Optimizing Online Advertising Strategies

Vineet Edupuganti
Stanford University
ve5@stanford.edu

Samir Sen
Stanford University
samirsen@stanford.edu

Abstract

Online advertising is increasingly becoming a predominant source of revenue for major retailers in the global market. The goal of any platform that relies on displaying ads to its user base is to try to find an optimal strategy for deciding which ads to purchase in real-time such that a certain key performance indicator (KPI) is maximized given a constrained budget. In this paper, we explore various approaches for modeling the bidding process including formulating the problem in the Multi-armed Bandits setting as well as considering the policy optimization in terms of a Markov Decision Process with state represented in terms of unspent budget, auction number, and a feature vector representing the current auction. Total clicks or predicted click-through rate (pCTR) serve as the respective rewards for these two settings. Though we optimize for user clicks, this is easily extensible to other performance indicators. We find that the multi-armed bandits formulation using the Upper Confidence Bound and Thompson Sampling algorithms provide the most promising outcomes on a simulated dataset in comparison to a random policy, and that optimizing CTR in a constrained budget setting using MDP value iteration results in strong improvement over random policies.

1 Introduction

Online advertising is a major industry worldwide, accounting for an estimated $333.25 billion in global spend in 2019 [1]. For many businesses, ads are the most commonly used method for driving customer happiness, conversions, and revenue. The way the digital ad system operates usually resembles an auction model in which multiple companies bid for the right to show their content to a given user. Thus, companies must intelligently allocate their limited money towards the best set of bids that can maximize future revenue.

The standard method that companies have used in digital campaigns to validate which advertisements to show and when is A/B testing, an approach that relies on showing different groups of users different content, evaluating the resulting behavior, and statistically analyzing the results [2]. However, there are numerous drawbacks with this approach. For one, the process of running A/B tests can be very long and result in substantial expense. Many organizations have multiple different versions of an ad that they would like to test, leading to several weeks of experimentation (sequentially) and extensive engineering and data science efforts [3]. Secondly, these tests are conducted at a given moment of time without any consideration for ad campaigns that have occurred earlier, which inherently limits the richness of the conclusions drawn. Finally, A/B tests are static in nature, meaning that if data distributions and user preferences change over time, a new A/B test must be run to reflect this updated information.

Given these concerns, reinforcement learning is an attractive method for enabling businesses to optimize future ad revenue. By balancing exploration and exploitation in real time, reinforcement learning allows for dynamic decision making regarding which ad to display to a user so as to maximize

.

overall engagement throughout a campaign. Furthermore, reinforcement learning offers flexibility to adjust to changes in user preferences while still prioritizing the ads that are most effective.

In this work, we introduce reinforcement learning approaches to optimize the process of deciding which ad to show a user in a web-based setting. We utilize two primary methods with slightly different applications: Multi-armed Bandits and Markov Decision Processes (MDPs). In the following sections, we describe the datasets used and the results achieved using various algorithms.

2 Background and Related Work

Bidding strategy is one of the critical parts of online advertising. In real-time bidding, a Demand-Side Platform (DSP) moderates the process of competing bids from multiple advertisers, with the winner of bid being determined by the Ad Exchange (which facilitates the auction transaction of buying and selling the ad impressions). Each advertiser receives partial information of the auction through the DSP and is then permitted to make a bid with millisecond latency [4]. Thus, it is crucial to learn a policy mapping such that with high probability a chosen bidding price for a given auction maximizes the KPI of interest (in our case CTR) over t auctions. Previous work has considered linear bidding strategies, where bid price for a given auction is linearly higher for ads with higher expected CTR [5]. Jin et. al., on the other hand, considered the multi-armed bandits setting attempting to choose bid price in such a way that the expected regret bound after t auctions is minimized [6]. However, in both approaches the additional constraint of budget is not a primary consideration (Jin et al. minimize total purchase cost but do not perform the optimization given a constrained budget).

Cai et. al. approached bidding optimization using MDP's through a two-step procedure [7]. First, they use the existing partial auction binary features from the iPinyou public auction dataset to predict CTR using logistic regression and use this as the reward for the MDP. To validate the choice of reward, they compute the mutual information between the market price (part of iPinyou dataset) and the predicted CTR to show a high correlation. Then, they use value iteration and linear programming to optimize the MDP such that each bidding choice considers a constrained budget and finds a strategy that conforms closely to the predicted CTR. Du et. al. build off of this approach considering the constrained MDP but using a non-linear program to optimize for expected cost per click rather than CTR which led to an overall improvement in bidding strategy [8].

3 Multi-armed Bandits

3.1 Approach

We can formulate the ad optimization problem as follows. We have d different ads, any of which can be shown to n successive users. At each time step, we monitor whether or not the user clicks on the ad we have chosen (assume this happens in real time). If a click occurs, we receive a reward r of 1 and otherwise a reward of 0. The objective is to choose an ad at each time step so as to maximize the reward, only knowing the previous sequence of actions and rewards. Note that this method ignores the budget of the advertiser and assumes they have the bandwidth to show an ad to each user.

3.2 Dataset

Due to the lack of publicly-available and relevant datasets, we synthetically generated a dataset with $n = 50,000$ and $d = 9$. These values, while somewhat arbitrary, were chosen because they resemble a realistic scenario for most companies, given that many sites would have about $50,000$ visitors over the course of a few weeks and most likely have fewer than 10 variations of an ad to test.

The data for this problem consisted of a $n * d$ matrix of users and the ads they would click on, as represented by 1's and 0's. For example, if $data[5, 3] = 1$, then user 5 would click on ad 3, meaning we would get a reward of 1 if we chose to display ad 3 to user 5. Note that while only one ad can be chosen at a given timestep (for a maximum reward of 1), multiple columns in a given row can have a value of 1 to represent the possibility of a user having affinity for multiple ads. To create the simulated data, we seeded the first row of the matrix with a single 1 corresponding to a random ad. For

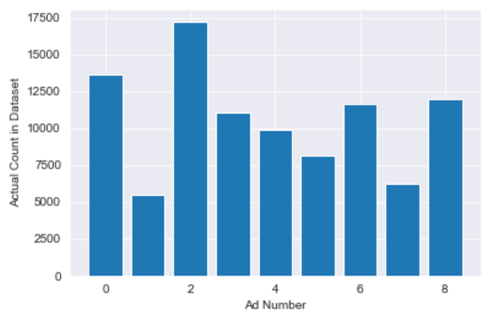

Figure 1: Click distribution for ads in simulated dataset

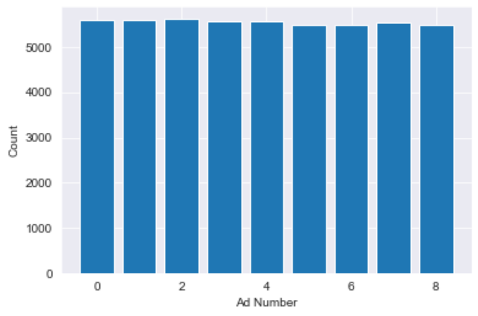

Figure 2: Distribution of ads chosen randomly

any ensuing row (user m), for each ad d_i, we define $P(\text{user m clicks on } d_i|\text{user m-1 clicked on } d_i)$ and $P(\text{user m clicks on } d_i|\text{user m-1 did not click on } d_i)$ to be random values between 0.05 and 0.3 upfront. While there are many valid approaches that we explored, this method was effective in ensuring sparsity and in most datasets generated (including the one used for experiments) resulted in some asymmetry among total clicks per ad, allowing for more interesting analysis.

3.3 Data Distribution

To gain a better sense of what the simulated data looks like using our approach, we produced a histogram of how many potential clicks (i.e. 1's) there were for each ad in question. From Fig. 1, we see that ad 2 and ad 0 have the highest counts, while ad 1 and ad 7 have the lowest. Thus, we would expect any reasonable policy to show the former ads more often and the latter ones less often.

3.4 Random Policy

As a baseline, we first explore a random policy whereby each user is shown a random ad. Figure 2 shows the number of times each ad is shown to a user. As expected, the distribution is uniform, while Table 1 indicates that this policy performs very poorly.

3.5 ϵ-Greedy

The ϵ-greedy algorithm balances exploration with exploitation by choosing a random action with probability ϵ and the best action based on prior data with probability $(1-\epsilon)$ [9]. Figure 3 shows that reward increases with the value of ϵ up till $\epsilon = 0.05$, at which point there is a steady decay in rewards corresponding to underexploiting the best ads. As shown in Table 1, this method has substantial variance, which meant that we had to obtain multiple samples at each value of ϵ and then compute the mean to produce reasonable reward estimates.

3.6 Upper Confidence Bound (UCB)

The UCB algorithm results in choosing the action (ad) with the highest upper confidence bound at any given point in time. This approach enables the exploitation of actions with high average rewards and low variances in addition to the exploration of actions that we still have significant uncertainty about. More specifically, in accordance with the UCB1 version of the algorithm, we choose the ad that maximizes the upper bound for each user m, where x_j represents the average reward associated with ad j and n_j represents the number of times we have shown that ad previously, as shown below [10].

$$j^* = \underset{j}{\operatorname{argmax}}\, \bar{x_j} + \sqrt{\frac{3}{2} log(\frac{m}{n_j})}$$

As expected, given the more sophisticated management of exploration and exploitation, UCB gives the best results on the dataset as shown in Table 1. Note that this algorithm is deterministic since we do not take extra effort to vary the first ad chosen. Figure 4 shows that ad number 2 is almost always

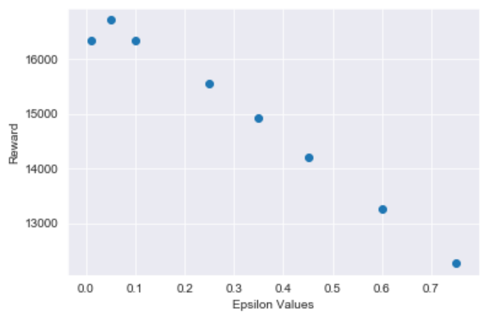

Figure 3: Reward versus value of ϵ

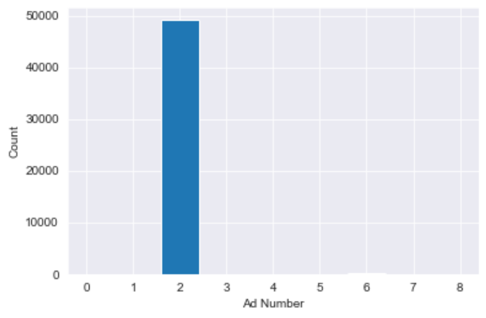

Figure 4: Distribution of ads chosen for the users with the UCB algorithm

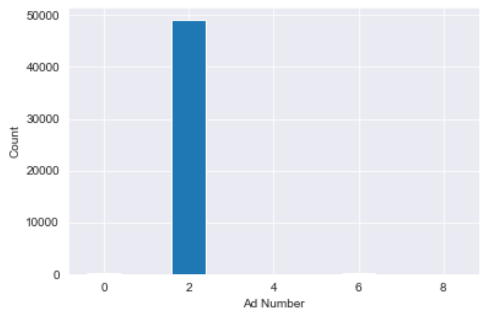

Figure 5: Distribution of ads chosen for the users with the Thompson Sampling algorithm

shown to the users. Given the distribution of the data in Fig. 1, we can determine that the superior performance is a result of heavily exploiting the ad with the highest engagement.

3.7 Thompson Sampling

The final algorithm we explore is Thompson sampling. For each ad, we maintain a Beta distribution of successes and failures (using a uniform prior). At any point in time, we can draw samples from each ad's distribution and select the ad that gives us the highest value (high chance of receiving a reward). Based on whether we observe a success (click) or failure, we can update the chosen ad's Beta distribution and continue the process with all ensuing users [11]. Given the relative simplicity of the data, Thompson sampling is reasonably efficient and can quickly converge on the set of ads that are most beneficial to repeatedly show.

Table 1 shows that Thompson Sampling is on par with the UCB algorithm, yielding slightly better results in some cases and slightly worse ones in others. Figure 5 shows that the decision making between the two algorithms is also very similar, although Thompson Sampling chooses ad 0 slightly more often. Irrespective, we see that both methods are better than ϵ-greedy (even with an optimized ϵ value), and far better than the random baseline, indicating the potential for certain reinforcement learning algorithms to aid in real-time ad selection.

4 Markov Decision Processes (MDPs)

Given that the constraint of budget is ignored in the Multi-armed Bandits case, we developed an MDP approach to model the bidding process using the publicly available iPinyou auction dataset. This dataset contains partial auction information with 10,000 rows per location and is stripped of all sensitive PII data. Each row contains 12 columns including timestamp, location, IP address, bidding price, market price, paying price, user clicked, and advertisement id. The data is heavily skewed with

	Mean Reward	Standard Deviation
Random	10574.4	85.6
ϵ-greedy ($\epsilon = 0.05$)	16725.6	263.3
UCB	17128.0	N/A
Thompson Sampling	17105.7	23.0

Table 1: Results using different algorithms on the simulated dataset

	Clicks	Win-Rate	Cost ($)
Random	19	2.31%	744,883
MDP	271	16.56%	687,794

Table 2: Results using different algorithms on the iPinyou dataset

$< 2\%$ of auctions having being clicked by the user. We apply the preprocessing approach adopted in Cai et. al. to reduce the dataset to include only whether the user clicked the ad, the market price, and the predicted click-through rate of the auction (using logistic regression of the auction feature vector from the original dataset) [7]. We split the resulting preprocessed dataset into training and validation sets with 15% of the data left for validation.

To formulate the task concretely, we represent the bidding process in terms of a Markov Decision Process constrained by budget where the we define the MDP in terms of a <*S, A, P, R, C*> tuple. We consider Click-through rate (CTR) and remaining budget as the *state*, bidding price as the *action*, number of clicks as the *reward* to maximize, and market price as the *cost*. We represent the transition probability as a function of the predicted CTR independent of the current state (S) or action (A). Thus, $P(S'|S, A) = p_{ctr}(S')$ as shown in [7] and is consistent with the MDP as state is represented in terms of CTR. Thus, the objective of the MDP is to solve the optimization problem and find the probabilistic policy map such that R is maximal with budget B:

$$max_p R = \sum_{s,a} p(s,a) \cdot R(s,a)$$

$$s.t \sum_{s,a} p(s,a) \cdot C(s,a) \leq B$$

Here $p(s, a)$ represents the probability of being in state s and taking action a. We want to the find the optimal policy distribution that maximizes the reward considering a budget constraint. We leverage *scipy* linear program libraries to compute the optimization over the preprocessed dataset.

Table 2 shows the results when comparing with a random bidding choice over the held out testing set of preprocessed data. Here, the random policy is generated by choosing a random bid price within the range of remaining budget at each given auction until the total budget has run out. Then, we use the truth values to compute performance statistics of each of the algorithms. We see that the MDP significantly outperforms random bidding in terms of both win rate and cost, while the random approach seems to approximate the initial click-rate skew found in the original data.

5 Conclusion

Ultimately, we see that both the Multi-armed Bandit and Markov Decision Process approaches to formulating advertisement optimization have substantial value, while overcoming efficiency and scale issues that arise with industry-standard techniques like A/B testing. In the case of Multi-armed Bandits, both the UCB and Thompson Sampling approaches perform very well comparing to random policy. In the MDP formulation, we find that the optimized policy distribution results in significantly higher number of clicks and won bids compared to randomly assigning a bid price considering total budget.

The primary shortcoming with the Multi-armed Bandit experiments conducted is that the simulated dataset, while designed to represent a real-world scenario, is likely not perfect. Thus, running our models on real data would add another layer of context to our results. Furthermore, given more time, additional effort could have been placed on exploring other, newer algorithms and variants (UCB-ALP, for instance).

In terms of the MDP approaches, in future work, we can attempt to improve on the algorithms by considering a continuous space of actions and state rather than the discrete setting as analyzed in this paper. It would be interesting to explore any improvements to number of clicks or cost efficiency that would result through reinforcement learning algorithms such as Q-learning.

Overall, this set of approaches presents an exciting set of possibilities for modernizing the advertising industry, and additional work centered on practicality can help bring this to fruition.

6 Contributions

Both V.E and S.S discussed and analyzed the background, related work, and problem formulation. V.E was responsible for all experimental work pertaining to the Multi-armed Bandits section, while S.S focused efforts on exploring Markov Decision Processes. The writing of this paper was split along those lines, with all other sections being equally divided. All code can be found at `https://github.com/samirsen/rtb-agent`.

References

[1] Jasmine Enberg. *Global Digital Ad Spending 2019*, 2019.

[2] Joel Barajas, Ram Akella, Marius Holtan, Jaimie Kwon, Aaron Flores, and Victor Andrei. Dynamic effects of ad impressions on commercial actions in display advertising. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, pages 1747–1751. ACM, 2012.

[3] Peep Laja. *12 A/B Testing Mistakes I See All the Time*, 2019.

[4] Vinh Gu. Real time bidding, (rtb), technology. 2019.

[5] Wilbur Joel, Dan. Budget optimization for sponsored search. 2017.

[6] Varisteas Jin, Sassioui. Improving real-time bidding using a constrained markov decision process. In *CoRR*, pages 39–1, 2018.

[7] Zhang Cai, Ren. Real-time bidding by reinforcement learning in display advertising. In *ELCC-PLC*, pages 39–1, 2018.

[8] Yuan Du, Zhang. Real-time bidding benchmarking with ipinyou dataset. 2019.

[9] Michel Tokic and Günther Palm. Value-difference based exploration: adaptive control between epsilon-greedy and softmax. In *Annual Conference on Artificial Intelligence*, pages 335–346. Springer, 2011.

[10] Alexandra Carpentier, Alessandro Lazaric, Mohammad Ghavamzadeh, Rémi Munos, and Peter Auer. Upper-confidence-bound algorithms for active learning in multi-armed bandits. In *International Conference on Algorithmic Learning Theory*, pages 189–203. Springer, 2011.

[11] Shipra Agrawal and Navin Goyal. Analysis of thompson sampling for the multi-armed bandit problem. In *Conference on Learning Theory*, pages 39–1, 2012.