explore alternative arms in order to learn the true distributions and receive the highest payout in the long term. They need to develop a sequential strategy to balance exploitation and exploration to maximize their payout. We can formally describe the multi-armed bandit in this way:

There are K probability distributions $< D_1, ..., D_K >$ with corresponding expected values $< \mu_1, ..., \mu_K >$ and variances $< {\sigma_1}^2, ..., {\sigma_K}^2 >$. Each distribution corresponds to a slot machine, and the distributions are initially unknown to the gambler. At each turn $t = 1, 2, \ldots, T$ the gambler chooses an arm with index $j(t)$ and receives a reward $r(t) \sim D_{j(t)}$ [2].

As a performance metric, we chose to use per-period regret over a time period t, where regret is simply the normalized difference between the reward of the optimal action and the cumulative average reward. More formally, we can define regret R at each period t as

$$R_t = \frac{(\mu^* - \mu_t)}{\mu^*},$$

where $\mu^* = \max_{i=1,..,k} \mu_i$ is the expected reward from the best arm.

The choice of the multi-arm bandit problem is motivated by its application to machine learning as a way to effectively measure the performance of learning algorithms. Due to a wide number of applications, the aim of this study is to inform the appropriate learning algorithm choice given a Gaussian or Bernoulli distribution and whether long or short-term performance is desired. We focus on the latter as some specific applications such as clinical trials have a limited sample size thus have a need for short term performance.

2 Evaluating Algorithms for Multi-Armed Bandit

An extensive empirical study on multiple algorithms was done by Vermorel and Mohri (2005) [1], but has since been followed by a series of new studies. These studies conclude the initial finding that naive algorithms such as ϵ-greedy often outperform more theoretically sound ones. A follow

up study done by Kuleshov and Precup (2014) found that variance was a largely influential factor in determining the success of a learning algorithm [2]. While many analyses were done on algorithms such as UCB1, recent attention in literature has shifted to the Thompson Sampling method, which is described in detail by Russo et.al (2018) [5].

In this study, we include two types of distributions for the slot-machines. The first type is the Gaussian distribution whose probability density function is defined as

$$f(x|\mu, \sigma^2) = \frac{e^{(x-\mu)^2/2\sigma^2}}{\sqrt{2\pi\sigma^2}}.$$

The second type is the Bernoulli distribution whose probability mass function is defined as

$$f(k; p) = \begin{cases} p & \text{if } k = 1 \\ 1 - p & \text{if } k = 0, \end{cases}$$

where k is an outcome.

We picked these two distributions because they encompass a wide variety of applications such as the "success/failure" of advertising.

In order to get an accurate result from our experiments, we averaged over 100 runs. This reduces error from random chance to get an overall comparison of the algorithms.

2.1 Algorithms

The formal definitions for the algorithms used are well known, with the exception of the vary-greedy algorithm. The vary-greedy algorithm was created as an efficient way to improve the epsilon-greedy algorithm. The formal definitions are detailled below:

2.1.1 ϵ-greedy

The ϵ-greedy algorithm is particularly popular because of its simplicity. For the same reason, it is often considered to be a naive algorithm. At each round $t = 1, \ldots, T$ the algorithm selects a random arm with probability ϵ and selects the arm with the highest empirical mean with probability $1 - \epsilon$.
Given empirical means $\mu_1(0), .., \mu_N(0)$,

$$p_i(t+1) = \begin{cases} 1 - \epsilon & \text{if } i = \arg\max_{j=1,..,N} \mu_j(t) \\ \epsilon/(N-1) & \text{otherwise.} \end{cases}$$

Two decay rates were chosen for ϵ. The first was exponential decay and the second logarithmic decay. Each decay mode had a scaling parameter.

2.1.2 Softmax (Boltzmann)

The Softmax algorithm is based on the Boltzmann distribution, where the probability of arm n being chosen is given by

$$P(n) = \frac{\mu_n^2 e^{\mu_n^2}/T}{\sum_{n=1}^{N} \mu_n^2 e^{\mu_n^2}/T},$$

where μ is the sample mean.

2.1.3 Thompson Sampling

This method was first proposed in 1933 for clinical trials as a method of allocating experimental effort, but has since seen a surge of interest as a bandit solution [5]. We specifically define this algorithm for Bernoulli distributions as it was not experimentally tested for Gaussian distributions. This method assumes the distributions of the means to be Beta(α, β) distributions with $\alpha, \beta = 1$ initially. There are N possible actions where each action n produces an expected reward of θ_n found by taking one sample from each Beta distribution. At each turn, the algorithm chooses the arm $x(t)$ with the highest mean $\theta^* = \max_{i=1,,N} \theta_i$. It then updates the distributions according to the rule

$$(\alpha_n, \beta_n) \leftarrow \begin{cases} (\alpha_n, \beta_n) & \text{if } x_t \neq n \\ (\alpha_n, \beta_n) + (r_{x(t)}, 1 - r_{x(t)}) & \text{if } x_t = n, \end{cases}$$

where $r_{x(t)} \in \{0, 1\}$ is the reward at turn t from machine $x(t)$.

2.1.4 Vary-Greedy

This algorithm is an extension of the ϵ-greedy algorithm, and is our attempt to improve Epsilon-Greedy. The formal definition is identical to the standard ϵ-greedy with one additional component. The algorithm will no longer include arms whose rewards are 1 or more standard deviation less than the mean θ^* of the experimental best arm while exploring.

$$p_i(t+1) = \begin{cases} 1 - \epsilon & \text{if } i = \arg\max_{j=1,..,N} \mu_j(t) \\ 0 & \text{if } \mu_i + \sigma_i < \mu^* \\ \epsilon/(N - 1 - k) & \text{otherwise.} \end{cases}$$

where N is the number of Arms and k is the number of Arms whose average is 1 standard deviation below the mean of best machine

2.2 Short term vs Long term behavior

We define short term as the first 200 turns and long term as the asymptotic behavior as t approaches infinity, truncated at $t = 1000$ due to computational constraints. From testing, we have seen that an algorithm that is

optimal in the first 200 turns can be outperformed by another algorithm in the long term. For this reason we chose to look into short term behavior as some applications have a limited sample size thus restricting their total number of turns. An example would be clinical trials having a limited amount of experimental drugs.

An example of the trade-off between short-term and long-term performance can be seen in Figure 2. This is a sample regret plot where Vary-Greedy has better performance in the short term, but is beaten by Softmax in the long term.

2.3 Spread-out distributions vs Overlapping distributions

We define overlapping distributions as a set of distributions where the differences in the means are less than the standard deviations (i.e. distributions encompasses one another). We define spread-out distributions as a set of distributions where the differences in means are greater than the standard deviations (i.e. distributions don't touch one another).

An example of how the distributions can affect the performance of the algorithms can be seen below in Figures 1 and 2. In Figure 1, the algorithms are evaluated using spread-out distributions and Vary-Greedy is optimal. In Figure 2, the algorithms are evaluated using overlapping distributions and Softmax is optimal.

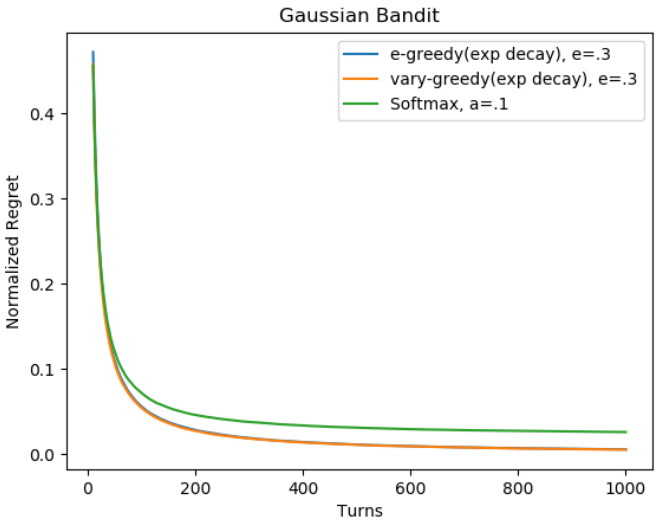

Figure 1: Vary-Greedy performs well in spread-out distributions

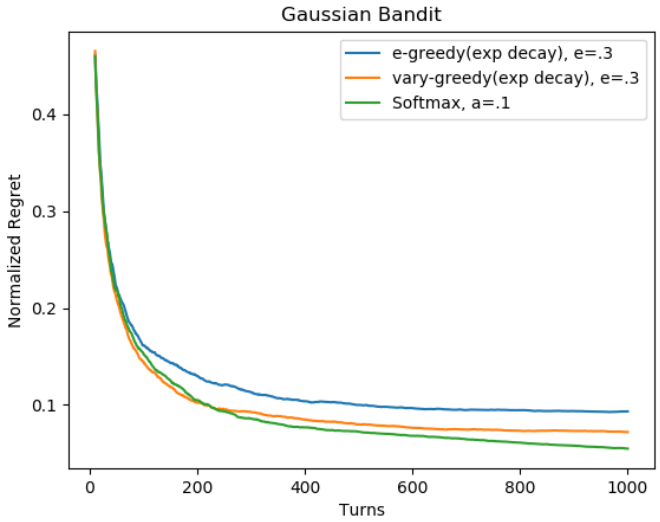

Figure 2: Softmax beats Vary Greedy in overlapping distributions

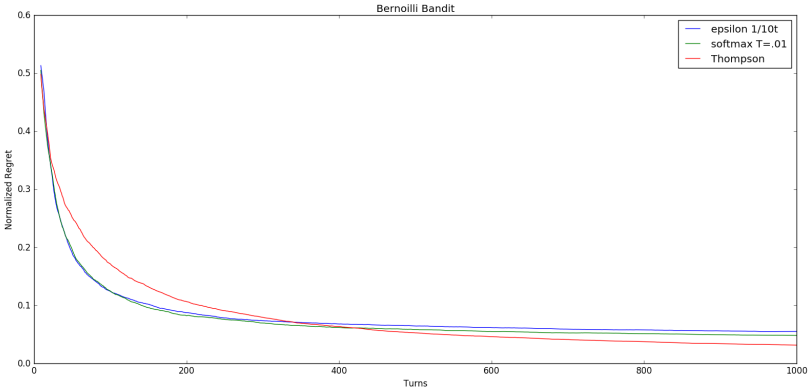

Figure 3: Thompson beats simple algorithms in spread-out distributions

3 Results

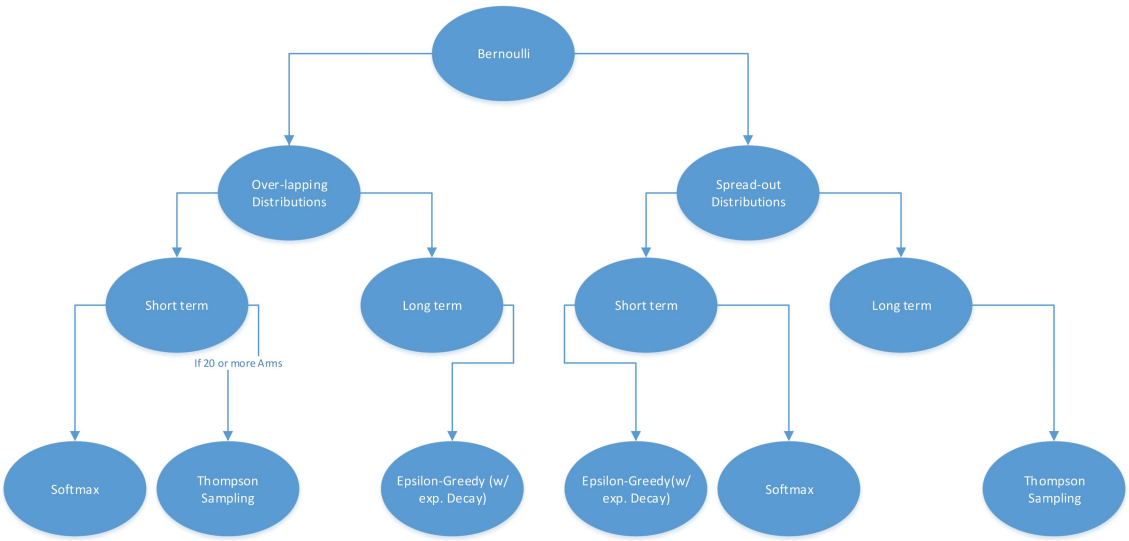

Figure 4: Tree Diagram depicting optimal algorithms based on each parameter tested for Bernoulli distribution

Optimal Algorithms for Gaussian Distributions

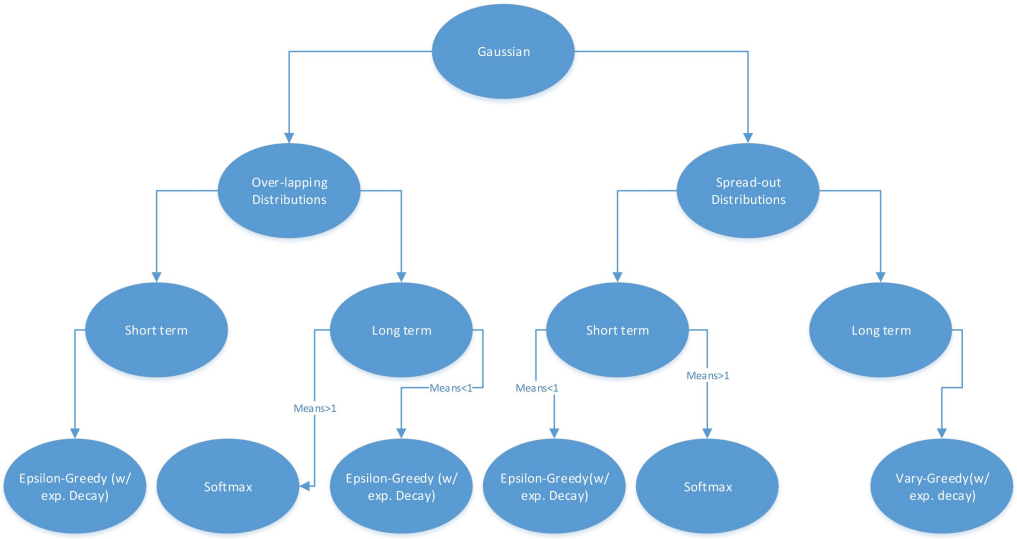

Figure 5: Tree Diagram depicting optimal algorithms based on each parameter tested for Gaussian distribution

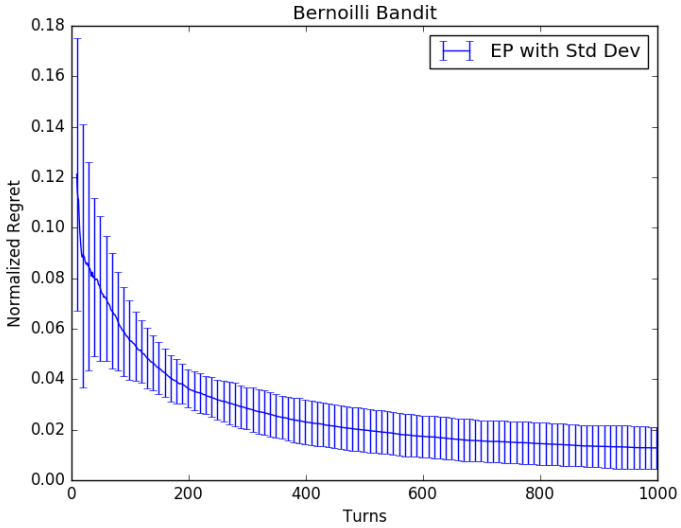

Figure 6: Standard deviation of Epsilon-Greedy over 100 runs

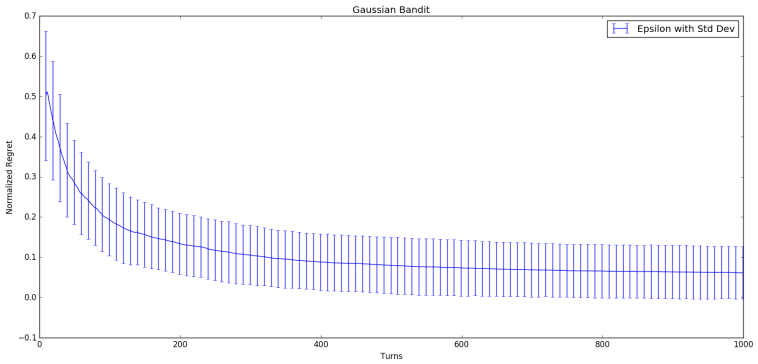

Figure 7: Standard deviation of Epsilon-Greedy over 100 runs (Gaussian)

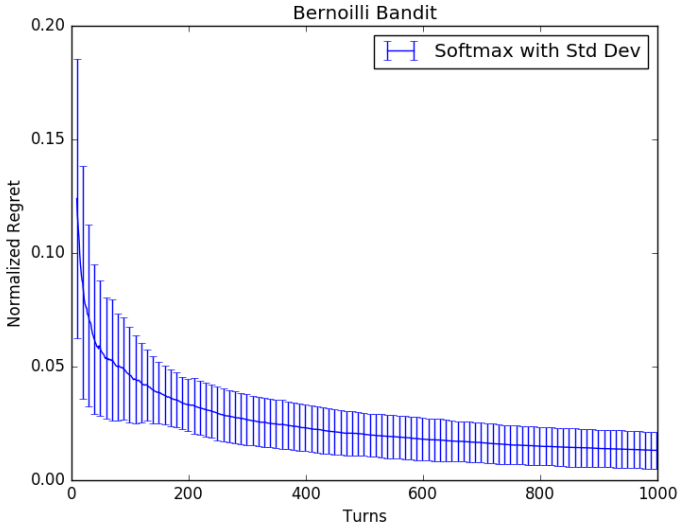

Figure 8: Standard deviation of Softmax over 100 runs

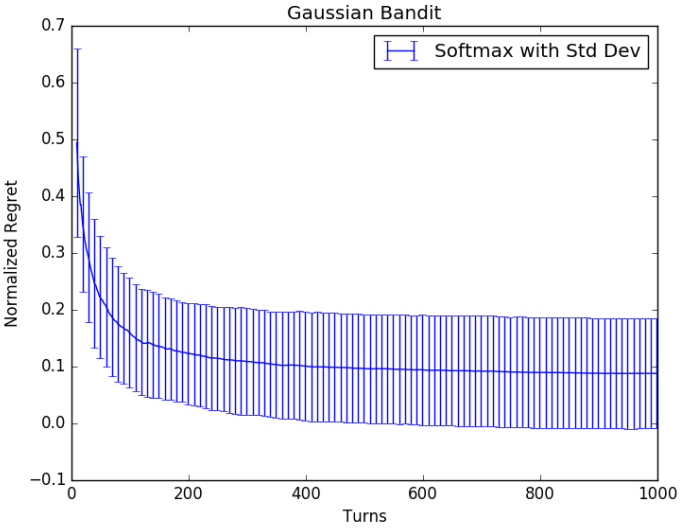

Figure 9: Standard deviation of Softmax over 100 runs (Gaussian)

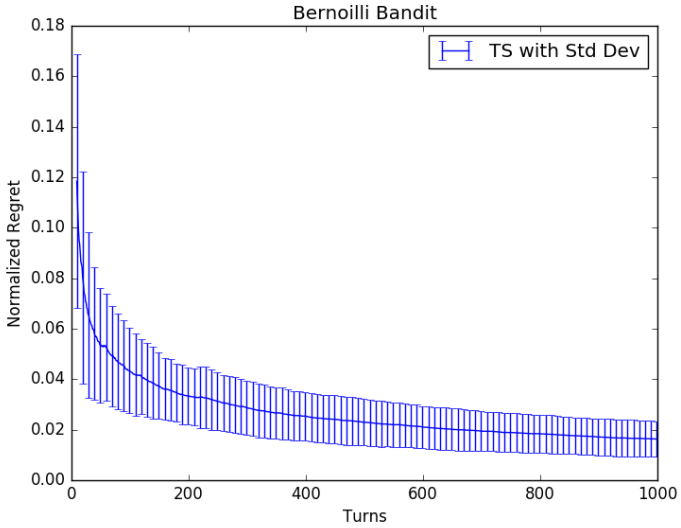

Figure 10: Standard deviation of Thompson Sampling over 100 runs

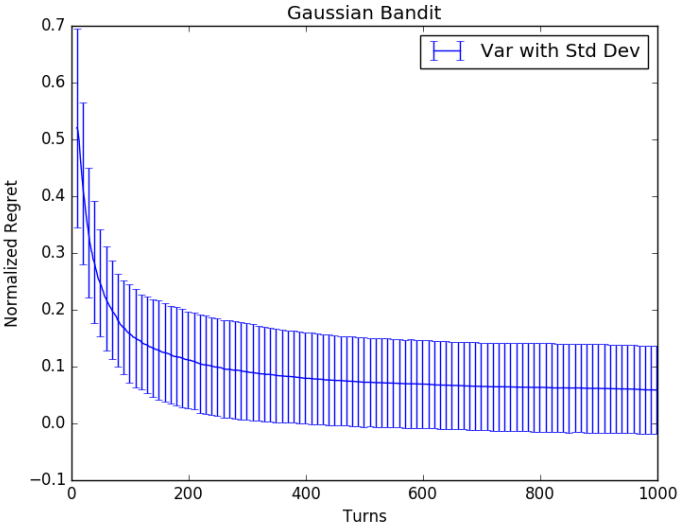

Figure 11: Standard deviation of Vary-Greedy over 100 runs (Gaussian)

4 Conclusion

Softmax and Vary-Greedy generally outperformed more complex algorithms like Thompson Sampling. This reinforces previous findings that suggest naive approaches are often superior [1][2]. Thus empirical data for algorithms is needed to inform algorithm choice and theoretically sound algorithms may still be worst in practice. While an analysis of many distributions would be needed to draw broader conclusions, the two distribution types had little impact on performance (figures 4,5). It may be that Gaussian and Bernoulli distributions give similar results as the repeated sampling of a Bernoulli distribution is a binomial distribution, which is essentially a discrete Gaussian. Other distributions such as the exponential may have different results.

The data from the distributions is different, however, in one key way: the variance of the convergence. The standard deviation of the averaged runs was obtained and plotted for each algorithm and distribution in figures 6-11. As can be seen, the standard deviation for the algorithms under the Gaussian distribution is much larger than that of the Bernoulli. A likely candidate for this difference is the continuous nature of the Gaussian. This leads to a higher chance that small sample size will result in an average that is much less then expected. If this happens to the best arm, it is likely to converge to a different arm in epsilon-greedy, especially if epsilon is decaying quickly. In Softmax, this will lead to a much smaller chance that the best arm is chosen, and so many trials might be needed to correct the initial bias. For epsilon greedy, this can be solved through a slower decay, at the cost of longer convergence times. In the case of Softmax, attempting to reduce this likelihood will cause worse convergence due to the arms having more equal probabilities. It is also important to note that the graphs were obtained for overlapping distributions, where such initial bias is more pronounced.

In addition to the effect on variance of performance, overlapping distributions also affect algorithm choice. Thompson Sampling and Vary-Greedy both benefit more from distributions that are spread-out than Epsilon Greedy and Softmax in the long term. This is due to both algorithms not testing clearly inferior distributions. In the case of Thompson sampling, after a few turns, it is unlikely that a machine with a low mean will have a beta distribution sample larger than the best machine. This removes it from consideration, reducing the number of arms to choose from. Similarly, Vary-Greedy will drop machines that are far below the best, which is easier if the distributions are spread-out. If averages are close, or we are interested in performance in the short term, these advantages are eliminated. Thus, quickly exploiting what you know is the better option. This is shown by the better performance of the nave algorithms in the short term, and with overlapping distributions (figures 4,5). As mentioned above, these algorithms do not always converge to zero, easily getting caught in local minima. Therefore, if the best long-term performance is desired, Thompson sampling may be a better pick, even with overlapping distributions. This is due to it having a strong likelihood that the regret will converge to zero [5]. Longer runs would be required to see if Thompson sampling does truly perform better with overlapping

distributions in the long-term. For Gaussian, a slower decay rate may be similarly better. Such additional exploration will slow learning but reduce the chance of non-optimal convergence.

References

[1] Vermorel, Joannes, and Mehryar Mohri. "Multi-armed bandit algorithms and empirical evaluation." European conference on machine learning. Springer, Berlin, Heidelberg, 2005.

[2] Kuleshov, Volodymyr, and Doina Precup. "Algorithms for multi-armed bandit problems." arXiv preprint arXiv:1402.6028 (2014).

[3] Raja, Sudeep. Multi Armed Bandits and Exploration Strategies. Multi Armed Bandits and Exploration Strategies Sudeep Raja MS/Phd Student at UMass Amherst, 28 Aug. 2016, sudeep-raja.github.io/Bandits/.

[4] Multi-Armed Bandits. The Data Incubator MultiArmed Bandits Comments, blog.thedataincubator.com/2016/07/multi-armed-bandits-2/.

[5] Li, Lihong. "A contextual-bandit approach to personalized news article recommendation." Proceedings of the 19th international conference on World wide web. ACM, 2010.

[6] Russo, Daniel, et al. "A Tutorial on Thompson Sampling.", 2017.