EE675A

Lecture 8: Policy Gradient Algorithms and Contextual Bandits

08/02/2023

Lecturer: Prof. Subrahmanya Swamy Peruru *Scribe: Sai Ashwith Dharoor*

Till the previous lecture, we tried to find the *best arm* by estimating the *mean reward* given by each arm and then choosing the one whose estimate is best. And to strike balance between exploration and exploitation, we have studied various algorithms which calculate these estimates and devise an optimal policy based on these estimates.

All the methods we have studied till now are *'Value Based Methods'* since they work by estimating the values $\bar{\mu}(a_i)$ and choosing $\bar{\mu}^*$ among them. Another way of choosing the best arm, in other words, devising the best *policy* to maximize our reward, is through **Policy Gradient Algorithms**, which don't have to estimate any values to choose the optimal arm.

1 Policy Gradient Algorithm [2]

We define a policy $\mathbf{\Pi}(\mathbf{a};\theta)$ over the set of arms $\mathcal{A}$, which is a function parameterized by the set of parameters θ and differentiable w.r.t θ. For example, if we have two arms a_1 and a_2, then $\mathbf{\Pi}$ can be of form:

$$\mathbb{P}(a_1) = f_1(\theta)$$
$$\mathbb{P}(a_2) = f_2(\theta)$$

And the performance measure for this policy, the *expected reward*, is given by:

$$\eta(\theta) = \sum_{a\in\mathcal{A}} \mathbf{\Pi}(a;\theta)\cdot\mu(a) = \mathbb{P}(a_1)\cdot\mu(a_1) + \mathbb{P}(a_2)\cdot\mu(a_2)$$

And to maximize the expected reward, we maximize $\eta(\theta)$ over θ, which leads us to optimal policy $\mathbf{\Pi}^*$. And we do this maximization using **Stochastic Gradient Ascent**:

$$\theta_{t+1} = \theta_t + \alpha\cdot\widehat{\Delta_\theta\eta(\theta_t)}$$

where $\widehat{\Delta\eta(\theta)}$ is a stochastic estimate whose expectation approximates the gradient of the performance measure with respect to its argument θ and α is the learning rate.

$$\mathbb{E}[\widehat{\Delta_\theta\eta(\theta)}] = \Delta_\theta\eta(\theta)$$

In the below derivation, we get a usable expression for $\widehat{\Delta_\theta \eta(\theta)}$.

$$
\begin{aligned}
& \eta(\theta) = \sum_{a \in \mathcal{A}} \mathbf{\Pi}(a;\theta) \cdot \mu(a) \\
\implies & \frac{d\eta(\theta)}{d\theta_i} = \sum_{a \in \mathcal{A}} \frac{\mathbf{\Pi}(a;\theta)}{d\theta_i} \cdot \mu(a) \\
\implies & \frac{d\eta(\theta)}{d\theta_i} = \sum_{a \in \mathcal{A}} \frac{\mathbf{\Pi}(a;\theta)}{d\theta_i} \cdot \frac{\mathbf{\Pi}(a;\theta)}{\mathbf{\Pi}(a;\theta)} \cdot \mu(a) \\
\implies & \frac{d\eta(\theta)}{d\theta_i} = \sum_{a \in \mathcal{A}} \left(\mu(a) \cdot \frac{d \log(\mathbf{\Pi}(a;\theta))}{d\theta_i} \right) \cdot \mathbf{\Pi}(a;\theta) \quad , \left\{ using \quad \frac{d \log(f(x))}{dx} = \frac{1}{f(x)} \cdot \frac{df(x)}{dx} \right\} \\
\implies & \frac{d\eta(\theta)}{d\theta_i} = \mathop{\mathbb{E}}_{a \sim \Pi} \left[\mu(a) \cdot \frac{d \log(\mathbf{\Pi}(a;\theta))}{d\theta_i} \right] \\
\implies & \frac{d\eta(\theta)}{d\theta_i} = \mathop{\mathbb{E}}_{\Pi} \left[\mathop{\mathbb{E}}_{v} \big[R_t | a_t = a \big] \cdot \frac{d \log(\mathbf{\Pi}(a;\theta))}{d\theta_i} \right] \qquad , \left\{ where \ \ 'v' \ is \ environment \right\} \\
\implies & \frac{d\eta(\theta)}{d\theta_i} = \mathop{\mathbb{E}}_{\Pi, v} \left[R_t \cdot \frac{d \log(\mathbf{\Pi}(a;\theta))}{d\theta_i} \right] \\
\therefore \quad & \widehat{\Delta \eta(\theta_i)} = R_t \cdot \frac{d \log(\mathbf{\Pi}(a;\theta))}{d\theta_i}
\end{aligned}
$$

In $\widehat{\Delta \eta(\theta_i)}$, R_t can be replaced with $\bar{\mu_t}(a)$ for faster convergence, as it is a better estimate of $\mu_t(a)$. These methods which learn both policy and value function are called '*Actor-Critic*' methods.

Based on the expression derived above, below written is the **Policy Gradient Ascent Algorithm**:

1. Initialize set of parameters $\theta^{(0)} = \{\theta_i^{(0)}\}$ arbitrarily.

2. At time t,

- Play arm $a_t \sim \mathbf{\Pi}(a, \theta^{(t)})$.
- Let's say we observed reward R_t.

3. $\forall\ i$, update $\theta_i^{(t+1)} = \theta_i^{(t)} + \alpha_t \cdot R_t \cdot \frac{d \log(\mathbf{\Pi}(a;\theta))}{d\theta_i}$

Policy gradient methods prove to be useful when dealing directly with policy is easier than estimating action-values, especially when action space and/or observation space are continuous.

2 Contextual Bandits [1]

Till now, the Multi-Armed Bandit problem that we have addressed had only a single state - the reward we get only depended on the arm that we chose. But consider the problem of showing ads to an online user. We have a set of ads $\{a_i\}$, which are the arms, and the reward we get is the number of clicks we get. But here, the reward we get not only depends on the ad we play for the user but also on various factors like the user's age, demographics, gender, etc. This set of features

related to user that affect the reward we get is called '*context*', or current state of the environment, i.e, the true mean of each arm is dependent on the current context or state.

2.1 Using Unsupervised Learning

As we have seen in the above example, the reward we get depends on the feature set of the current user. We can solve this Contextual Bandits problem by dividing it into multiple normal Bandits problems. This can be done by dividing the users into different sets based on their feature vector and then considering each set as a normal Bandits problem. To divide the feature vectors into different sets, clustering algorithms can be used and then each cluster can be treated as a normal Multi-Armed Bandit setting.

2.2 Using Supervised Learning

Let $\boldsymbol{x} \in \mathbb{R}^d$ be the feature vector. In contextual bandits, true mean μ is not dependent just on the arm a, but also on the context vector $\boldsymbol{x}$. For each arm a_i, define a parameter vector $\theta^{a_i} \in \mathbb{R}^d$. The true mean of each arm is estimated as:

$$\mu_{a_i}(\boldsymbol{x}) = (\theta^{a_i})^T \cdot \boldsymbol{x} = \sum_j \theta_j^{a_i} \cdot x_j$$

, where θ^{a_i} is initially unknown and is to be found out. Since we considered mean to be a linear function of features, this method is called **Linear Bandits**. And the policy to choose best arm is given by:

$$a_t = \underset{a_i}{argmax} \sum_j \theta_j^{a_i} \cdot x_j$$

And to estimate θ^{a_i}, Linear Regression can be used. The data that we use to do this is the set of tuples $\{(\boldsymbol{x}_t, a_t, R_t)\}$, which are collected by interacting with the environment. All the samples corresponding to arm a_i are used to solve the Linear Regression problem for the arm a_i.

Since the exploration-exploitation dilemma exists here as well, algorithms like ϵ-greedy, UCB, etc. can be extended to Contextual Bandits.

Lin-UCB is one such algorithm, which is an extension of UCB to Contextual Bandits. The lower bound on the regret of this algorithm is $\mathcal{O}(\sqrt{kdT})$, where k is the number of arms, d is the dimension of feature vector $\boldsymbol{x}$ and T is the time horizon.

References

[1] L. Li, W. Chu, J. Langford, and R. E. Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th international conference on World wide web*. ACM, apr 2010.

[2] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, second edition, 2018.