Multi armed Bandits

Chuck Chan

April 3, 2021

Problem

- Conditions
  - Fixed set of limited resources and time
  - Resources must be allocated between competing choices to maximize expected gains
  - Each choice has properties that are partially known at the time of allocation and becomes better understood as time passes or by allocating resources
  - Assumes the distribution for rewards from choices are fixed but unknown
- Classic Example
  - Gambler deciding which slot machine to play and how many times to play each, which order to play them, and if they should continue or switch machines.

Explore & Exploit tradeoff

- A tradeoff is created between time spent on Exploring and Exploiting the choices
  - Exploration is the time spent gathering information on the possible choices.
  - Exploitation is spending the time on the choice that is expected to maximizes the gains.
- Regret (ρ) is the difference between gains from the strategy taken and maximum gains given complete information. It is used to evaluate how well a strategy performs.
- Some algorithms can be stuck on suboptimal solutions like local means

Multi Armed bandit Applications

- Clinical Trials
  - Adaptive allocation strategy to improve data collection by allocate more samples for exploring promising treatments
- Portfolio management
  - Making portfolio choices by exploiting correlations among arms to balance risk and amount of passive and active investments
- Dynamic pricing
  - Real-time pricing for product balancing demand with maximizing profits
- Recommender Systems
  - Recommend relevant items to users based on contextual information
- Social Media Marketing
  - Maximize user awareness of a product through social networks by selecting seed users for exposure
- Information Retrieval
  - Determines selection process for information returned by search
- Anomaly detection
  - Identify deviating nodes that are used for predicting fraudulent transactions that are given to a human investigator

https://arxiv.org/pdf/1904.10040.pdf

Solutions

- ε Greedy
  - Assigns probability to balance exploration / exploitation
- Softmax / Botzmann Exploration
  - Algorithm converges on an efficient arm
- Pursuit Algorithm
  - Modifies probability of choosing each arm based on rewards
- Reinforcement Comparison
  - Calculates a preference for each arm and compares an mean rewards
- Upper Confidence bound family of algorithms
  - Construct and updates a mean and confidence interval for the reward of each arm
- Gittens Index
  - Maintains a value function for each arm
- Thompson Sampling
  - Maintains a mean and standard deviation for each arm and selects based on prior probabilities

ε Greedy Strategies

- Strategy to balance explore and exploit options
  - ε is the probability of choosing explore and 1- ε is exploit
  - Each round an arm is selected based on a probability is generated randomly of explore and exploit
  - During exploration each non-best arms are all considered equivalently
- Represented as:

$$p_i(t+1) = \begin{cases} \boxed{1 - \varepsilon + \varepsilon/k} \\ \boxed{\varepsilon/k} \end{cases}$$

Probability the best arm gets played Corresponding to $i = \text{argmax}_{j=1,\ldots,K}\, \widehat{\mu_j}(t)$

Probability a random arm getting selected

Probability of choosing arm i at time t+1

k is the number of different arms

Softmax / Boltzmann Exploration

- Select arms based on rewards from a probability from the Boltzmann distribution
- Arms with larger empirical means have a higher chance of being picked
- Hyperparameter τ controls the randomness of the choice
  - τ = 0 means the algorithm becomes greedy picking the largest empirical mean
  - When τ is large, arms are selected uniformly at random
- Represented as:

average reward for arm i

$$p_i(t+1) = \frac{e^{\widehat{\mu_i}(t)/\tau}}{\sum_{j=1}^{k} e^{\widehat{\mu_j}(t)/\tau}}, i = 1, \dots, n$$

average reward for all arms

Probability of choosing arm i at time t+1

Pursuit Algorithm

- Starts with uniform probabilities assigned to each arm $p_i(0) = 1/k$ and a learning rate $\beta \epsilon\ (0,1)$
- Probabilities are updated after each turn t
- Represented as:

Update if $i = \underset{j}{\operatorname{argmax}}\ \widehat{\mu_j}(t)$

$$p_i(t+1) = \begin{cases} p_i(t) + \beta(1 - p_i(t)) \\ p_i(t) + \beta(0 - p_i(t)) \end{cases}$$

Update of arm i otherwise

Learning rate between (0,1)

Reinforcement Comparison

- Maintain an distribution over all actions that is updated
- Maintain an average expected reward $\bar{r}(t)$
- Probability of selecting an arm is computed by comparing empirical mean with the average expected rewards
  - Probability is increased if it is above the average expected rewards
  - Decreased if equal or lower
- Probability of selecting an arm i :

Preference for each arm i

$$p_i(t) = \frac{e^{\pi_i(t)}}{\sum_{j=1}^{k} e^{\pi_j(t)}}$$

average preference for arm j

- Preference update if arm $j(t)$is played and reward $r(t)$ is received:

$$\pi_{j(t)}(t+1) = \pi_{j(t)}(t) + \beta(r(t) - \bar{r}(t))$$

Average expected rewards

Learning rates between 0,1

- Mean rewards update at every turn:

$$\bar{r}(t+1) = (1-\alpha)\bar{r}(t) + \alpha r(t)$$

Reward at time t

Upper Confidence Bounds algorithm family

- construct a confidence interval of what each arm’s true performance might be
  - Factors the uncertainty caused by variance using empirical means
  - Limited series of pulls tracking times each arm is played
- Algorithm assumes each arm will perform as well as its upper confidence bound selecting the arm with the highest one
- Algorithm variants
  - UCB-1
  - Tuned UCB-1 (includes variance)
  - Bayes UCB (includes prior distribution information)

Upper Confidence Bounds 1

- maintains the number of times each arm is played and empirical means
- Use Hoeffding's inequality to assign upper bound to an arm's mean reward
- Initially each arm is played once
- Followed by a greedy algorithm:

$$j(t) = \underset{i=1,\ldots,k}{\arg\max}\left(\underbrace{\hat{\mu}_i}_{\text{average reward for arm } i} + \overbrace{\sqrt{\frac{2lnt}{\underbrace{n_i}_{\text{Number of times arm } i \text{ is played}}}}}^{\text{confidence interval for the average reward}}\right)$$

Upper Confidence Bounds 1

- Smaller n_i the larger the Confidence Intervals and less accurate mean rewards, Larger results in smaller confidence and more accurate mean rewards
- As t increases the algorithm converges to the optimal action
- Regret calculation

$$8\sum_{i:\mu_i<\mu^*}\frac{\ln(t)}{\mu^*-\mu_i}+\left(1+\frac{\pi^2}{3}\right)\sum_{i=1}^{k}\mu^*-\mu_i$$

- Regret is O(log(n)) bounded growth, optimizing regret to a multiplicative constant

Tuned UCB1 (Audibert, et al 2009)

- Performs better than UCB
- Includes variance of each arm
- Equation to pick arms at turn t:

Highest variance for reward distribution of Bernoulli random variable (p=0.5)^2

Variance calculated by sum of squares of reward

$$j(t) = \underset{i=1,\ldots,k}{\arg\max}\left(\hat{\mu}_i + \sqrt{\frac{lnt}{n_i} min\left(\frac{1}{4}, \hat{\sigma}_i(t) + \frac{2lnt}{n_i}\right)}\right)$$

average reward for arm *i*

Number of times arm *i* is played

Bayesian UCB

- Uses Priors about distributions in determining confidence intervals
- Assume rewards normally distributed
- Algorithm:

Hyperparameter determining the size of Confidence interval

Standard deviation of rewards of arm i

$$j(t) = \arg\max_{i=1,\dots,k} \left(\hat{\mu}_i + \frac{c\ \sigma_i(t)}{\sqrt{n_i}} \right)$$

average reward for arm *i*

Number of times arm *i* is played

Thompson Sampling

- also known as posterior sampling & probability matching
- Aims to quickly identify arm with highest mean, by starting with a posterior belief for the mean and adjusting the mean for the arm played
- Algorithm converges to the optimal solution.
- Chooses arm with posterior probability of being the best arm
  - Not choosing the one most likely to be effective
  - Gives benefit of doubt to less explored choices
- Accounts for uncertainty about means, higher uncertainty ensured a less explored arm is pulled
- The variance in the posterior enables exploration

Thompson sampling for Bernoulli rewards

- Sampling distribution with Bernoulli rewards
- Start with Beta priors using *Beta(1,1)* for each arm
- *Beta(α,β)* priors are updated at round t where:
  - Successes updates *Beta(α+1,β)* if 1 is observed
  - Failures updates *Beta(α,β+1)* if 0 is observed
- Arm i is selected that maximizes rewards calculated by $\hat{\theta}_k = \frac{\alpha_k}{\alpha_k+\beta_k}$
- Beta posteriors are updated.

Thompson Sampling with Gaussian Priors

- Rewards for arm i is N(μ_i, 1)
- Starting priors set N(0,1)

Empirical mean

- Sample $\tilde{\mu}_{i,t}$ from posterior mean for $N\left(\hat{\mu}_{i,t}, \frac{1}{n_{i,t}+1}\right)$ arm i
- Arm i is selected that maximizes $\tilde{\mu}_{i,t}$
- Update the empirical mean $\hat{\mu}_{i,t}$ for i $_t$

Number of observations

Gittins index

- Measures reward that can be achieved in a stochastic process using dynamic allocation indexes
- Used for Bayesian MAB problems where
  - Each arm has a reward distribution that has an known prior
  - The arms are independent
  - The objective is to maximize the expected discount reward
- Prior probabilities are updated each time a decision is made with posterior probability
- An single arm is represented as a Markov chain and reward function
- Policy is a function that determines the decision of which arm to chose next.

Gittens Index Calculation and Policy

- A Gittens Index (GI) is calculated for each arm and the reward is allocated as stochastic process of success and failure with an unknown probability of success
- A policy is implemented and depends on the relative accuracy of the GI limited by:
  - decisions involving arms with similar GI values being rare
  - Cost of a suboptimal action is small when the GI is close to the optimal arm, giving a boundary for the loss rewards of the policy in terms of GI where the GI accuracy limits the losses and rewards.
- Each time an arm is selected the GI for that arm is updated.
- Some policies may reduce / retire arms

https://arxiv.org/pdf/1909.05075.pdf

Calibration method for GI calculation

- Value function for single arm(risk arm):

$$V(\Sigma, n, \gamma, \lambda) = \max\left\{\underbrace{\frac{\Sigma}{n} - \lambda + \gamma E_Y[V(\Sigma + Y, n+1, \gamma, \lambda)]}_{\text{Value of the risk arm}} ; \underbrace{0}_{\text{Value of the safe arm reduced to 0}}\right\}$$

- The fix reward arm λ minimizes the value function to close to zero and is found by recursion of a calibration algorithm given some initial bounds

Variables

λ an arm of known fixed reward

γ discount factor

Y random variable of the risk arm state with state (Σ, n)

Σ Bayes sum of rewards of the risk arm

n Bayesian number of observations of the risk arm

Bernoulli MAB GI calculation

- Reward of one or zero
- Value function for discrete single arm of a disc

$$V(\Sigma, n, \gamma, \lambda) = \max\left\{\frac{\Sigma}{n} - \lambda + \gamma\left[\frac{\Sigma}{n} V(\Sigma + Y, n+1, \gamma, \lambda) + \left(1 - \frac{\Sigma}{n}\right) V(\Sigma, n, \gamma, \lambda)\right]; 0\right\}$$

Variables

λ an arm of known fixed reward
γ discount factor
Y random variable of the risk arm state with state (Σ, n)
Σ Bayes sum of rewards of the risk arm
n Bayesian number of observations of the risk arm

Algorithms for the multi-armed bandit problem (2000)

- Tested multiple MAB solutions against different bandit characteristics
- Algorithms tested
  - ε Greedy
  - Boltzmann Exploration
  - Upper Confidence Bounds
  - Reinforcement Comparisons
  - Pursuit Algorithms
- Characteristics for MABs
  - Different distributions
  - Number of arms
  - Reward variances
- Tested algorithms in Medical Clinical trial settings

https://arxiv.org/pdf/1402.6028.pdf

Algorithms for the multi-armed bandit problem (2000)

- Simple heuristic algorithms like Boltzmann exploration out performed advance algorithms
  - Both ε Greedy and Boltzmann Exploration had versions that had decreasing hyperparameters ε and τ but were found to have no advantage by Vermorel and Mohri (2005)
- Performance of algorithms affected by number of arms and reward variances
- Algorithms vary depending on Bandit instances
  - UCB family is good for bandits with small number of arms and high reward variance, but degrades as number of arms increase

Adversarial Bandits

- Multi-arm Bandit variant where an adversary determines the outcome of based on the arm that is selected
- Goal is to minimize negative outcomes
- Examples
  - iterated prisoner's dilemma
  - Clinical trials
- Solutions include EXP3 algorithm
  - Weights are adjusted for favorable arms

https://arxiv.org/pdf/1904.07272.pdf

Exponential weight for Exploration & Exploitation (EXP3)

- Weight are adjusted over time to favor good arms and reduced for less promising arms
- Algorithm initialized a vector of weights each at 1
- Setup:

Weight of arm i

Hyperparameter for exploration of arms uniformly at random where 1= pure exploration

$$p_i(t) = (1-\gamma)\frac{w_i(t)}{\sum_{j=1}^{k} w_j(t)} + \frac{\gamma}{k}$$

Sum weight of all arms

k is the number of different arms

- Draw arms i_t randomly according to $p_i, \ldots, p_k$ and observe rewards $x_{i_t} \in [0,1]$

- Define estimated reward $\hat{x}_j(t) = \frac{x_j(t)}{p_j(t)}\ for\ j = i_t, otherwise\ 0\ for\ all\ j$

- Update weights $w_j(t+1) = w_j(t)e^{\gamma \hat{x}_j(t)/k}$

Contextual Bandits

- Multi-armed bandit variant where choosing an arm is dependent on a set of contextual information. The reward is observed for the chosen action
- Goal:
  - Balance explore and exploit behavior
  - Use context as effectively as possible, generalize across context
  - Handle selection bias from skewed data of exploring while exploiting
- Characteristics
  - K arms
  - d-dimensional feature vector $x_{i,t}$ for every arm i, at time t
  - Linear parametric model with parameter θ and expected rewards for arm i, at time $t : x_{i,t}\theta$
  - Optimal arm depends on context $x_t^* = \ arg\max_{x_{i,t}} \ x_{i,t}\theta$

Contextual Bandits

- Determine a policy to choose action based on context
  - Policies decide prior to learning
  - Associating context with the best action is treated as a classification problem
- Examples
  - News / Advertising from a website that is served to users with a set of browsing history
  - Medical doctor assigning a treatment with the most favorable outcome based on symptoms
  - Clinical trial with patient data
  - Recommender systems for media with user preferences

Contextual bandit solutions

- Follow the Leader (FTL)
  - Find the empirically best policy and use it to select the best action
- Thompson Sampling
  - Use least square estimator for rewards, the state transition is learned by comparing against a fixed policy.
  - Transition probabilities for each state and action is stored in a Dirichlet posterior
- EXP4
  - Modified EXP3 algorithm with experts advice influencing probability
- LinUCB
  - Modified UCB algorithm that uses a data matrix of features for context, a linear function with hyperparameters, and a vector corresponding to feedback

https://tor-lattimore.com/downloads/book/book.pdf

More Bandit variations

- Lipschitz Bandit, where an algorithm has information on the similarity between arms.
  - Used for dynamic pricing problems
- Bandits with Knapsack, bandit problems with a supply constraints
  - Used in dynamic pricing problems with limited supply
- Combinatorial bandit, where a set of k arms are selected
  - Used for online advertising and social influence maximation.
- Adversarial Bandits with corruption, where the adversary knows the actor's choice and has the ability to corrupt the reward of the selected arm.

The End