Prediction Markets for Multi-Agent Multi-Armed Bandits

Michael Zhang, Catherine Tu[*]
Harvard University
Cambridge, MA 02138
CS 136: Economics and Computation, Fall 2018

Abstract

The multi-armed bandit problem is a classic problem demonstrating the exploration-exploitation dilemma in decision making under uncertainty. Here we consider the premise of multiple agents interacting to make an aggregate decision, hoping to balance individual agent reward optimization with a higher-level aggregate goal–in this case learning the true reward distribution of a given bandit. We explore the use of market design mechanisms–specifically cost-based automated market making scoring rules–to balance individual agent exploitation and early-onset bias with aggregate optimization, proposing a multi-agent algorithm and demonstrating in preliminary settings how constructing a prediction market with multiple interacting agents leads to model-based learning of the bandit setting with few individual agent bandit pulls.

1 Introduction

Multi-armed bandits (MABs) are in interesting class of problems related to decision making and reward maximization. The space of such algorithms demonstrates a trade-off between accuracy and contextual understanding with training complexity and data magnitude. On one hand, reinforcement learning techniques have recently garnered their fair share of attention in the news and academic literature for learning incredibly complex functions mapping out equally impressive state, action, and reward spaces, all at the cost of a momentous requirement for data. On the other, simply trying to learn the association between unknown rewards and potential actions make, as is the case in MABs, makes up a relatively more quick to learn and data-frugal class of algorithms.

For this reason we use the multi-armed bandit case to motivate further exploration, proposing the *multi-agent multi-armed* bandit (MAMA) problem. While machine learning with multiple models is far from a unique or new concept, recently there has been a resurgance in more specifically “multi-agent" settings. As opposed to more classical ensemble learning methods such as random forest or boosting, which essentially aggregate the outputs of multiple classification or regression models in a weighted expectation scheme, the notion of an “agent" seems to suggest greater autonomy and perhaps flexibility. There is a natural tie-in with economics as notions such as designing “incentive" or maximizing “reward" are substituted in favor of “gradient descent" or “least squares estimation" when describing an optimization problem. Indeed, this problem may not always be convex in the agent case, and when dealing with multi-agent bandit settings, we can model the problem accordingly as an agent trying to explore various possible states to elicit a max reward over a limited exploration quota.

[*] Apologies are college emails are too long and mess up the author centering... Authors can be reached at `michael_zhang@college.harvard.edu, catherine_tu@college.harvard.edu`

Preprint. Work in progress.

1.1 Multi-armed Bandits

Briefly, bandits can be imagined as slot machines with multiple levers or arms, each which can be pulled and probabilistically outputs some reward. In the classic problem context, we would like to figure out which arm to pull to give us the highest expected reward. If the bandits are relatively few-armed, one way to do this would be to pull every arm, perhaps multiple times in a stochastic setting, and then establish some sufficient statistic to inform one of the most profitable arm. However, we can quickly imagine settings where this does not hold up in the real world. In a classic casino case, it costs money to pull each arm, and often such payments are rigged so that gamblers can expect net loss. Accordingly, we might want to figure out the probabilistic distribution underlying the reward mechanism of each arm with the least number of actual pulls from our own perspective as possible. For multi-armed bandits and other related problems, we can use the *Thompson Sampling* algorithm. However, in impending sections we introduce motivation for a multi-agent multi-armed bandit scenario, which in turn encourages the use of market design along the lines of information elicitation and prediction markets. We first introduce the problem setup and related concepts, present our models and algorithms, display some preliminary results, and conclude with final discussions.

1.2 Problem Setup and Motivating Example

Beyond the general class of multi-armed bandit problems, we motivate our project in a multi-agent setting where free exploration of the arms is limited. Noteably, we do not work with context, so rewards always follow some pre-set underlying probability distribution. Additionally, we are not just interested in figuring out the best arm to pull at any round, but rather getting an accurate representation of the entire joint distribution. One example where this might apply to the real world involves health-care and clinical decision-making in the hospital. We can assume that a patient can be treated with one of any number of drugs, and we may have some underlying knowledge but not full certainty of the outcome of each treatment. Related to the fundamental issue of causal inference, we cannot simply administer multiple drugs to the patient and compare the results. Slightly more subtly, it may not be in our best interest to try to explore this action space of arms freely, as we also care about the patient's well-being as well. Perhaps we face this decision after treating some subset of the wider population, and acknowledge that although we have the knowledge to treat patients that look like those we have treated before, we would ideally like to not limit ourselves only to those that have walked through our doors before and gain some information and curative edge in an effort to treat more individuals. Unfortunately, we cannot merely treat the next person we see with some arbitrary concoction of drugs we have not seen before to gain information on that combination of states. In addition, the potential state space is perhaps infeasibily large, and so we would not be able to reasonably explore all conditions even without regard to patient well-being.

However, what we can benefit from is the at least somewhat reasonable assumption that there are multiple care-takers with the same motivations as us acting in the space as well. Even more, we assume that they also have some experience taking care of a certain subset of people under their belt, and that the patients that they have seen may not be representative of the patients we have seen. Accordingly, swapping information and learning how to treat more people, perhaps through a market mechanism, would be in our best interest. Accordingly we may model this as a heterogenous market, where individual hospitals or care-takers participate as agents each with their own trained priors, and we want to aggregate this info to inform better decision-making at the individual level. We motivate two ideal outcomes: (1) the information aggregation of the market yields stronger and more accurate beliefs than any single individual, and (2) the aggregation of information through a market place circumnavigates issues where directly observing the same events that caused others to learn their priors may not be possible (examples might be patient privacy, or data non-interoperability).

These types of problems may generalize to the bandit case. Faced with some unknown distribution of rewards over some set of actions, we have perhaps explored a fair bit and built a reasonable set of priors that takes care of all states and actions observed already. However, we also acknowledge that our history may not be representative of the entire state-action-reward space, and would ideally like to build up to some fully generalized joint distribution capable of explaining relationships in actions and states we have not seen. We are cautious to explore on our own, but have the benefit of information lying in the form of priors in the agents around us waiting to be aggregated.

1.3 Prediction Markets

One tool at our disposal would be a *prediction market*, an information elicitation mechanism that aggregates information among its participants, who may update their prior beliefs in accordance with the exchange of information and achieve the objective of selecting the optimal arm to pull. While historically used heavily with human participants in the context of forecasting geopolitical events or other polling contexts, we consider their structure in a machine learning setting as a bandit-solving algorithm. To replicate the heterogenous nature of participants in a market, we consider a setting where various algorithms have trained on disparate sets or partitions of a larger data set, and motivate trying to learn beyond our own training examples by observing the beliefs of other models without actually having access to the data they trained on.

In a more traditional sense, prediction markets are used to aggregate information from other market participants about future events. By eliciting information from a crowd of individuals, the market can harness the knowledge of all the participants, which might allow making a more informed decision (1). This is especially useful in the context where each agent has limited knowledge of a subset of the data and thus cannot make an informed choice by themselves. We will explore how the prediction market mechanism can facilitate the information exchange among such a population of agents. In addition, we note how prediction markets combined with Thompson Sampling can extend the algorithm's effectiveness to realizing a full generative model and joint distribution of underlying parameters, as opposed to typically just eliciting some probability belief of the best arm to pull. We now introduce our bandit market model, going further into the exact type of prediction market mechanism and its desirable properties.

2 Model

We consider the following problem setup inspired by a classic MAB problem. Let A denote the set of independent arms. From the perspective of a single player i, pulling arm $A_j \in A$ yields i.i.d. random rewards under a distribution specified by unknown parameter θ_j. For simplicity we model reward function R such that

$$R(A_j) \sim \text{Bern}(\theta_j), \text{ s.t. } R(A_j) \in \{0, 1\}$$

which we refer to as the Bernoulli bandit problem (BBP) (2). We are interested in learning the joint distribution $\Theta = (\theta_1, \theta_2, \ldots, \theta_n)$, where $n = |A|$. To motivate a prediction market, we now consider the setting where we have some total set of all outcomes Ω explained by Θ, but individual agents i are only exposed to some collection of events $\mathcal{F}_i = \{\omega_m, \ldots \omega_n\}$ where all elements in $\mathcal{F}_i \in \Omega$. Each event is a tuple $\{A_j, R(A_j)\}$, which denotes that pulling arm A_j yields reward $R(A_j)$. Importantly, we partition the space such that no two agents learn off overlapping events, so $\mathcal{F}_i \cap \mathcal{F}_j \in \emptyset$ for $i \neq j$.

Accordingly, a brief sketch of our algorithm is as follows. Each agent A_i is coupled with a bandit B_i that behaves according to $\mathcal{F}_i$. The agent solves for the underlying parameters of its own bandit, obtaining a set of priors. These beliefs are then tapped in a prediction market made of multiple agents, and the corresponding quantities bought under a log-scoring cost-based automated market maker mechanism are used to inform the agents further. We now go on to talk more about two specific mechanisms in our algorithm: Thompson Sampling and the log-scoring cost-based automated market maker (LSMR AMM).

2.1 Thompson Sampling and a Bayesian Framework

Solving the problem of maximizing rewards is related to a classic exploration-exploitation trade-off. Intuitively, on one hand we want to continue pulling arms that we know produce rewards. However, in order to discover which arms are good or bad, we need to explore the action space of arms.

One guiding way to do this lies in Thompson Sampling, a widely studied algorithm for BBPs (3). We assume Bayesian priors on the Bernoulli means θ_i, taking advantage of the Beta-Binomial conjugacy.

Accordingly, we have

$$\begin{aligned} R(A_i) &\sim \text{Bern}(\theta_i) \\ \theta_i &\sim \text{Beta}(\alpha, \beta) \\ \theta_i \mid R(A_i) &\sim \text{Beta}(\alpha + I_{R(A_i)=1}, \beta + I_{R(A_i)=0}) \end{aligned}$$

where $I_{R(A_i)=1}$ is the the indicator variable for if our Bernoulli trial's reward $R(A_i)$ resulted in a success.

The Thompson Sampling algorithm initially assumes arm A_i to have prior $(1, 1)$ on θ_i, which by representation seems reasonable given

$$\text{Beta}(1, 1) \sim \text{Unif}(0, 1)$$

Intuitively, we first sample from our initial prior distributions, picking the arm with the largest outcome, or sampled Bernoulli mean. Pulling this arm produces some reward 0 or 1, and this in turn updates our prior. We then update the corresponding Beta parameter, which over time narrows the distribution surrounding our arms. Accordingly, we can think of the Beta distributions as shifting along probability support $[0, 1]$, continuously updating to represent our updating beliefs that some event is true.

Algorithm 1: Thompson Sampling for Bernoulli Bandits

```
For each arm $A_i \in A$, set $S_i = 0$, $F_i = 0$.
foreach $t = 1, 2, \dots$ do
    For each $A_i \in A$, sample $\hat{\theta}_i(t)$ from the Beta$(S_i + 1, F_i + 1)$ distribution.
    Pull arm $A_i(t) := \arg\max_i \hat{\theta}_i(t)$ and observe reward $R(A_i(t))$.
    if $R(A_i(t)) = 1$ then
        Set $S_{i(t)} = S_{i(t)} + 1$
    else
        Set $F_{i(t)} = F_{i(t)} + 1$
```

2.2 Prediction Market Mechanisms

After the training phase, we want the agents to continue to to learn through participating in a prediction market. The online learning's motivations are two-fold. For one, information aggregation across all agents may lead to a more accurate predictions overall. This argument stems from standard literature and previous supporting work describing the benefits of ensembles over individual models, such that we can imagine tapping into whatever the equilibrium beliefs are of the market for a prediction that captures the beliefs of everyone in the market. However, we also consider the individual learning aspects of a market, where this information elicitation also informs an agent on how well it performs, causing it to update its beliefs accordingly.

In order to carry this out, we design a multi-contract market place implemented by a cost-based automated market maker market with logarithmic scoring rule. We represent the underlying beliefs of an arm resulting in a reward through contracts, such that the number of contracts is the number of bandit arms. Agents sample their posterior distributions given the data they have been exposed to, which becomes their new priors, and these reported beliefs are then submitted as bids reflecting their beliefs to the market. We next go into the reasoning behind our market design mechanism of choice.

2.2.1 Automated Market Maker

While prediction markets in real life and with human participants may more often take the form of a continuous double auction, or CDA, we propose the use of an alternative categorized by several properties and benefits.

Most fundamentally, our mechanism takes the form of an *automated market maker* (AMM), a design that allows the buying or selling of any quantity of contract also not limited to integer units(1). Aside from this perceived flexibility and enhanced liquidity, an automated market maker is crucial to elicit beliefs in a market which may particularly one-sided. If no one is willing to take the other side of a trade in a CDA, no trades occur and we cannot form a deep opinion on what the equilibrium price in a prediction market should be. Especially in the context of bandit-solving algorithms that may very much have some correlation in their arm reward priors we would like to elicit info even if it is one-sided, and AMMs allow us to do so. Briefly, AMMs operate by maintaining a ledger of quantities of competing contracts, which together can be used to calculate the instantaneous contract price or market belief for a certain event. An example of such contract pairs includes paying $1 if it rains and does not rain tomorrow respectively. Buying more of one contract increases it's price while lowering the price of its competing partner, and barring financial constraints agents may freely buy or sell quantities until a point such that the instantaneous market price reflects their internal belief. When agents can do so, we say that the market is *myopic*, which is one condition that in conjunction with prices adding up to 1 and prices increasing with increasing buys allows us to interpret instantaneous prices as an aggregate belief on uncertain future events.

We additionally desire differentiability and monotonicity, two properties satisfied through a *cost-based market maker* (1), and through the following market-scoring rule, we obtain several other desirable marketplace properties.

2.2.2 Logarithmic Market Scoring Rule

Another condition in our prediction market design is the use of a cost-based logarithmic market scoring rule or LMSR. A market scoring rule uses a *strictly proper scoring rule* to elicit belief reports from multiple agents. We desire the strictly proper such that agents uniquely maximize their expected payment in a market by reporting their true belief. In addition, through a conjuction of cost-based and log-scoring, we are able to obtain conditions such as strictly positive prices for all contracts, liquidity, and no-round-trip arbitrage, all important for the health and stability of a market with live participants (1), and conditions that motivate our algorithm design. These properties follow from the given definition of the instantaneous price on contract k in the LMSR market maker:

$$p_k = \frac{e^{x_k}/\beta}{\sum_{j=0}^{m-1} e^{x_j/\beta}}$$

where x_i represents the quantity for contract i, and β is some constant > 0 allowing for liquidity control. In our AMM, we initialize each contract to price 0.5. If given the agent's Thompson Sampling priors, the agent believes the current price is too low, following myopic incentives it will bid quantity x until the instantaneous prices matches its internal belief, such that

$$x = \begin{cases} \beta \ln\left(\frac{p}{1-p}\right) + a - b & \text{if the current price is too low} \\ \\ \beta \ln\left(\frac{1-p}{p}\right) + b - a & \text{if the current price is too high} \end{cases}$$

where β is the LSMR AMM liquidity constant, p is the updated belief, and b and a are the quantities of bids for the outcome to happen and not happen respectively before adding x (see appendix for derivations).

2.3 Market-based Updating

We use the quantities of the LMSR to update the individual parameters of the arms and also update the individual distributions of the agents. Taking advantage of the information aggregation mechanism in the prediction market, we aim to use the aggregated bids to make a more informed decision of which arm to pull. For slight justification, given the Beta priors of each individual agent, each bid is expressed by taking the average of sampling from the distributions repeatedly. By the Central Limit Theorem, these beliefs then take the form of a Normal distribution. Additionally, because each agent bids a certain quantity of its belief to the market, and affine transformations of Normal random variables are still normal, we consider a sample of bids for each contract to represent the information aggregated by the market. Defining market beliefs to be Normal distributions parameterized by mean $\bar{\mu}_j$ and variance $\bar{\sigma}_j^2$ for each contract j, we then set these parameters to estimators akin to the MLEs explaining the bids, given by

$$\bar{\mu}_j = \frac{\sum_{i=0}^{m-1} x_i p_i}{\sum_{i=0}^{m-1} x_i} \text{ and } \bar{\sigma}^2 = \frac{\sum_{i=0}^{m-1} x_i (p_i - \bar{\mu}_i)^2}{\sum_{i=0}^{m-1} x_i}$$

where x_i is the quantity associated with a bid from agent i to bring it to probability belief or price p_i (derivation in the appendix). We then follow the Thompson Sampling algorithm across the market's belief distributions to pick an arm, and all agents participating in the market update their Beta beliefs accordingly considering subsequent successes and failures. Through the aggregation of information and updating in a prediction market, we hope that out algorithm is able to go beyond the usual capabilities of Thompson Sampling, which typically converges on collecting a large amount of information for a single arm, and instead build up a more general understanding of a complete joint probability distribution of rewards across all arms. In addition, we hope to incorporate an element of “learning" from other participants' priors, without actually having access to the data that they learned their priors from (Algorithm 2).

3 Preliminary Results

We now detail the findings of our experiments, which consisted of implementing and testing Thompson Sampling, as well as building a larger prediction market that could take in agent bids to elucidate the generation of probability distributions.

Because multi-armed bandits belong to a class of interactive learning problems that deal with taking some action and actually receiving some corresponding reward, we noted a relative lack of strictly MAB problem-centric data sets online. In order to elucidate our model, we thus simulated datasets according to a bandit structure. Specifying a 5-dimensional probability vector to denote our underlying probability distribution, we sampled from Bernoulli distributions for each parameter, such that the resulting array represented the corresponding outcomes of pulling a single arm. Without knowing the parameters, the agent picks one arm, and receives the reward associated with that arm only.

3.1 Thompson Sampling

To initialize the belief priors of agents, we trained agents on their individual bandits through the Thompson Sampling algorithm. 3 agents were initialized with Beta priors Beta$(1, 1) \sim$ Unif$(0, 1)$ to reflect a uniform belief distribution for all arms, updating over time. They each trained on disjoint multinomial data sets with support $\{0, 1\}$ parameterized by $\Theta = (0.45, 0.45, 0.45, 0.35, 0.55)$. With just this initial data exposure, empirical rewards did not obviously seem to get more frequent. However more rewards over time were observed with the same Thompson Sampling algorithm applied to the overall prediction market beliefs (Figure 1).

Additionally, we observe Beta belief updates beginning to center around the specified parameters. As noted in Figure 2, even with the same underlying distribution and following the same updating algorithm, the individual agents build up different posteriors from the partitioned data sets exposed to them through their corresponding bandits.

We also took a look at the cumulative reward and regret of the agents trained on their individual bandits, as seen in Figure 3. While cumulative reward is an intuitive metric to measure how well an agent is performing, regret may be more informative. For each round, we define regret to be the difference between the agent choice's payoff and the max payoff that could have occurred in that round from picking a not necessarily different arm. The cumulative regret is thus this difference summed over time.

Algorithm 2: Prediction Market Update for Bernoulli Bandits

```
Set data set $\mathcal{F}^*$ for prediction market
foreach agent $i$ do
    Initialize with Beta$(1, 1)$ prior
    Set data set $\mathcal{F}_i \in \mathcal{F} \setminus \mathcal{F}^*$
    Assign $\mathcal{F}_i$ to bandit $B_i$ with arms $A_{ij} \in A_i$
    Update $i$'s priors through Thompson Sampling on $B_i$
Initialize prediction market with data set $\mathcal{F}^*$
foreach $ix \in \{1, 2, \dots, n\}, n = |\mathcal{F}^*|$ do
    For each arm $A_i \in A$, initialize contract ledger $C_i \in C$
    foreach $i$ do
        Set $p_j$ = average of Thompson-sampled posterior.
        Update market price to $p_j$ by submitting quantity $x_j$ following LSMR
        Append $x_j$ to ledger $C_i$
    foreach $C_i$ do
        Set $p_j$ = average of Thompson-sampled posterior.
        Update market price to $p_j$ by submitting quantity $x_j$ following LSMR
        Append tuples $(x_j, p_j)$ to ledger $C_i$
    foreach $C_i$ do
        Set $\bar{\mu}_i$ = mean parameter from MLE of $(x_j, p_j), \dots \in C_i$
        Set $\bar{\sigma}_i^2$ = variance parameter from MLE of $(x_j, p_j), \dots \in C_i$
        Set $\hat{S}(C_i)$ = sample from normal $\mathcal{N}(\bar{\mu}_i, \bar{\sigma}_i^2)$
    Select $A_i$ corresponding to $C_i = \arg\max_j \hat{S}(C_j)$
    Reset all $C_i \in C$
    foreach $i$ do
        Update $i$'s parameters given $R(A_i)$ through Thompson Sampling
```

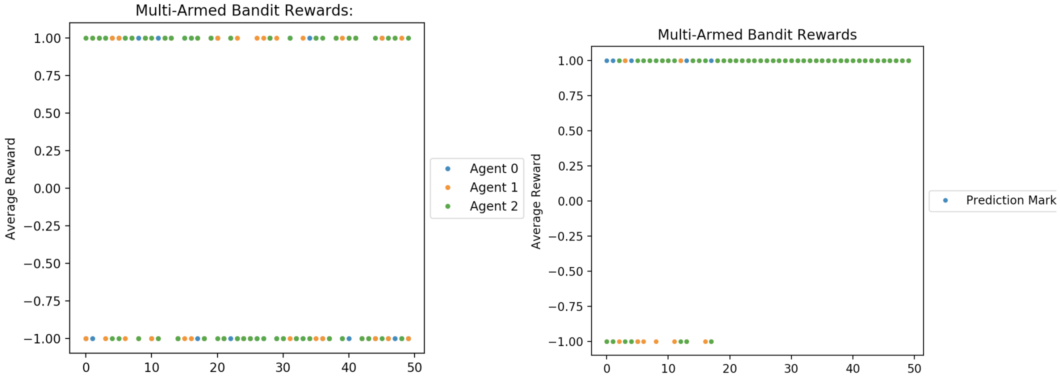

Figure 1: **Thompson Sampling Average Reward over Time**. Agents were initialized with their individual partitioned bandits, picking rewards according to the Thompson Sampling algorithm. For illustrative purposes, picking an arm without a reward inflicted a negative 1 penalty on the agent.

3.2 Market Updates

We next looked at the effects of our prediction market on the effectiveness at building a generative model and identifying the likely parameters behind the overall model. Witholding one of the training folds to help run the prediction market, agents exhibited a tendency towards identifying the overall distribution parameters without knowledge of the underlying multivariate Bernoulli used to generate the data, and only with direct exposure to their own Bandit.

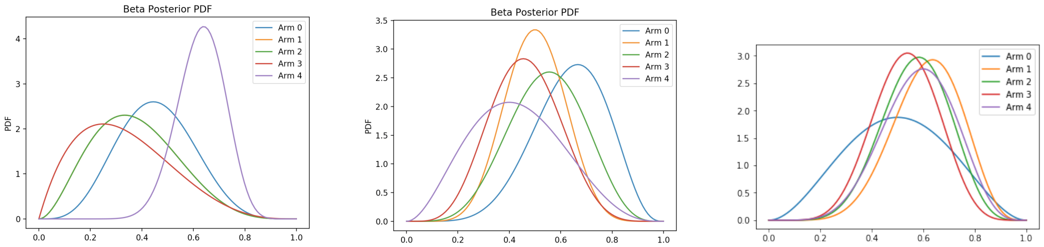

Figure 2: **Beta Posterior Updating through Thompson Sampling**. Agent conjugate priors distributed Beta(α, β) were updated over time given rewards exposed through Thompson Sampling. Overall distributions were initialized with 1000 records, and agents trained on 250 record data sets. A fourth block was saved for prediction market updating.

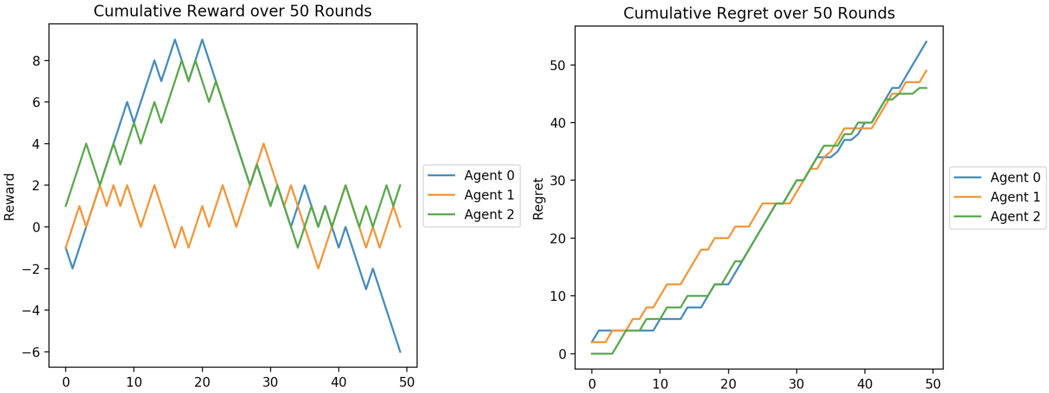

Figure 3: **Cumulative Reward and Regret over Prior Training**. Reward outputs in $\{-1.1\}$ allow for convenient interpretation of how well an agent does over time, with negative or break-even outcomes around 0 denoting a poorly-trained “weak" learner.

To compare the efficacy of the prediction market, posterior beliefs from the prediction market were compared against the distributions of agents with the same learned posteriors as those participating in the market, but instead learning directly from the withheld data set by using Thompson Sampling on the data itself.

As seen in Figure 4, the underlying agent beliefs for each arm begin to agree on the arm that pays out the highest reward. This is desirable behavior when trying to exploit the highest paying arm in a traditional bandits setting, but through the biased exploitation of better-reward arms we get uneven information reveal on the other arms. Accordingly, traditional Thompson Sampling does not seem to give us the best functionality for estimating the entire joint distribution.

These resulting distributions are in contrast to those observed in Figure 5. Here we plot the distributions of the agents, still following their Beta-Bernoulli conjugate updating schemes, but now also updating based on feedback from the market. We see that the distributions seem to be much more centered around the original θ's describing the simulated data set.

3.3 Generative Model Analysis

Given these updated parameters, we finally consider the efficacy of learning generative models through prediction markets. One main motivation comes from re-examining the scarce data argument again. In situations where new agents are initiated without any priors and we do not want to train them through making real decisions, we might imagine wanting a set up where new agents could train on simulated data emitted from older agents. Of course this approach would only work if the older agents held beliefs that were representative of the larger population distribution. Accordingly, we sampled data-generating parameters from both learned agents participating in the prediction market

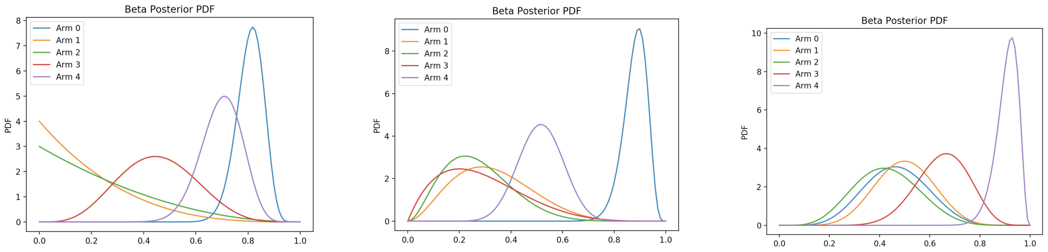

Figure 4: **Beta Posteriors through Thompson Sampling Directly**. Agents were updated over time given rewards exposed through Thompson Sampling, just like in the prior-generation case, but instead using the withheld prediction market data set.

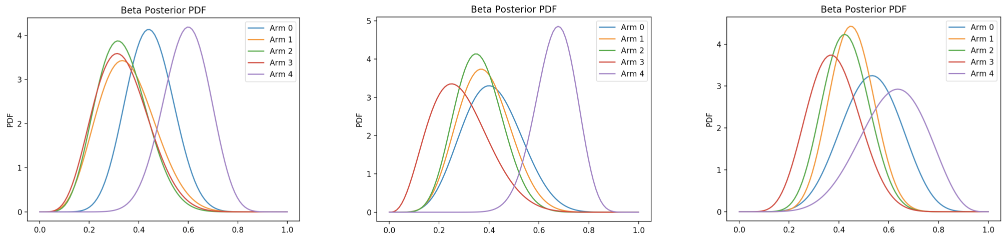

Figure 5: **Beta Posteriors through LMSR Thompson Sampling**. Agents were updated over time given rewards exposed through Thompson Sampling, just like in the prior-generation case, but instead using the withheld prediction market data set.

and learning through Thompson Sampling only. While not actually done, strongly motivated future work would involve generating Bernoulli data sets given these parameters, and trained new agents on the simulated data before letting them act on another withheld data set generated from the original Bernoulli priors. As seen in Figure 6, while both agents display beliefs with relatively narrow tails, prediction market-participating agents seem to exhibit distributions centered closer to the original parameters.

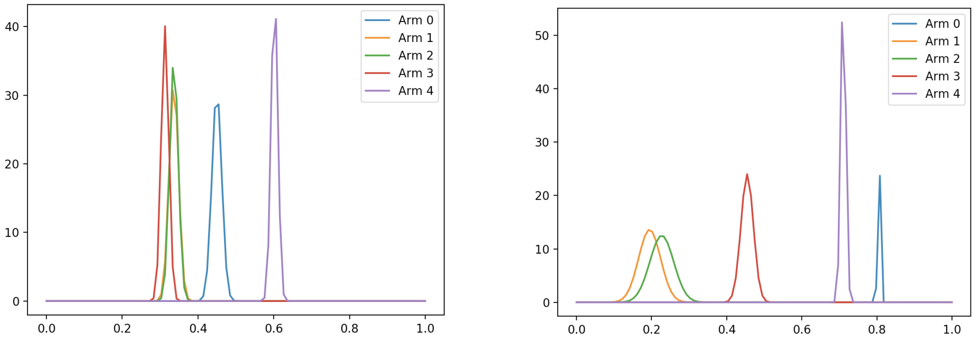

Figure 6: **Sampled Parameters for Bernoulli Data Generation**. Prediction market and Thompson Sampling-only agents (left and right respectively) parameters sampled from their belief distributions. Distributions were constructed by sampling repeatedly and calculating the Normal distributions.

4 Discussion

Through our exploration of prediction markets applied to the multi-armed bandits problem, we found several interesting empirical results motivated by connections between heterogenous market participants and machine learning models learning on disparate data partitions. From an initial motivation of trying to learn more beyond the given data that's immediately available, we explored connections between information aggregation, ensemble learning, and new mechanisms that allow for generative model construction and joint distribution inference solely from Thompson Sampling. While our original context was the multi-armed bandit setting, we also acknowledge that there are multiple related problem classes and applications for future work.

Perhaps a natural follow-up is the realm of *contextual* bandits, where in addition to an action each decision must also be accompanied with concern for the surrounding context. Recommendation settings have traditionally been discussed through the lens of collaborative filtering and similarity models, but picking a good recommendation can also be viewed as a bandit problem. Considering the Netflix Challenge, picking an arm is akin to displaying a movie recommendation, and we count a success if the movie is considered a good recommendation (maybe rating 4 or 5 on a 5-point scale). However, two main issues stand out obviously between our work and a solution. For one, movie recommendations span much more than 5 arms, and with new movies being released every year and others retiring from listing we need to consider how we can model a dynamic arm space. Additionally, while in the space of all movies, certain movies might popularly be regarded as good movies and others bad, this is not interesting in terms of a personalization aspect. Additionally, for more complicated settings such as going back to our opening example and recommending a treatment for patients, we can imagine such personalization being more important. Accordingly, the distribution of recommendations is not just defined by the underlying parameters of possible arms, but also the surrounding context of that arm, and our learning function must increase in complexity to accommodate for this. Learning contextual bandits are thus an exciting challenge to tackle and standing motivation for combining market design and machine learning.

5 Appendix

5.1 Calculating Quantity to Bid in LMSR AMM

From the text, we note that the instantaneous price on contract k in the LMSR market maker is given by

$$p_k = \frac{\exp(x_k/\beta)}{\sum_{i=0}^{m-1} \exp(x_j/\beta)}$$

which for a two outcome contract reduces to

$$p_k = \frac{\exp(x_0/\beta)}{\exp(x_0/\beta) + \exp(x_1/\beta)}$$

where x_i denotes the standing quantity of contracts in favor of outcome o_i. Because agents enter the market expressing their own beliefs on the probability of a contract's success, for the two-outcome case they should bid quantities x such that

$$p_k = \frac{\exp((x+b)/\beta)}{\exp((x+b)/\beta) + \exp(a/\beta)} \text{ or } p_k = \frac{\exp(b/\beta)}{\exp(b/\beta) + \exp((x+a)/\beta)}$$

where b and a denote the standing quantities in favor of outcomes o_0 and o_1 respectively. In the case that an agent enters the market and the standing price is too low (agent thinks o_0 is more likely), they should accordingly bid

$$\begin{aligned}
x \text{ s.t. } p &= \frac{\exp((x+b)/\beta)}{\exp((x+b)/\beta) + \exp(a/\beta)} \\
&\Rightarrow e^{(x+b)/\beta} = pe^{(x+b)/\beta} + pe^{a/\beta} \\
&\Rightarrow -pe^{a/\beta} = (p-1)e^{(x+b)/\beta} \\
&\Rightarrow e^{(x+b)/\beta} = \left(\frac{p}{1-p}\right) e^{a/\beta} \\
&\Rightarrow (x+b)/\beta = \ln\left(\frac{p}{1-p} e^{a/\beta}\right) \\
&\Rightarrow \boxed{x = \beta \ln\left(\frac{p}{1-p}\right) + a - b}
\end{aligned}$$

Similarly, if we think that the price on outcome o_1 is too low, we bid quantity x such that

$$\begin{aligned}
p &= \frac{\exp(b/\beta)}{\exp(b/\beta) + \exp((x+a)/\beta)} \\
&\Rightarrow e^{b/\beta} + e^{(x+a)/\beta} = \frac{1}{p} e^{b/\beta} \\
&\Rightarrow e^{(x+a)/\beta} = \left(\frac{1-p}{p}\right) e^{b/\beta} \\
&\Rightarrow \frac{x+a}{\beta} = \ln\left(\frac{1-p}{p}\right) + \frac{b}{\beta} \\
&\Rightarrow \boxed{x = \beta \ln\left(\frac{1-p}{p}\right) + b - a}
\end{aligned}$$

5.2 Modified Maximum Likelihood Estimator

In our setup, we describe a setting where ledgers keeping track of the number of bids for each contract can then be used to infer some overall market belief parameter regarding the contract. Under the typical Normal MLE derivation, we wish to infer the underlying mean and variance parameters μ, σ^2 given some data set with integer size.

Under the LMSR cost-based AMM mechanism the agents bid some quantity that we do not assume to be integer quantities, we cannot use the exact same calculation. Instead, we observe a setting where we might want to observe underlying θ from a set of weighted values $\{x_1p_1, x_2p_2, \ldots x_np_n\}$ for quantities x_i and beliefs p_i for agents $i \in \{1, 2, \ldots n\}$. However, based on the concept of sufficient statistics, or statistics that essentially capture all information about some underlying parameter, we claim that we can still infer our desired parameters from our non-integer bids.

To do so, note that under the Neyman-Fisher Factorization Theorem, a statistic T is sufficient for θ if for data vector $\mathbf{x}$, $f(\mathbf{x}; \theta) = f(t, \theta)h(\mathbf{x})$ where the function $g(t, \theta)$ only depends on $t = t(\mathbf{x})$ and θ while $h(\mathbf{x})$ does not depend on θ.

For the Normal distribution then, we have for observed data $x_1, x_2, \ldots x_n$ which we assume to belong to some distribution $\mathcal{N}(\mu, \sigma^2)$:

$$\begin{aligned}
f(x_1, \ldots x_n) &= \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{-\frac{(x_i - \mu)^2}{2\sigma^2}\right\} \\
&= \frac{1}{\sqrt{2\pi\sigma^2}^n} \times \exp\left\{\frac{-1}{2\sigma^2}\left(-n\mu^2 + 2\mu\sum_{i=1}^{n} x_i\right)\right\} \times \exp\left\{\frac{-1}{2\sigma^2}\sum_{i=1}^{n} X_i^2\right\}
\end{aligned}$$

where we note that

$$h(x_1, \ldots, x_n) = \exp\left\{ \frac{-1}{2\sigma^2} \sum_{i=1}^{n} X_i^2 \right\}$$

and

$$f(t, \theta) = \frac{1}{\sqrt{2\pi\sigma^2}^n} \times \exp\left\{ \frac{-1}{2\sigma^2} \left(-n\mu^2 + 2\mu \sum_{i=1}^{n} x_i \right) \right\}$$

such that θ is not in h and $T = \sum_{i=1}^{n} X_i$ is a sufficient statistic. So we are interested in maintaining the sum of all values, which we contain through the linear combination of quantities and bid values for each contract. Accordingly, when finding the MLE, we replace $\sum_{i=1}^{n} p_i$ with $\sum_{i=1}^{n} x_i p_i$ and n with $\sum_{i=1}^{n} x_i$. Finding the actual MLE then (under the typical case), we now solve for the $\bar{\mu}$ and $\bar{\sigma}^2$ to maximize

$$L(\bar{\mu}, \bar{\sigma}^2) = f(p_1, \ldots, p_n \; \bar{\mu}, \bar{\sigma}^2) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\bar{\sigma}^2}} \exp\left\{ -\frac{(p_i - \bar{\mu})^2}{2\bar{\sigma}^2} \right\}$$

which is the same as solving for

$$\arg\max_{\bar{\mu}, \bar{\sigma}^2} \ell = \arg\max_{\bar{\mu}, \bar{\sigma}^2} \ln \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\bar{\sigma}^2}} \exp\left\{ -\frac{(p_i - \bar{\mu})^2}{2\bar{\sigma}^2} \right\}$$

under monotonicity of a natural log transform and where $\ell = \log L(\bar{\mu}, \bar{\sigma}^2)$. Simplifying the RHS then gives

$$\ell = -\frac{n}{2} \ln(\bar{\sigma}^2) - \frac{n}{2} \ln(2\pi) - \frac{1}{2\bar{\sigma}^2} \sum_{i=1}^{n} (p_i - \bar{\mu})^2$$

Now taking the partial derivative with respect to $\bar{\mu}$ and setting the result to 0, we get

$$\frac{\partial \ell}{\partial \bar{\mu}} = -\frac{1}{2\bar{\sigma}^2} \sum_{i=1}^{n} -2(p_i - \bar{\mu}) = 0$$

where solving for $\bar{\mu}$ gives us $\bar{\mu} = \frac{1}{n} \sum_{i=1}^{n} p_i$. We next take the partial derivative of ℓ with respect to $\bar{\sigma}^2$ and solve for $\bar{\sigma}^2$ such that the derivative is equal to 0, getting

$$\begin{aligned} \frac{\partial \ell}{\partial \bar{\sigma}^2} &= -\frac{n}{2\bar{\sigma}^2} + \frac{1}{2\bar{\sigma}^4} \sum_{i=1}^{n} (p_i - \bar{\mu})^2 = 0 \\ \Rightarrow \; \frac{\partial \ell}{\partial \bar{\sigma}^2} &= -n\bar{\sigma}^2 + \sum_{i=1}^{n} (p_i - \bar{\mu})^2 = 0 \qquad \text{(multiply by } 2\bar{\sigma}^4\text{)} \\ \Rightarrow \; \bar{\sigma}^2 &= \frac{1}{n} \sum_{i=1}^{n} (p_i - \bar{\mu})^2 \end{aligned}$$

Making our substitutions then, we end up with

$$\bar{\mu} = \frac{\sum_{i=1}^{n} x_i p_i}{\sum_{i=1}^{n} x_i} \text{ and } \bar{\sigma}^2 = \frac{\sum_{i=1}^{n} x_i (p_i - \bar{\mu})^2}{\sum_{i=1}^{n} x_i}$$

5.3 Code

Code including training environment, agents, bandits, data simulation, and prediction market classes was also written for CS 136: Economics and Computation, and found at .

6 Acknowledgments

Special thanks to Scott Kominers and David Parkes for the awesome semesters in Ec 2099 and CS 136; while this shout-out is probably only a drop in the sea of similar-sounding praise they get for being two of the coolest professors around, it really has been pleasure to sit in their lectures. Additional thanks goes to Serena Yeung, another expert figure, who deserves big thanks for agreeing to listen to us ramble about our thoughts while we really needed to just be sitting down and writing out the math behind them.

References

[1] David Parkes and Sven Seuken. *Economics and Computation*. Cambridge University Press 2018.

[2] Lillian Weng. *The Multi-Armed Bandit Problem and Its Solutions*. Github Pages. January 23, 2018.

[3] Daniel J. Russo, Benjamin Van Roy, Abbas Kazerouni, Ian Osband, and Zheng Wen. *A Tutorial on Thompson Sampling*. Stanford 2018.

[4] Pavel Surmenok. *Contextual Bandits and Reinforcement Learning*. Medium 2017.