

# Off-Policy Evaluation and Counterfactual Methods in Dynamic Auction Environments

Ritam Guha\* and Nilavra Pathak†

\*Department of Computer Science and Engineering, Michigan State University, East Lansing, MI

† Marketing Data Science, Expedia Group, New York, NY

guharita@msu.edu, npathak@expediagroup.com

**Abstract**—Counterfactual estimators are critical for learning and refining policies using logged data, a process known as Off-Policy Evaluation (OPE). OPE allows researchers to assess new policies without costly experiments, speeding up the evaluation process. Online experimental methods, such as A/B tests, are effective but often slow, thus delaying the policy selection and optimization process.

In this work, we explore the application of OPE methods in the context of resource allocation in dynamic auction environments. Given the competitive nature of environments where rapid decision-making is crucial for gaining a competitive edge, the ability to quickly and accurately assess algorithmic performance is essential. By utilizing counterfactual estimators as a preliminary step before conducting A/B tests, we aim to streamline the evaluation process, reduce the time and resources required for experimentation, and enhance confidence in the chosen policies. Our investigation focuses on the feasibility and effectiveness of using these estimators to predict the outcomes of potential resource allocation strategies, evaluate their performance, and facilitate more informed decision-making in policy selection. Motivated by the outcomes of our initial study, we envision an advanced analytics system designed to seamlessly and dynamically assess new resource allocation strategies and policies.

**Index Terms**—Off-Policy Evaluation, A/B Tests, Counterfactual Learning, Dynamic Auctions.

## I. INTRODUCTION

Modern-day marketplaces are increasingly designed as two-sided platforms. In such marketplaces, the goal is to maximize the utility on both the buyers and the sellers. This in turn enhances the overall revenue generated within the marketplace, and is critical in ensuring the long-term viability and competitiveness of the platform. Marketplaces typically implement complex auction mechanisms to achieve these goals. Unlike traditional auctions, that focus solely on price-based allocation, modern auctions consider a variety of factors. It includes the strategic behaviors of both buyers and sellers, the dynamic nature of their payment policies, and the overall impact on marketplace revenue [1], [2]. Key industrial applications of these advanced auction mechanisms

include Energy Markets, Telecommunications, Digital Marketing and many others.

Given the complexity of these auctions, payment policies have also become increasingly dynamic [2]. Validating a new payment policy in the real-world settings presents significant challenges. A/B testing is commonly used for online evaluation of the new policies, which is both expensive and time-consuming. Moreover, in user-facing commercial applications, A/B tests can risk exposing users to poor-performing policies, which can directly harm the application's revenue. As an alternative, simulation-based learning has been proposed as a new direction for evaluation [3]. However, these systems are still in their infancy, often relying on a deep understanding of the auction environment or on assumptions that may not hold true in real-world scenarios. While offline validation has its drawbacks, particularly the necessity of logged data, it has demonstrated significant advantages in applications such as recommendation and ranking. This method, known as Off-Policy Evaluation (OPE) [4], is employed in reinforcement learning and decision-making contexts to estimate the performance of a new policy using historical data collected under a different policy. Essentially, OPE predicts how well a new or hypothetical strategy would perform based on data generated by an existing strategy. In industrial settings, particularly in recommendation and ranking systems, OPE provides a practical and efficient alternative to traditional online evaluation methods.

In this paper, we explore the limitations and potential of using OPE in a dynamic auction environment. Our goal is to develop a measurement system capable of evaluating the payment policies using existing data alone, thereby enabling us to assess the feasibility of improving performance by testing new policies. To evaluate the viability of this approach, our study addresses the following key research questions:

- Conduct a comprehensive analysis of discrete ver-

sus continuous policy evaluation methods for dynamic competitive auctions.

- Assess the feasibility of identifying the best-performing policy among three options based on the outcomes of two prior tests.
- Learning optimal policies through optimization of policy estimators and evaluating their effectiveness via simulation.

## II. OVERVIEW OF THE APPROACH

### A. Problem Description

In this dynamic auction environment, resources are considered to be theoretically infinite, and an indeterminate number of agents compete to acquire them. Each agent assigns a personal value to the resources and aims to maximize their utility through participation. Agents operate according to a payment policy ( $\pi$ ), which influences their willingness to pay to acquire the resources for which they are competing.

The determination of the winning agent involves considering both the payment amount and the potential revenue generated from the allocation [2]. Winning the auction incurs costs for the agent, related to the resources expended. The effectiveness of the allocation is evaluated through metrics such as *reach*, *resources*, and *returns*. *Reach* measures the impact or extent of the allocation, this is analogous to views in marketing. *Resources* denote the quantity of units allocated after conversion, such as purchases, download, etc. *Returns* reflect the overall utility of winning the allocation.

Agents have the flexibility to design their payment policies to optimize their performance based on these metrics. This allows them to balance the costs incurred with the expected benefits derived from the allocated resources.

### B. Evaluation Pipeline

We adopt our pipeline based on the Open-Bandit Modules [4] but adapt it for our use-cases. The only controllable units in this case are the payment policies. An overview of the pipeline is provided in the Figure 1.

The pipeline consists of the following components:

- **Action Space:** In our problem, the action space consists of payment policies, which are continuous variables. For non-continuous evaluators, we discretize them by grouping them into bins.
- **Policy Modeling:** Instead of directly using the original payment policy, we identified proxies by training Random Forest Classifiers (for discrete action space) / Regressors (for continuous action

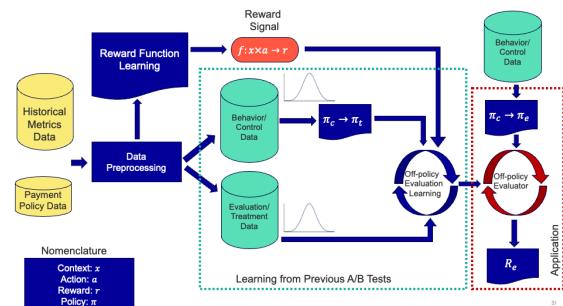


Figure 1: Off-Policy Evaluator Hyperparameter Learning and Application.

space) on the collected data. This approach is both generalizable and data-driven, making it applicable to any test scenario, regardless of the underlying model.

- **Reward Modeling:** A reward model is required for some of the metrics outlined in Section II-C. In these cases, we employed a Random Forest Regressor to model the reward.
- **Policy Evaluation:** Finally, we compare the rewards predicted by the OPE methods with the actual rewards observed during the A/B tests to assess the accuracy of the estimators. Our experiments involve comparing a wide range of discrete and continuous evaluators. The evaluators used in these comparisons are listed in Section II-C. The OPEs consist of some tunable hyperparameters which are learned to correct the distribution mismatch between the pre-tested policy and the new candidate policy. The data collected during the past A/B tests is used for this task.
- **Off-Policy Learning:** Off-policy learning is the approach where an agent learns about an optimal policy based on the off-policy evaluator. Due to the dynamic nature of the policies and the complex underlying probability associated with them, we learn proxy payment models for simulation. We elaborate more about the process in Section II-D.

### C. Off-Policy Evaluation

The main idea behind using OPE in dynamic environments is to hypothetically test the performance of different payment policies offline before selecting one for the final A/B test. This additional evaluation allows us to identify the most promising payment policies from various options, thereby reducing the need for multiple

A/B tests and lowering the chances of inconclusive results.

Various OPEs have been developed to enhance the accuracy of evaluations and manage noise in the data. OPEs can be categorized into two types based on the nature of the policy output: discrete and continuous. Discrete OPEs use the context-action distributions of the policies to generate importance weights for the data, which are then used to produce the final estimates. In contrast, continuous OPEs employ kernel density estimation to translate differences between policy actions into sample weights. Below, we provide a brief description of some of the evaluators considered in our work. A detailed description is available in the appendix.

- **Inverse Probability Weighting (IPW): [5], [6], [7]** Re-weights observed rewards by the ratio of the evaluation policy to the behavior policy, providing unbiased estimates but often with high variance.
- **Self-Normalized Inverse Probability Weighting (SNIPW): [8], [9]** Normalizes rewards using self-normalized importance weights, trading off unbiasedness for increased stability.
- **Direct Method (DM): [6], [10]** Estimates the policy value using a model of the expected rewards based on observed data, which depends heavily on the accuracy of the reward model.
- **Doubly Robust (DR): [6], [11], [12], [7]** Combines IPW with a reward model to reduce variance, providing consistent estimates if either the importance weights or the reward model are accurate.
- **Self-Normalized Doubly Robust (SNDR): [6], [9]** Applies self-normalized importance weighting within the DR framework to improve stability while maintaining double robustness.
- **Continuous Evaluators: [13]** Use kernel density estimation to model continuous treatment spaces, allowing for more precise OPE by avoiding information loss due to binning.

1) *Logged Data Collection:* We motivate our approach by using three prior conducted A/B tests. We consider three A/B tests as listed below:

A/B Test	Control Policy	Treatment Policy
Test-1	X	Y
Test-2	X	Z
Test-3	Y	Z

Table I: Comparison of Policies in Two A/B Tests

In Table I, we compare policies **X** and **Y** in *Test-1*, and **X** and **Z** in *Test-2*. Thus a major research question is – “Given the data from *Test 1* and *Test 2*, can we compare

**Y vs Z directly without conducting an actual A/B test?**”. To answer this question, we evaluate OPE approaches. We use the results from the *Test-3* as a ground truth for this comparison where we conducted an actual test between policies **Y** and **Z**.

The test measures are quantified by *Lifts*, which are measured between the metrics in the treatment group and the control group<sup>1</sup>:

$$\text{Lift (\%)} = \left( \frac{\text{Metric}_{\text{Treatment}} - \text{Metric}_{\text{Control}}}{\text{Metric}_{\text{Control}}} \right) \times 100$$

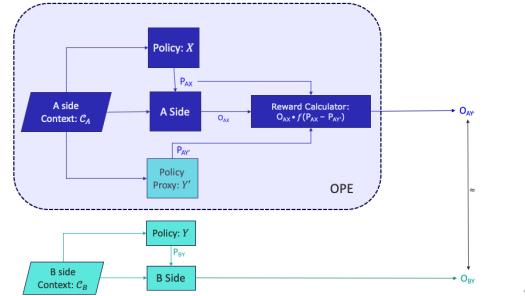


Figure 2: Off-Policy Evaluation Process Flow

The flowchart in Fig 2 illustrates the OPE process, where each test has two sides: Control (*A-side*) and Treatment (*B-side*). On the *A-side*, context  $C_A$  is input to Policy  $X$ , generating an reward outcome  $O_{AX}$  and predicted action  $P_{AX}$ . A reward calculator then adjusts the reward outcome to  $O_{AY'}$  using a function  $f$  that compares  $P_{AX}$  with the action predicted by proxy policy  $Y'$ ,  $P_{AY'}$ . On the *B-side*, context  $C_B$  is used by Policy  $Y$ , resulting in a reward outcome  $O_{BY}$  and predicted action  $P_{BY}$ . The goal is to verify that  $O_{AY'}$ , the OPE-derived reward outcome on *A-side* using  $Y'$ , approximates  $O_{BY}$ , the observed reward outcome on *B-side*, assuming equivalent contexts  $C_A$  and  $C_B$ . This comparison validates the OPE method’s accuracy in predicting how Policy  $Y$  would perform on *A-side*.

2) *Learning Policy Proxies:* Instead of using the policies directly for the evaluation, we learned proxies for the policies using random forest regression/classification models. Basically, we learned mappings from context space ( $\mathcal{C}$ ) to the action space ( $\mathcal{A}$ ) as:  $\mathbf{X}', \mathbf{Y}', \mathbf{Z}'$  ( $\pi : \mathcal{C} \rightarrow \mathcal{A}$ ). In case of discrete OPEs,  $\mathcal{A}$  becomes a discrete space, while for continuous OPEs, it becomes a continuous space. For discrete case, a random forest classifier serves as the mapping structure, while a regres-

<sup>1</sup>To ensure that the observed lift is not due to random chance, we perform a statistical significance test (T-Test) to get confidence intervals of the measured lifts.

sor is considered for the continuous case. For discrete scenarios, the continuous action space is divided into multiple bins and the actions are mapped to a specific bin. The bin numbers serve as the label for the data to be learned for the mapping.

---

**Algorithm 1** Optimal Payment Policy Learning (OptPaL) through OPE

---

**Require:** Context data  $\mathcal{C}$ , Off-policy eval OPE, Learning rate  $\alpha$ , Max iterations  $n_{max}$

- 1: Initialize OptPaL as an Multi-layer Perceptron (MLP) model with random weights  $W$
- 2: **for**  $i = 1$  to  $n_{max}$  **do**
- 3:   Predict payment using OptPaL:  $\mathbf{P} \leftarrow \text{OptPaL}(\mathcal{C})$
- 4:   Evaluate  $\mathbf{P}$ :  $(Cost, Returns) \leftarrow \text{OPE}(\mathcal{C}, \mathbf{P})$
- 5:   Compute loss:  $\mathcal{L} \leftarrow f(Cost, Returns)$
- 6:   Update weights:  $W \leftarrow W - \alpha \cdot \nabla \mathcal{L}$
- 7:   **if** Converged( $\mathcal{L}$ ) **then**
- 8:     **break**
- 9:   **end if**
- 10: **end for**
- 11: **return** OptPaL

---

#### D. New Policy Learning

Off-policy learning goes beyond mere evaluation by leveraging OPE results to iteratively refine and enhance policies. The primary objective of the new policy learning process is to identify a counterfactually optimal policy that maximizes expected rewards within the given environment for the past logged data. A key advantage of continuous off-policy evaluation is its differentiability, which facilitates the application of gradient descent methods to fit models and learn optimal policies. Following hyperparameter optimization, for continuous evaluators, we obtain individual OPEs for each of the outcome metrics under consideration.

Once the OPEs are established, we can employ gradient descent search on an underlying machine learning model to find the optimal payment policy based on a loss function involving *Cost* and *Returns*. The purpose of this exercise is to evaluate, hypothetically, how different payment strategies could have improved outcome metrics in the past. If we can develop a reliable proxy model for determining optimal payment policy, this model can be applied to inform future payment strategies or create the best-policy in the hindsight for comparison. We provide a description of the optimal policy learning using the off-policy evaluators in Algorithm 1.

### III. EXPERIMENTAL EVALUATION

In our initial study, we pose two different research questions:

- Given two separate A/B Tests, conducted under different conditions, can we determine the best policy?

We consider two prior A/B tests as the basis of our evaluation which we call *Test-1* and *Test-2*. The results, as evaluated in the two tests, are provided in Fig 3. At first glance, it seems that the **Policy Z** had the better lift, but *Test-1* and *Test-2* are conducted under separate conditions. Thus it is difficult to say whether **Z** is truly better than **Y**. Thus our initial goal is to simulate under similar conditions which one is better in a simulated scenario of **Y** vs **Z**.

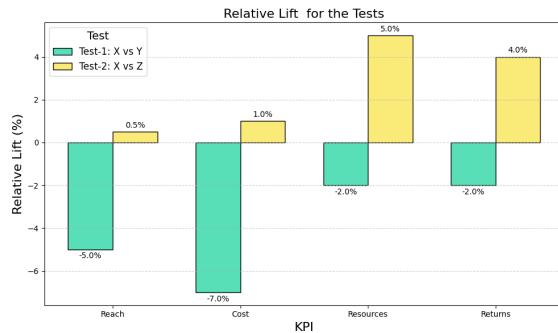


Figure 3: Relative Lifts for the two past A/B tests (*Test-1* and *Test-2*).

- Can we learn an optimal policy from the data itself ?

After assessing the suitability of OPE, we sought to explore counterfactual scenarios to determine what the optimal payment policy might have been in the past. Based on this analysis, we aim to learn an optimal policy by optimizing over the evaluation metrics.

#### A. Comparing Discrete vs Continuous case for payment

We need to compare the discretized action space and continuous action space to check how much accuracy we are gaining in terms of different metrics for moving to the continuous version. Both approaches are compared with respect to the Mean Absolute Percentage Error (MAPE) [14] for different metrics in the *Test-1* and *Test-2* experiments. The MAPEs aggregated over the OPE estimators for both tests are shown in Figure 4.

**Observation– Continuous OPE tends to outperform discretized OPE.** This is intuitive because continuous OPE can capture the smooth, nuanced dependencies of the key metrics across small changes in payment values, providing a more detailed and accurate evaluation of policies. In contrast, discretized OPE may miss these

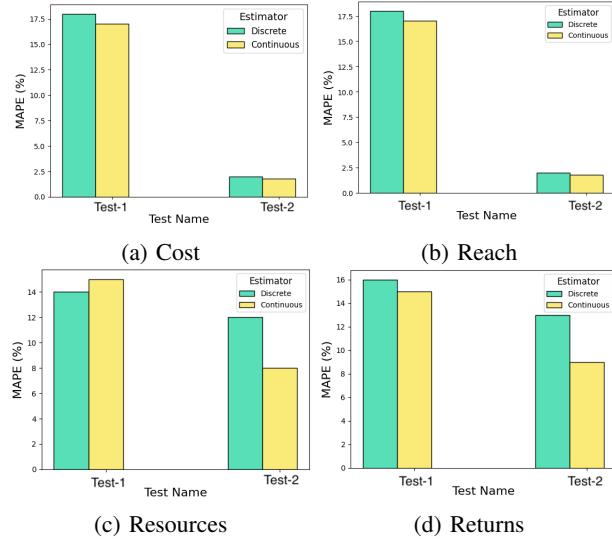


Figure 4: MAPE Estimate: Discretized Action vs Continuous Action Evaluation

*fine-grained variations due to its coarser resolution. Overall, we get around 20% reduction in MAPE by moving from the discretized version to the continuous version.*

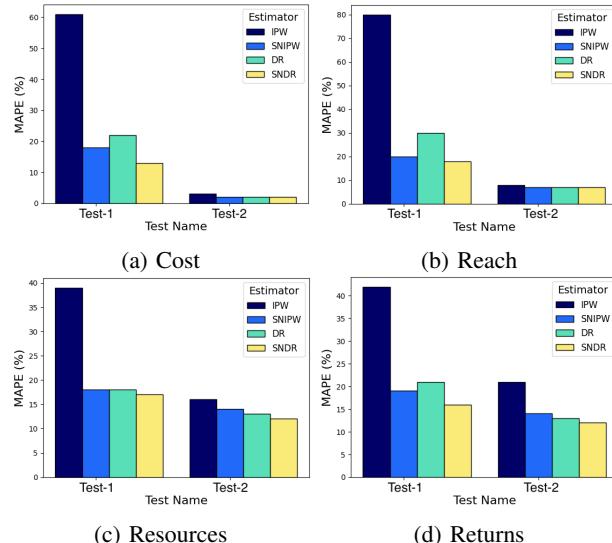


Figure 5: MAPE Estimate: Evaluation of Different Continuous Evaluators

### B. Comparison of Continuous Version of the Estimators

After verifying that continuous evaluators work better than the discrete evaluators, we evaluate the continuous

version of the estimators mentioned in Section II-C. In Figure 5, we show the results of the top four evaluators for both tests.

**Observation- SNDR works the best in our case.** The optimal kernel and bandwidth for the estimator can be determined through a separate hyperparameter tuning experiment. As more data is aggregated and used to predict overall rewards, the uncertainty in these predictions diminishes. For instance, predicting the total number of resources acquired over a week is more reliable than predicting at a daily grain. This reduction in uncertainty with larger aggregated data highlights the importance of selecting appropriate hyperparameters to achieve accurate predictions.

### C. Counterfactual Test of Policies: Y vs Z

1) *Use Case I: Assessing the Accuracy of OPE:* To assess the accuracy of the learning proxies and the OPE, we conducted an evaluation — the *Test-3*, comparing policies **Y** and **Z**. Following this, we constructed a **Proxy Policy Y** based on the past data collected for Policy **Y**. We then replaced **Proxy Policy Y** with Policy **X** in *Test-2* which we refer to as *Counterfactual Test-2* and compared the outcomes with the original *Test-3* results, as illustrated in Figure 6.

**Takeaways – Directional Lifts are Correlated:** For the mentioned metrics we were able to achieve similar directional lifts which shows that the OPEs were able to estimate the proper directional lifts for **Policy Y** through **Proxy Policy Y**. Improving the learning algorithms can potentially lead to accurate reward prediction and thus effective policy evaluation.

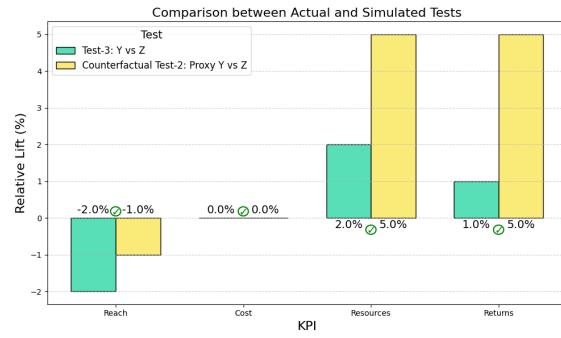


Figure 6: Relative Lifts for the Actual (**Y vs Z**) and Simulated (**Proxy Y vs Z**) Tests

2) *Use Case II: Estimating the Impact of Policy Z in Test-1 in Place of Policy X, and Comparison with Counterfactual Test-2:* After verifying that the OPEs are able to accurately predict the directional lifts for a

counterfactual test, our next objective is to utilize the counterfactual evaluation process to compare different payment policies in different contexts without doing actual A/B tests. In the *Test-1*, **Policy Y** showed negative lifts in *reach*, *resources*, and *returns*, while in the *Test-2*, **Policy Z** demonstrated significant lifts in *resources* acquired and the subsequent *returns*. These results suggest that **Policy Z** is likely to outperform **Policy Y** in a direct comparison.

To create a hypothetical evaluation, named as *Counterfactual Test-1*, we simulated a scenario where a proxy version of **Policy Z** (**Z**) replaced **Policy X** as the control in the *Test-1*. The treatment was set to **Policy Y**. The resulting lifts for *Counterfactual Test-1* and *Counterfactual Test-2* scenarios are compared in Fig 7. In *Counterfactual Test-1*, the lifts indicate that **Proxy Policy Z** outperforms **Policy Y** across all KPIs. In *Counterfactual Test-2*, although **Policy Z** shows a slight negative lift in *Reach*, this is offset by substantial positive lifts in *Resources* and *Returns*.

**Insights – Robustness of Policy Z:** The consistency in performance of **Policy Z** across both tests suggests that it is generally a more robust choice for improving key metrics.

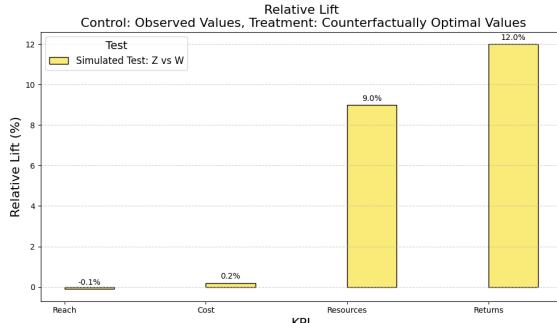


Figure 7: Relative Lifts for two Counterfactual Tests

#### D. New Policy Learning through OPE Optimization

Finally, we conducted new model learning to identify the optimal policy in retrospect. For our use case, we selected profit maximization as the objective function. As a standard practice in machine learning, the negative of the profit maximization is used as the loss function, defined as:

$$\min_p f = -(\text{Returns}_{\text{OPE}}(p) - \text{Cost}_{\text{OPE}}(p)),$$

where  $p = \text{OptPaL}(\mathcal{C})$

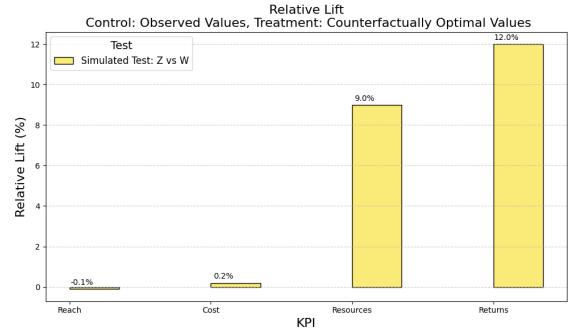


Figure 8: Relative Lifts for the Actual and Simulated Tests

Here, the returns and cost are estimated using the OPE models. We employed a multi-layer perception (MLP) named **OptPaL** with two hidden layers having  $\frac{|\mathcal{C}|}{2}$  and  $\frac{|\mathcal{C}|}{4}$  nodes, respectively. The input to **OptPaL** is the context  $\mathcal{C}$ , while the output is the payment value  $p$ . By optimizing **OptPaL** through gradient descent, we achieved a non-diminishing objective value after training the counterfactual model for 1000 epochs. To compare the relative performance of the new optimal policy (**Policy W**:  $\text{OptPaL}(\mathcal{C})$ ) with the existing **Policy X**, we utilized the *Test-2*.

The results of the simulated test are shown in Figure 8. The optimal policy **W** was able to allocate the same expenditure as **Policy X** while maintaining comparable *Reach* and *Cost*, but it increased the overall *Resource* count and total *Return*. This demonstrates the efficacy of **Policy W** and provides insights into the predictive uncertainty by showing how much error was made due to the limitations in our predictions.

**Takeaways:** Counterfactual policy learning helps us discover new optimal policies and can be used for baseline evaluations of new policies.

## IV. CONCLUSION

In this study, we have demonstrated the effectiveness of off-policy evaluation (OPE) techniques in refining payment strategies by leveraging counterfactual scenarios. Our results underscore the advantages of continuous OPE methods over traditional discretized approaches, providing more precise and nuanced insights into policy performance. Through counterfactual policy evaluation, we are able to estimate directional lifts for new policies, offering a preliminary assessment of their utility prior to conducting A/B tests. Furthermore, we developed optimal policies by optimizing the continuous estima-

tor, thereby identifying the most effective strategies in retrospect.

## V. FUTURE VISION

The future of policy evaluation envisions a fully automated system designed to seamlessly and dynamically assess new payment strategies and policies. This system will integrate scalable engineering and advanced methodologies to deliver continuous, accurate, and actionable insights. We provide a description of the system design of the Advanced Analytics Platform in the Figure 9.

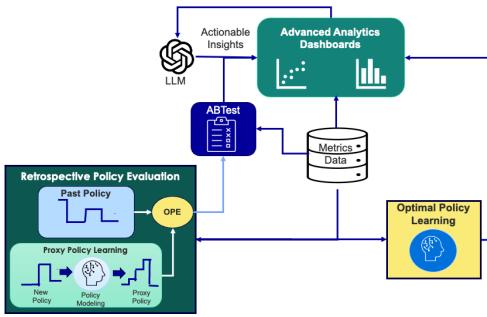


Figure 9: Advanced Analytics Platform

The foundational components of the system will include:

- **Retrospective Policy Evaluation:** This component focuses on learning the optimal policy in hindsight, allowing us to determine the best course of action based on past data. It involves simulating outcomes of new algorithmic policies, offering foresight into potential impacts and enabling the evaluation of policies before deployment. This subsystem will help decide whether a new proposed policy should undergo A/B testing.
- **Optimal Policy Learning:** Aims to generate an optimal policy from historical data. This will help us compare and analyze the regret of past actions.
- **Comprehensive Monitoring and Insights:** The system will provide intuitive and actionable insights into policy performance, trends, and recommendations, enabling decision-makers to understand and respond effectively. Integrating Large-Language Models will further refine the learning process and enhance actionable insights.

Our next research direction will focus on enhancing proxy policy learning and exploring Structural Causal Model-Based Counterfactual Evaluation [15] to improve the accuracy of policy assessments. Additionally, we will

consider large dimensional action spaces [16] to better account for complex auction environments.

## REFERENCES

- [1] Paul Milgrom. Auction research evolving: Theorems and market designs. *American Economic Review*, 111(5):1383–1405, 2021.
- [2] Guillaume Haerlinger. *Market design: auctions and matching*. MIT Press, 2018.
- [3] Olivier Jeunen, Sean Murphy, and Ben Allison. Learning to bid with auctiongym. *AdKDD*, 2022.
- [4] Yuta Saito, Shunsuke Aihara, Megumi Matsutani, and Yusuke Narita. Open bandit dataset and pipeline: Towards realistic and reproducible off-policy evaluation. *arXiv preprint arXiv:2008.07146*, 2020.
- [5] Alex Strehl, John Langford, Lihong Li, and Sham M Kakade. Learning from logged implicit exploration data. *Advances in neural information processing systems*, 23, 2010.
- [6] Miroslav Dudik, Dumitru Erhan, John Langford, and Lihong Li. Doubly robust policy evaluation and optimization. *Statistical Science*, 29(4):485–511, 2014.
- [7] Yi Su, Maria Dimakopoulou, Akshay Krishnamurthy, and Miroslav Dudik. Doubly robust off-policy evaluation with shrinkage. In *International Conference on Machine Learning*, pages 9167–9176. PMLR, 2020.
- [8] Adith Swaminathan and Thorsten Joachims. The self-normalized estimator for counterfactual learning. *advances in neural information processing systems*, 28, 2015.
- [9] Nathan Kallus and Masatoshi Uehara. Intrinsically efficient, stable, and bounded off-policy evaluation for reinforcement learning. *Advances in neural information processing systems*, 32, 2019.
- [10] Alina Beygelzimer and John Langford. The offset tree for learning with partial labels. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 129–138, 2009.
- [11] Mehrdad Farajtabar, Yinlam Chow, and Mohammad Ghavamzadeh. More robust doubly robust off-policy evaluation. In *International Conference on Machine Learning*, pages 1447–1456. PMLR, 2018.
- [12] Yi Su, Lequn Wang, Michele Santacatterina, and Thorsten Joachims. Cab: Continuous adaptive blending for policy evaluation and learning. In *International Conference on Machine Learning*, pages 6005–6014. PMLR, 2019.
- [13] Nathan Kallus and Angela Zhou. Policy evaluation and optimization with continuous treatments. In *International conference on artificial intelligence and statistics*, pages 1243–1251. PMLR, 2018.
- [14] Ritam Guha, Anirudh Suresh, Jared DeFrain, and Kalyanmoy Deb. Virtual metrology in long batch processes using machine learning. *Materials and Manufacturing Processes*, 38(15):1997–2008, 2023.
- [15] Alexander Balke and Judea Pearl. Probabilistic evaluation of counterfactual queries. *Probabilistic and Causal Inference: The Works of Judea Pearl*, page 237, 2011.
- [16] Yuta Saito and Thorsten Joachims. Off-policy evaluation for large action spaces via embeddings. In *International Conference on Machine Learning (ICML)*, 2022.
- [17] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2019.

## APPENDIX

### A. Evaluators Detailed Description

1) *Inverse Probability Weighting (IPW)*: IPW estimates the policy value of evaluation policy( $\pi_e$ ) [5], [6], [7]:

$$\hat{V}_{\text{IPW}}(\pi_e; \mathcal{D}) := \mathbb{E}_n[w(x_i, a_i) \cdot r_i] \quad (1)$$

where  $\mathcal{D} = \{(x_i, a_i, r_i)\}_{i=1}^n$  is logged bandit data with  $n$  observations collected by behavior policy  $\pi_b$ ;  $w(x, a) := \pi_e(a|x)/\pi_b(a|x)$  is the importance weight given  $x$  and  $a$ .  $\mathbb{E}_n[\cdot]$  is the empirical average over  $n$  observations in  $D$ . When the clipping is applied, a large importance weight is clipped as  $\hat{w}(x, a) := \min\{\lambda, w(x, a)\}$  where  $\lambda(> 0)$  is a hyperparameter to specify a maximum allowed importance weight. IPW re-weights the rewards by the ratio of the evaluation policy and behavior policy (importance weight). When the behavior policy is known, IPW is unbiased and consistent for the true policy value. However, it can have a large variance, especially when the evaluation policy significantly deviates from the behavior policy.

2) *Self-Normalized Inverse Probability Weighting (SNIPW)*: SNIPW estimates the policy value of evaluation policy ( $\pi_e$ ) [8], [9] as

$$\hat{V}_{\text{SNIPW}}(\pi_e; \mathcal{D}) := \frac{\mathbb{E}_n[w(x_i, a_i) \cdot r_i]}{\mathbb{E}_n[w(x_i, a_i)]} \quad (2)$$

SNIPW normalizes the observed rewards by the self-normalized importance weight. This estimator is not unbiased even when the behavior policy is known. However, it is still consistent for the true policy value and gains some stability in OPE.

3) *Direct Method (DM)*: DM [6], [10] first trains a supervised ML model, such as ridge regression and gradient boosting, to estimate the reward function  $q(x, a) = \mathbb{E}[r|x, a]$ . It then uses the estimated rewards to estimate the policy value as follows.

$$\begin{aligned} \hat{V}_{\text{DM}}(\pi_e; \mathcal{D}, \hat{q}) &:= \mathbb{E}_n \left[ \sum_{a \in \mathcal{A}} \hat{q}(x_i, a) \pi_e(a|x_i) \right] \\ &= \mathbb{E}_n[\hat{q}(x_i, \pi_e)] \end{aligned} \quad (3)$$

$\hat{q}(x, a)$  is the estimated expected reward given  $x$  and  $a$ .  $\hat{q}(x_i, \pi) := \mathbb{E}_{a \sim \pi(a|x)}[\hat{q}(x, a)]$  is the expectation of the estimated reward function over  $\pi$ . If the regression model ( $\hat{q}$ ) is a good approximation to the true mean reward function, this estimator accurately estimates the policy value of the evaluation policy. If the regression function fails to approximate the reward function well, however, the final estimator is no longer consistent.

4) *Doubly Robust (DR)*: Similar to DM, DR [6], [11], [12], [7], estimates the reward function ( $q(x, a) = \mathbb{E}[r|x, a]$ ). It then uses the estimated rewards to estimate the policy value as follows.

$$\hat{V}_{\text{DR}}(\pi_e; \mathcal{D}, \hat{q}) := \mathbb{E}_n[\hat{q}(x_i, \pi_e) + w(x_i, a_i)(r_i - \hat{q}(x_i, a_i))] \quad (4)$$

When the clipping is applied, a large importance weight is clipped as  $\hat{w}(x, a) := \min\{\lambda, w(x, a)\}$  where  $\lambda(> 0)$  is a hyperparameter to specify a maximum allowed importance weight. DR mimics IPW to use a weighted version of rewards, but DR also uses the estimated mean reward function (the regression model) as a control variate to decrease the variance. It preserves the consistency of IPW if either the importance weight or the mean reward estimator is accurate (a property called double robustness). Moreover, DR is semi-parametric efficient when the mean reward estimator is correctly specified.

5) *Self-Normalized Doubly Robust (SNDR)*: Similar to SNIPW, the SNDR estimator applies the self-normalized importance weighting technique to gain some stability. The SNDR estimator computes the policy value of the evaluation policy  $\pi_e$  as

$$\hat{V}_{\text{SNDR}}(\pi_e; \mathcal{D}, \hat{q}) := \mathbb{E}_n \left[ \hat{q}(x_i, \pi_e) + \frac{w(x_i, a_i)(r_i - \hat{q}(x_i, a_i))}{\mathbb{E}_n[w(x_i, a_i)]} \right] \quad (5)$$

6) *Continuous Evaluators*: In traditional discretized off-policy evaluators, significant drawbacks arise due to the loss of information when continuous values are binned together. This binning process can obscure subtle differences within the data, making it challenging to evaluate small changes, as these nuances are often lost within the bins. Furthermore, discretized approaches rely on ad-hoc modeling of the policies, which may not accurately reflect the underlying continuous nature of the data.

In contrast, continuous off-policy evaluation (OPE) is more appropriate for contexts with continuous treatment spaces. To address these challenges, we employ Multivariate Kernel Density Estimation (KDE) to model the conditional probability density of the continuous variables (e.g., payment, denoted as  $b$ ) with respect to the context (denoted as  $x$ ):

$$P(T = b|X = x) = \frac{P(T = b, X = x)}{P(X = x)}$$

Once we model the probability densities, different kernels are applied to map the differences between

payments to importance weights. Essentially, the OPE strategy assigns greater weight to data samples that exhibit smaller differences in payment amounts relative to the currently evaluated sample. The estimated reward is calculated using the continuous evaluation metric [13], which is defined as:

$$\hat{v}_\tau = \frac{1}{nh} \sum_{i=1}^n K \left( \frac{\tau(x_i) - t_i}{h} \right) \frac{y_i}{Q_i} \quad (6)$$

In this equation:

- $n$  is the number of data samples,
- $h$  is the bandwidth,
- $K$  is the kernel function,
- $\tau(x_i)$  represents the action that would have been taken by the evaluation policy,
- $t_i$  represents the action taken by the behavioral policy,
- $Q_i = P(t_i|x_i)$  is the probability density of the action under the behavioral policy, and
- $y_i$  is the observed reward.

Two critical hyperparameters in this process are the kernel  $K$  and the bandwidth  $h$ . The bandwidth controls the scale of proximity considered between treatments: a bandwidth too large introduces high bias as it averages over a broader dataset, while a bandwidth too small increases variance by focusing too narrowly. Therefore, careful tuning of both the kernel and bandwidth is essential to optimize the OPE process for specific key metrics.

To perform hyperparameter tuning, we utilize Optuna [17], an efficient optimization framework, to individually optimize the kernel and bandwidth for each metrics.