Ad Recommendation Systems for Life-Time Value Optimization

Georgios Theocharous
Adobe Research
theochar@adobe.com

Philip S. Thomas
UMassAmherst
and Adobe Research
phithoma@adobe.com

Mohammad Ghavamzadeh
Adobe Research and INRIA
ghavamza@adobe.com

ABSTRACT

The main objective in the ad recommendation problem is to find a strategy that, for each visitor of the website, selects the ad that has the highest probability of being clicked. This strategy could be computed using supervised learning or contextual bandit algorithms, which treat two visits of the same user as two separate independent visitors, and thus, optimize greedily for a single step into the future. Another approach would be to use reinforcement learning (RL) methods, which differentiate between two visits of the same user and two different visitors, and thus, optimizes for multiple steps into the future or the life-time value (LTV) of a customer. While greedy methods have been well-studied, the LTV approach is still in its infancy, mainly due to two fundamental challenges: how to compute a good LTV strategy and how to evaluate a solution using historical data to ensure its "safety" before deployment. In this paper, we tackle both of these challenges by proposing to use a family of off-policy evaluation techniques with statistical guarantees about the performance of a new strategy. We apply these methods to a real ad recommendation problem, both for evaluating the final performance and for optimizing the parameters of the RL algorithm. Our results show that our LTV optimization algorithm equipped with these off-policy evaluation techniques outperforms the greedy approaches. They also give fundamental insights on the difference between the click through rate (CTR) and LTV metrics for performance evaluation in the ad recommendation problem.

Categories and Subject Descriptors

I.2 [**Artificial Intelligence**]: Learning, Control Methods; H.1.2 [**User/Machine Systems**]: Human information processing

Keywords

Ad recommendation, reinforcement learning, off-policy evaluation

1. INTRODUCTION

The goal of most ad recommendation problems is to generate a strategy that, for each user of the website, selects an ad that, if presented, has the highest probability of being clicked. These days, almost all industrial ad recommendation systems use supervised learning or contextual bandit algorithms (especially contextual bandits that also take into account the important problem of *exploration*). These methods are based on the i.i.d. assumption of the visits (to the website) and do not discriminate between a visit and a visitor, i.e., each visit is considered as a new visitor that has been sampled i.i.d. from the population of the visitors. As a result, these algorithms are greedy and do not try to optimize for long-term performance. The click through rate (CTR) is a good metric to evaluate the performance of such greedy algorithms. Despite their success, these methods seem to be insufficient as users establish longer-term relationships with the websites they visit by returning back. This increase in *returning visitors* further violates the foundational assumption underlying the supervised learning and bandit algorithms: that there is no difference between a visit and a visitor. This insufficiency of the current approaches motivates the new class of solutions that we propose.

The reinforcement learning (RL) algorithms that have been designed to optimize the long-term performance of the system (expected sum of rewards/costs) seem to be a suitable candidate for ad recommendation systems. The nature of these algorithms allows them to take into account all the available knowledge about the user at the current visit, and to then select an offer that maximizes the total number of times she will click over multiple visits, also known as the user's life-time value (LTV). Unlike greedy approaches, RL algorithms differentiate between a visit and a visitor, and consider all the visits of a user (in chronological order) as a system trajectory generated by her. In this approach, while the visitors are i.i.d. samples from the population of the users, their visits are not. This means that although we can evaluate the performance of the RL algorithms using CTR, this is not the metric that they optimize, and they should be evaluated based on the expected total number of clicks per user (trajectory), a metric we call LTV. This long-term approach to the ad recommendation problem allows us to make decisions that are better than the short-sighted decisions made by greedy techniques, such as to propose an offer to a user that might be a loss to the company in the short term, but has an effect that makes the user engaged with the website/company and brings her back to spend more money in the future.

Despite these desirable properties, there are two major obstacles hindering the widespread application of the RL technology in ad recommendation: **1)** how to compute a good LTV policy in a scalable way and **2)** how to evaluate the performance of a policy returned by a RL algorithm without deploying it, and only using the historical data that has been generated by one or more other policies. The second problem, also known as *off-policy evaluation*, is of extreme importance not only in recommendation systems and online marketing, but in many other domains such as health care and finance. It may also help us with the first problem, in selecting the right representation (features) for the RL algorithm and in optimizing its parameters, which in turn will help to have a more scalable algorithm and to generate better policies. Unfortunately, unlike the greedy algorithms for which there exist several *biased* and *unbiased* off-policy evaluation techniques (e.g., [9, 15, 8]), there are not many applied, yet theoretically founded, methods to guarantee that the obtained policy performs well in the real system without having a chance to deploy/execute it.

One approach to tackle this issue is to first build a model of the real world (a simulator) and then use it to evaluate the performance of a RL policy [17]. The drawback of this *model-based* approach is that accurate simulators, especially for recommendation systems, are notoriously hard to learn. In this paper, we use a recently proposed *model-free* approach that computes a lower confidence bound on the expected return of a policy using a concentration inequality [18] to tackle the off-policy evaluation problem. We also use two approximate techniques for computing this lower confidence bound (instead of the concentration inequality), one based on Student's t-test [20] and the other based on bootstrap sampling [4].

This off-policy evaluation method takes as input historical data from existing policies, a baseline policy, a new policy, and a confidence level, and outputs whether the new policy is better than the baseline, with the given confidence. This high confidence off-policy evaluation method plays a crucial role in many aspects of building a successful RL-based ad recommendation system. First, it allows us to select a champion in a set of policies without the need to deploy expensive A/B testing. Second, it can be used to select a good set of features for the RL algorithm, and in effect to scale it up. Third, it can be used to tune the RL algorithm. For example, many batch RL algorithms such as fitted Q iteration (FQI) [5] do not have a monotonically improving performance along their iterations. Thus, an off-policy evaluation framework is useful to keep track of the best performing strategy along the iterations.

In general, using RL to develop algorithms for LTV marketing is still in its infancy. Related work has used toy examples and has appeared mostly in marketing venues [12, 7, 19]. An approach directly related to our work first appeared in [11], where the authors used public data of an email charity campaign and showed that RL policies produce better results than myopic. They used batch RL methods and heuristic simulators for evaluation. [14] recently proposed an on-line RL system that learns concurrently from multiple customers. The system was trained and tested on a simulator and does not offer any performance guarantees. Unlike previous work, we deal with real data, where we are faced with the challenges of learning RL policies in a high dimensional problem and evaluating these policies in an off-policy fashion, with guarantees.

In the rest of the paper, we first summarize the different methods for computing lower bounds on the performance of a policy. We then describe the difference between CTR and LTV metrics for policy evaluation in the ad recommendation problem, and the fact that CTR could lead to misleading results when we have returning visitors. We present practical algorithms for myopic and LTV optimization that combine various powerful ingredients, such as the robustness of random-forest regression, feature selection, and off-policy evaluation for parameter optimization. Finally, we finish with experimental results that demonstrate how our LTV optimization algorithm outperforms a greedy approach.

2. PRELIMINARIES

We assume that the environment can be modeled as a *Markov decision process* (MDP), which is a tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, d_0, \gamma)$, where $\mathcal{S}$ is the set of possible states, which may be finite, countably infinite, or continuous (uncountable), $\mathcal{A}$ is the finite set of admissible actions, $\mathcal{P}(s, a, s')$ is the probability of transitioning to $s' \in \mathcal{S}$ when action $a \in \mathcal{A}$ is taken in state $s \in \mathcal{S}$, $\mathcal{R}(s, a) \in \mathbb{R}$ is the reward received when action a is taken in state s, d_0 is the initial distribution over states, and $\gamma \in [0, 1]$ is a parameter for discounting rewards. Furthermore, we assume that the MDP reaches a terminal state within T transitions.

In the context of our ad recommendation problem, $\mathcal{S}$ is the set of possible feature vectors that describe a user, $\mathcal{A}$ is the set of ads that can be displayed, $\mathcal{P}$ governs the (unknown) dynamics of the users, including whether or not they click on an ad, $\mathcal{R}(s, a)$ is 1 when a user in state s clicks on the ad a, and 0 otherwise. We also set the system horizon to $T = 20$, since in our data we rarely have a user with more than 20 visits to the website.

The agent's decision rule, which we call a *policy*, π, is such that $\pi(a|s)$ denotes the probability of taking action a in state s. Each episode (a sequence of at most T changes to the state, starting from a state drawn from d_0), produces a *trajectory*, $\tau = \{s_1, a_1, r_1, s_2, a_2, r_2, \ldots, s_T, a_T, r_T\}$. We define $R(\tau)$ to be the *return* of trajectory τ, i.e., the sum of the (discounted) rewards observed along τ. In the off-policy evaluation method, we use the normalized discounted sum of clicks, and define $R(\tau)$ as

$$R(\tau) := \frac{\sum_{t=1}^{T} \gamma^{t-1} r_t - R_-}{R_+ - R_-},$$

where r_t is the reward at time t and R_- and R_+ are upper and lower bounds on $\sum_{t=1}^{T} \gamma^{t-1} r_t$. The goal is to find a policy that maximizes the expected performance

$$\rho(\pi) := \mathbb{E}[R(\tau)|\pi].$$

The goal in RL is to search for a policy that maximizes this expected performance, $\rho(\pi)$ [16].

We assume that a policy (or policies) has been deployed to produce a history of data, $\mathcal{D}$. Formally, $\mathcal{D}$ contains n trajectories $\{\tau_i\}_{i=1}^{n}$, each labeled with the policy π_i that produced it. We call these policies, *behavior policies*, because they were used to control the past behavior of the system, and the policy produced by a RL algorithm (e.g., fitted Q-iteration or least squares policy iteration) that we would like to evaluate its performance, the *evaluation policy*, π.

3. BACKGROUND: OFF-POLICY EVALUATION WITH PROBABILISTIC GUARANTEES

In this section we review three approaches to off-policy evaluation that provide strong probabilistic guarantees about the performance of an evaluation policy, π_e, using only the available historical data, $\mathcal{D}$. Specifically, they compute a lower bound, ρ_-, on the true performance, $\rho(\pi)$, for any confidence, $1-\delta \in [0,1]$. All three approaches use *importance sampling* [13] to create an unbiased estimate of $\rho(\pi_e)$ from each trajectory, $\tau \in \mathcal{D}$. We write $\hat\rho(\pi_e|\tau_i,\pi_i)$ to denote this estimator, called the *importance weighted return*, i.e.,

$$\hat\rho(\pi_e|\tau_i,\pi_i) := R(\tau_i)\prod_{t=1}^{T}\frac{\pi_e(a_t^{\tau_i}|s_t^{\tau_i})}{\pi_i(a_t^{\tau_i}|s_t^{\tau_i})}.$$

For brevity, we use X_i to represent the random variable $\hat\rho(\pi_e,\tau_i,\pi_i)$. We also use $\bar\rho := \frac{1}{n}\sum_{i=1}^{n}\hat\rho(\pi_e,\tau_i,\pi_i)$ to denote the sample mean of the importance weighted returns. Since each of the importance weighted returns is an unbiased estimate of $\rho(\pi)$, so is their sample mean, i.e., $\mathbb{E}[\bar\rho] = \rho(\theta)$.

3.1 Concentration Inequality (CI)

A straightforward approach to provide lower confidence bounds on the performance of the evaluation policy, $\rho(\pi)$, would be to use the Chernoff-Hoeffding inequality to bound it using $\bar\rho$. However, as shown by [18], this inequality is not well-suited to this application due to its sensitivity to the range of the importance weighted returns.

[18] then derived a concentration inequality that *is* well-suited to this application by extending the empirical Bernstein bound in [10] to have less dependence on the range of the random variables. This is achieved by bounding a statistic similar to the truncated mean (unlike the truncated mean, their statistic does not require discarding data). They also present a system that automatically estimates the optimal threshold beyond which data is truncated. We write $\rho_-^{\mathrm{CI}}(\mathbf{X},\delta)$ to denote the lower bound produced by this approach, where $\mathbf{X}$ is any set of random variables and $1-\delta$ is the desired confidence level.

3.2 Student's t-Test

The CI approach is safe but overly conservative. One way to improve it is to introduce additional reasonable assumptions, which can be leveraged to achieve a tighter (less conservative) bound. One way to do this is to utilize the central limit theorem (CLT). Specifically, by the CLT, the sample mean of many samples from *any* distribution is approximately normally distributed. For our application, this means that $\bar\rho$ becomes approximately normally distributed as the number of trajectories used to compute it increases. Therefore, if we assume that $\bar\rho$ is normally distributed, we can use Student's t-test to get a significantly tighter bound than that of the CI approach (which holds in the more general case where there are no assumptions about the true distribution of $\bar\rho$).

While the t-test can produce a significantly tighter bound than the CI approach, its bound is only approximate due to the false assumption that $\bar\rho$ is normally distributed. This means that, when we desire an error rate of at most δ, Student's t-test may result in a higher error rate. However, this is not a significant problem for two reasons. First, in our application we will have tens of thousands of users, each of which produces a trajectory. With this many trajectories, by the CLT, $\bar\rho$ might be almost normally distributed. Second, because the importance weighted returns tend to come from a distribution with a heavy upper tail [18], the t-test tends to produce overly conservative lower bounds. We write $\rho_-^{\mathrm{TT}}(\mathbf{X},\delta)$ to denote the lower bound produced by this approach, where $\mathbf{X}$ is any set of random variables and $1-\delta$ is the desired confidence level. More formally, we have

$$\hat X := \frac{1}{n}\sum_{i=1}^{n}X_i, \qquad \sigma := \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}\left(X_i-\hat X\right)^2},$$

$$\rho_-^{\mathrm{TT}}(\mathbf{X},\delta) := \hat X - \frac{\sigma}{\sqrt{n}}t_{1-\delta,n-1},$$

where $t_{1-\delta,\nu}$ denotes the inverse of the cumulative distribution function of the Student's t distribution with ν degrees of freedom, evaluated at the probability $1-\delta$ (i.e., $\mathrm{tinv}(1-\delta,\nu)$ in MATLAB).

3.3 Bias Corrected and Accelerated Bootstrap

As mentioned in Section 3.2, when $\bar\rho$ is far from being normally distributed, the lower bound produced by the t-test approach, $\rho_-^{\mathrm{TT}}(\mathbf{X},\delta)$, may not be accurate. One way to address this issue is first to use bootstrapping to estimate $\bar\rho$'s true distribution, then use this estimate to transform the data so that it is approximately normally distributed, and finally apply the t-test to this transformed data. A popular method for this sort of bootstrapping is *Bias Corrected and accelerated* (BCa) bootstrap [4]. We write $\rho_-^{\mathrm{BCa}}(\mathbf{X},\delta)$ to denote the lower-bound produced by BCa, where $\mathbf{X}$ is any set of random variables and $1-\delta$ is the desired confidence level.

The primary drawback of this approach is that the distribution of $\bar\rho$ is only approximated, and thus, after being transformed it is still only *approximately* normally distributed. This means that the t-test still does not produce an exact bound. However, by correcting for the heavy tails in our data, BCa tends to produce lower bounds that are not as overly-conservative as using Student's t-test directly. Although the bounds produced by BCa are only approximate (they can have an error rate higher than δ), they have been considered reliable enough to be used in many different domains, particularly in medical fields [3, 6]. Detailed information on the implementation of BCa can be found in [2].

4. CTR VERSUS LTV

Any ad recommendation policy could be evaluated for its greedy/myopic or long-term performance. For greedy performance, click through rate (CTR) is a reasonable metric, while lifetime value (LTV) seems to be the right choice for long-term performance. These two metrics are formally defined as follows:

$$\mathrm{CTR} = \frac{\text{Total \# of Clicks}}{\text{Total \# of }\textbf{Visits}}\times 100,$$

$$\mathrm{LTV} = \frac{\text{Total \# of Clicks}}{\text{Total \# of }\textbf{Visitors}}\times 100.$$

CTR is a well-established metric in digital advertising and can be estimated from historical data (off-policy) in unbiased (inverse propensity scoring; [9]) and biased [15] ways. [18] recently proposed a practical approach for LTV estimation, which we extend, by replacing the concentration inequality both with t-test and BCa, and apply for the first time to real online advertisement data. The main reason that we use LTV is that CTR is not a good metric for evaluating long-term performance and could lead to misleading conclusions. Imagine a greedy advertising strategy at a website that directly displays an ad related to the final product that a user could buy. For example, it could be the BMW website and an ad that offers a discount to the user if she buys a car. Users who are presented such an offer would either take it right away or move away from the website. Now imagine another marketing strategy that aims to transition the user down a sales funnel before presenting her the discount. For example, at the BMW website one could be first presented with an attractive finance offer and a great service department deal before the final discount being presented. Such a long-term strategy would incur more interactions with the customer and would eventually produce more clicks per customer and more purchases. The crucial insight here is that the policy can change the number of times that a user will be shown an advertisement—the length of a trajectory depends on the actions that are chosen. A possible visualization of this concept is presented in Figure 1.

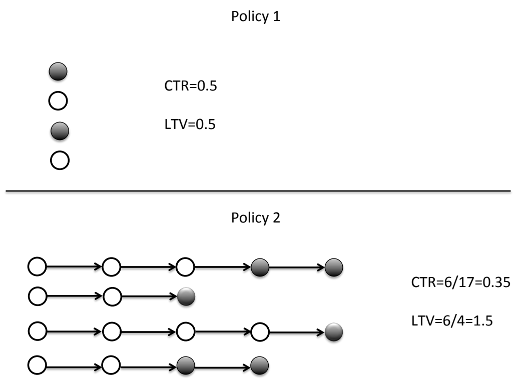


Figure 1: The circles indicate user visits. The black circles indicate clicks. Policy 1 is greedy and users do not return. Policy 2 optimizes for the long-run, and users come back multiple times and click toward the end. Even though Policy 2 has a lower CTR than Policy 1, it results in more revenue, as captured by the higher LTV. Therefore, LTV is a better metric for evaluating policies for ad recommendation than CTR.

5. RECOMMENDATION ALGORITHMS

For greedy optimization, we used the random forest algorithm [1] to learn to map features to actions. Random-forests is a state of the art ensemble learning method for regression and classification, which is robust to overfitting, and which is often used in industry for big data problems. The system is trained by using a random forest for each of the offers/actions to predict the immediate reward. During execution we use an epsilon-greedy strategy where we choose the max prediction with probability 0.9 and amongst the rest of the offers with probability $0.1/(|A|-1)$, see Algorithm 1.

Algorithm 1 GREEDYOPTIMIZATION($\mathbf{X}_{\text{train}}, \mathbf{X}_{\text{test}}, \delta$) : compute a greedy strategy using $\mathbf{X}_{\text{train}}$, and predict the $1-\delta$ lower bound on the test data $\mathbf{X}_{\text{test}}$ and the value function.

```
1: $y = \mathbf{X}_{\text{train}}(\text{reward})$
2: $x = \mathbf{X}_{\text{train}}(\text{features})$
3: $\bar{x} = \text{informationGain}(x, y)$ {feature selection}
4: $\text{rf}_a = \text{randomForest}(\bar{x}, y)$ {for each action}
5: $\pi_e = \text{epsilonGreedy}(\text{rf}, \mathbf{X}_{\text{test}})$
6: $\pi_b = \text{randomPolicy}$
7: $W = \hat{\rho}(\pi_e|\mathbf{X}_{\text{test}}, \pi_b)$ {importance weighted returns}
8: return $(\rho^{\dagger}_{-}(W, \delta), \text{rf})$ {bound and random forest}
```

For the LTV optimization problem we used a state of the art RL algorithm called FQI [5], which is able to handle high dimensional continuous and discrete variables. Due to various implementation constrains of the company we used the random forest as the function approximator. When an arbitrary function approximator is used in the FQI algorithm it does not converge monotonically but rather oscillates during training iterations. To alleviate oscillation problems of FQI using random forest and for better feature selection, we used our evaluation framework within the training loop. The loop keeps track of the best FQI result according to a validation data set, see Algorithm 2.

Algorithm 2 LTVOPTIMIZATION($\mathbf{X}_{\text{train}}, \mathbf{X}_{\text{val}}, \mathbf{X}_{\text{test}}, \delta, K, \gamma$) : compute a LTV strategy using $\mathbf{X}_{\text{train}}$, and predict the $1-\delta$ lower bound on the test data $\mathbf{X}_{\text{test}}$

```
1: $\pi_b = \text{randomPolicy}$
2: $Q = \text{RF.GREEDY}(\mathbf{X}_{\text{train}}, \mathbf{X}_{\text{test}}, \delta)$ {start with greedy value function}
3: for $i = 1$ to $K$ do
4:     $r = \mathbf{X}_{\text{train}}(\text{reward})$ {use recurrent visits}
5:     $x = \mathbf{X}_{\text{train}}(\text{features})$
6:     $y = r_t + \gamma \max_{a \in A} Q_a(x_{t+1})$
7:     $\bar{x} = \text{informationGain}(x, y)$ {feature selection}
8:     $Q_a = \text{randomForest}(\bar{x}, y)$ {for each action}
9:     $\pi_e = \text{epsilonGreedy}(Q, \mathbf{X}_{\text{val}})$
10:    $W = \hat{\rho}(\pi_e|\mathbf{X}_{\text{val}}, \pi_b)$ {importance weighted returns}
11:    $\text{currBound} = \rho^{\dagger}_{-}(W, \delta)$
12:    if $\text{currBound} > \text{prevBound}$ then
13:        $\text{prevBound} = \text{currBound}$
14:        $Q_{\text{best}} = Q$
15:    end if
16: end for
17: $\pi_e = \text{epsilonGreedy}(Q_{\text{best}}, \mathbf{X}_{\text{test}})$
18: $W = \hat{\rho}(\pi_e|\mathbf{X}_{\text{test}}, \pi_b)$
19: return $\rho^{\dagger}_{-}(W, \delta)$ {lower bound}
.
```

6. EXPERIMENTS

For our experiments we used a data set from the banking industry. On the company website when customers visit, they are shown one of a finite number of offers. The reward is one when a user clicks on the offer and zero otherwise. We extracted/created features, in the categories shown in Table 1. We collected data from a particular campaign for a month that had seven offers and approximately 200,000 interactions. About 20,000 of the interactions were produced by a random strategy. When users visit the bank web-site the first time, they are assigned either to a random strategy or a targeting strategy for the rest of the campaign lifetime. We split the random strategy data into a test set and a validation set. We use the targeting data for training to optimize the greedy and LTV strategies described in Algorithms 1 and 2. We used aggressive feature selection, for the greedy strategy and selected 20% of the features. For LTV the feature selection had to be even more aggressive due to the fact that the number of recurring visits is approximately 5%. We used information gain for the feature selection module. With our algorithms we produce performance results both for the CTR and the LTV metric. To produce results for CTR we assumed that each visit is a unique visitor.

Cum action	There is one variable for each offer, which counts the number of times each offer was shown
Visit time recency	Time since last visit
Cum success	Sum of previous reward
Visit	The number of visits so far
Success recency	The last time there was positive reward
Longitude	Geographic location [Degrees]
Latitude	Geographic location [Degrees]
Day of week	Any of the 7 days
User hour	Any of the 24 hours
Local hour	Any of the 24 hours
User hour type	Any of weekday-free, weekday-busy, weekend
Operating system	Any of unknown, windows, mac, linux
Interests	There are finite number of interests for each company. Each interest is a variable hat gets a score according to the content of areas visited within the company websites
Demographics	There are many variables in this category such as age, income, home value...

Table 1: Features

From our experimental results we first observed that without any feature selection we would not see any lift from the random policy. With aggressive feature selection we are able to produce incredible lifts using both the greedy and LTV approaches. Second, we observed that every strategy has both a CTR and an LTV metric, as shown in Figure 2. Third, we observed that the GreedyOptimization algorithm performs the best under the CTR metric and the LtvOptimization algorithm performs the best under the LTV metric as expected, see Figures 3(a) and 3(d). Fourth, we observed that the bounds for the *t*-test are tighter than those of CI, but they make the false assumption that importance weighted returns are normally distributed. See Figures 3(b) and 3(e). Finally, we observed that the bounds for BCa seem to give slightly higher confidences than the *t*-test approach for same performance. These bounds do not make a Gaussian assumption, but still make the false assumption that the distribution of future empirical returns will be the same as what has been observed in the past, see Figures 3(c) and 3(f).

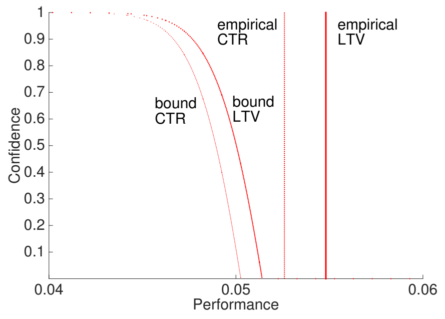


Figure 2: This figure shows the bounds and empirical importance weighted returns for the random strategy. It shows that every strategy has both a CTR and LTV metric.

7. SUMMARY AND CONCLUSIONS

In this paper we described an approach for successfully training and evaluating personal ad recommendation strategies. We used off policy evaluation frameworks with statistical guarantees to help us test the performance of the policies but to also optimize the algorithm parameters. We used the conservative CI bound that makes no assumption about the form of the distribution of returns, but also approximate bounds that make a false assumption. We expect the CI bound to get better as we include more data for evaluation. However, the approximate bounds seem to give clear ranking of the different strategies.

Overall in this paper we make multiple contributions. First, unlike most existing work on recommendation systems, we tackled the problem of life-time value recommendations and show how this approach gives us the best results. We were able to solve a real world problem efficiently with a relatively small campaign. Second, we identified the relationship between CTR and LTV and demonstrated with examples and experiments why CTR is not a good measure for the performance of LTV systems. Third, we are the first to apply LTV metrics and bounds to real world data. And fourth, we combined state of the art ingredients such as off-policy evaluations, the power of of random-forest regression and aggressive feature election to devise efficient optimization algorithms both for CTR and LTV.

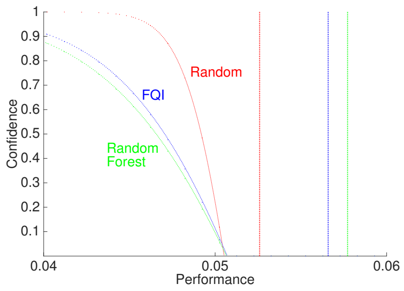
8. REFERENCES

[1] L. Breiman. Random forests. *Mach. Learn.*, 45(1):5–32, Oct. 2001.

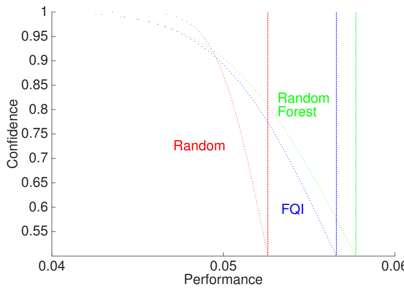
[2] J. Carpenter and J. Bithell. Bootstrap confidence intervals: when, which, what? a practical guide for medical statisticians. *Statistics in Medicine*, 19:1141–1164, 2000.

[3] L. Champless, A. Folsom, A. Sharrett, P. Sorlie, D. Couper, M. Szklo, and F. Nieto. Coronary heard disease risk prediction in the Atherosclerosis Risk in Communities (ARIC) study. *Journal of Clinical Epidemiology*, 56(9):880–890, 2003.

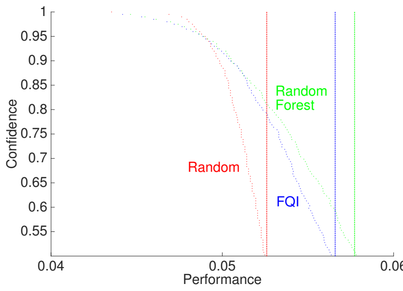
[4] B. Efron. Better bootstrap confidence intervals. *Journal of the American Statistical Association*, 82(397):171–185, 1987.



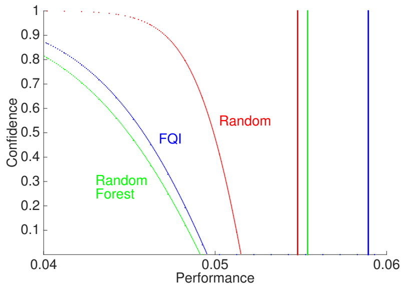
(a) CTR bounds for CI



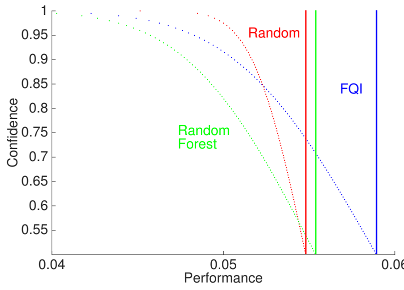
(b) CTR bounds for TT



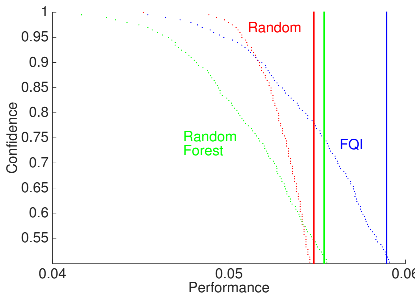
(c) CTR bounds for BCa



(d) LTV bounds for CI



(e) LTV bounds for TT



(f) LTV bounds for BCa

Figure 3: The Figures show the bounds according to the 3 methods for CTR and LTV. They also show the mean importance weighed returns.

[5] D. Ernst, P. Geurts, and L. Wehenkel. Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research*, 6:503–556, 2005.

[6] A. Folsom, L. Chambless, B. Duncan, A. Gilbert, and J. Pankow. Prediction of coronary heart disease in middle-aged adults with diabetes. *Diabetes Care*, 26(10):2777–2784, 2003.

[7] J. Jonker, N. Piersma, and D. V. den Poel. Joint optimization of customer segmentation and marketing policy to maximize long-term profitability. *Expert Systems with Applications*, 27(2):159 – 168, 2004.

[8] J. Langford, L. Li, and M. Dudík. Doubly robust policy evaluation and learning. In *Proceedings of the 28th International Conference on Machine Learning*, pages 1097–1104, 2011.

[9] L. Li, W. Chu, J. Langford, and R. Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th International Conference on World Wide Web*, pages 661–670, 2010.

[10] A. Maurer and M. Pontil. Empirical Bernstein bounds and sample variance penalization. In *Proceedings of the Twenty-Second Annual Conference on Learning Theory*, pages 115–124, 2009.

[11] E. Pednault, N. Abe, and B. Zadrozny. Sequential cost-sensitive decision making with reinforcement learning. In *Proceedings of the eighth international conference on Knowledge discovery and data mining*, pages 259–268, 2002.

[12] P. Pfeifer and R. Carraway. Modeling customer relationships as markov chains. *Journal of interactive marketing*, pages 43–55, 2000.

[13] D. Precup, R. S. Sutton, and S. Singh. Eligibility traces for off-policy policy evaluation. In *Proceedings of the 17th International Conference on Machine Learning*, pages 759–766, 2000.

[14] D. Silver, L. Newnham, D. Barker, S. Weller, and J. McFall. Concurrent reinforcement learning from customer interactions. In *In 30th International Conference on Machine Learning*, 2013.

[15] A. Strehl, J. Langford, L. Li, and S. Kakade. Learning from logged implicit exploration data. In *Proceedings of Neural Information Processing Systems 24*, pages 2217–2225, 2010.

[16] R. Sutton and A. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, 1998.

[17] G. Theocharous and A. Hallak. Lifetime value marketing using reinforcement learning. In *The 1st Multidisciplinary Conference on Reinforcement Learning and Decision Making*, 2013.

[18] P. Thomas, G. Theocharous, and M. Ghavamzadeh. High confidence off-policy evaluation. In *AAAI*, 2015.

[19] G. Tirenni, A. Labbi, C. Berrospi, A. Elisseeff, T. Bhose, K. Pauro, and S. Poyhonen. The 2005 ISMS Practice Prize Winner Customer-Equity and Lifetime Management (CELM) Finnair Case Study. *Marketing Science*, 26:553–565, 2007.

[20] W. Venables and B. Ripley. *Modern Applied Statistics with S*. Springer, New York, fourth edition, 2002.