A Large-scale Open Dataset for Bandit Algorithms

Yuta Saito
Tokyo Institute of Technology
saito.y.bj@m.titech.ac.jp

Shunsuke Aihara
ZOZO Technologies, Inc.
shunsuke.aihara@zozo.com

Megumi Matsutani
ZOZO Technologies, Inc.
megumi.matsutani@zozo.com

Yusuke Narita
Yale University
yusuke.narita@yale.edu

Abstract

We build and publicize the *Open Bandit Dataset and Pipeline* to facilitate scalable and reproducible research on bandit algorithms. They are especially suitable for *off-policy evaluation* (OPE), which attempts to predict the performance of hypothetical algorithms using data generated by a different algorithm in use. We construct the dataset based on experiments and implementations on a large-scale fashion e-commerce platform, ZOZOTOWN. The data contain the ground-truth about the performance of several bandit policies and enable the fair comparisons of different OPE estimators. We also provide a pipeline to make its implementation easy and consistent. As a proof of concept, we use the dataset and pipeline to implement and evaluate OPE estimators. First, we find that a well-established estimator fails, suggesting that it is critical to choose an appropriate estimator. We then select a well-performing estimator and use it to improve the platform's fashion item recommendation. Our analysis succeeds in finding a counterfactual policy that significantly outperforms the historical ones. Our open data and pipeline will allow researchers and practitioners to easily evaluate and compare their bandit algorithms and OPE estimators with others in a large, real-world setting. Our open data and pipeline are publicly available at `https://github.com/st-tech/zr-obp`.

1 Introduction

Interactive bandit and reinforcement learning systems (e.g. personalized medicine, ad/recommendation/search platforms) produce log data valuable for evaluating and redesigning the systems. For example, the logs of a news recommendation system record which news article was presented and whether the user read it, giving the system designer a chance to make its recommendation more relevant. Exploiting log data is, however, more difficult than conventional supervised machine learning: the result is only observed for the action chosen by the system but not for all the other actions the system could have taken. The logs are also biased in that the logs over-represent the actions favored by the system.

A potential solution to this problem is an A/B test that compares the performance of counterfactual systems in an online environment. However, A/B testing counterfactual systems is often difficult, since deploying a new policy is time- and money-consuming, and entails a risk of failure.

This leads us to the problem of *off-policy evaluation* (OPE), which aims to estimate the performance of a counterfactual policy using only log data collected by a past (or behavior) policy. Such an evaluation allows us to compare the performance of candidate counterfactual policies to decide which policy should be deployed. This alternative approach thus solves the above problem with the A/B test approach. Applications range from contextual bandits (Bottou et al., 2013; Kato et al.,

Preprint. Under review.

2020; Li et al., 2012, 2010, 2011; Narita et al., 2019; Strehl et al., 2010; Swaminathan & Joachims, 2015a,b; Swaminathan et al., 2017; Wang et al., 2017) and reinforcement learning in the web industry (Farajtabar et al., 2018; Irpan et al., 2019; Jiang & Li, 2016; Kallus & Uehara, 2019; Liu et al., 2018; Narita et al., 2020; Thomas & Brunskill, 2016; Thomas et al., 2015a,b; Xie et al., 2019) to other social domains such as healthcare (Murphy et al., 2001) and education (Mandel et al., 2014).

Issues with current experimental procedures. While the research community has produced theoretical breakthroughs, the experimental evaluation of OPE remains primitive. Specifically, it lacks a public benchmark dataset for comparing the performance of different methods. Researchers often validate their methods using synthetic simulation environments Kallus & Uehara (2019); Kato et al. (2020); Liu et al. (2018); Voloshin et al. (2019); Xie et al. (2019). A version of the synthetic approach is to modify multi-class classification datasets and treat supervised machine learning methods as bandit policies to evaluate off-policy estimators Dudík et al. (2014); Farajtabar et al. (2018); Vlassis et al. (2019); Wang et al. (2017). An obvious problem with these studies is that there is no guarantee that their simulation environment is similar to real-world settings. To solve this issue, Gilotte et al. (2018); Gruson et al. (2019); Narita et al. (2019, 2020) use proprietary real-world datasets. Since these datasets are not public, however, it remains challenging to reproduce the results, and compare their methods with new ideas in a fair manner. This is in contrast to other domains of machine learning, where large-scale open datasets, such as the ImageNet dataset (Deng et al., 2009), have been pivotal in driving objective progress (Dwivedi et al., 2020; Hu et al., 2020; Girshick et al., 2014; He et al., 2016; Long et al., 2015).

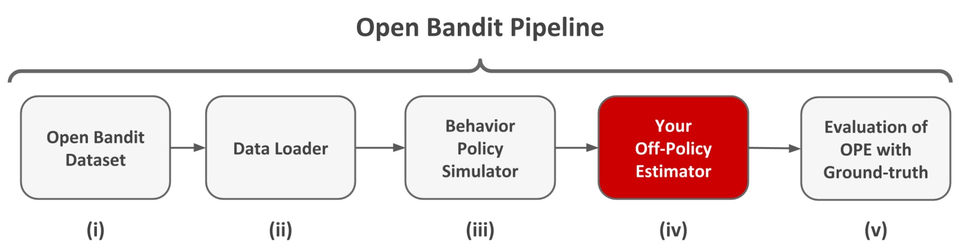

Figure 1: Overview of the *Open Bandit Dataset and Pipeline*

Notes: **(i)** Open Bandit Dataset provides a large-scale real-world setting. **(ii)** Data loader automates dataset preprocessing. **(iii)** Our pipeline includes implementation of behavior policies used in ZOZOTOWN production. **(iv)** Researchers can easily use their own OPE estimator. **(v)** Our data and pipeline enables the evaluation of OPE with the ground-truth.

Contributions. Our goal is to implement and evaluate OPE of bandit algorithms in realistic and reproducible ways. We release the *Open Bandit Dataset*, a logged bandit feedback collected on a large-scale fashion e-commerce platform, ZOZOTOWN.[1] ZOZOTOWN is the largest fashion EC platform in Japan with over 3 billion USD annual Gross Merchandise Value. When the platform produced the data, it used Bernoulli Thompson Sampling (Bernoulli TS) and Random policies to recommend fashion items to users. The dataset includes an A/B test of these policies and collected over 26 million records of users' clicks and the ground-truth about the performance of Bernoulli TS and Random. To streamline and standardize the analysis of the Open Bandit Dataset, we also provide the *Open Bandit Pipeline*, a series of implementations of dataset preprocessing, behavior bandit policy simulators, and OPE estimators. Figure 1 illustrates the overview of our open data and pipeline.

To illustrate how to use the dataset and pipeline, we combine them with state-of-the-art OPE estimators to improve the function of the platform. First, we select the best OPE method by comparing three different estimators. Specifically, for each estimator, we use the log data of one of the behavior policies to predict the click through rates (CTR) of the other policy. We then assess the accuracy of the prediction by comparing it with the ground truth contained in the data. We compare the three estimators by their prediction performance. This exercise shows the following:

> **Empirical Result 1.** Inverse Probability Weighting (IPW; (Strehl et al., 2010)) and Doubly Robust (DR; (Dudík et al., 2014)) w[illegible]predict performance of coun-

[1]https://corp.zozo.com/en/service/

terfactual policies (prediction errors being lower than 8.5%), while Direct Method (DM; (Beygelzimer & Langford, 2009)) produces much larger prediction errors.

This result is presented in Figure 2, where IPW and DR predict the ground-truth policy values of Bernoulli TS and Random well. In contrast, DM exhibits poor predictions. This experiment suggests that a well-established estimator like DM may fail to predict the performance of a counterfactual policy. It is therefore essential to select an appropriate method, such as IPW and DR in this case.

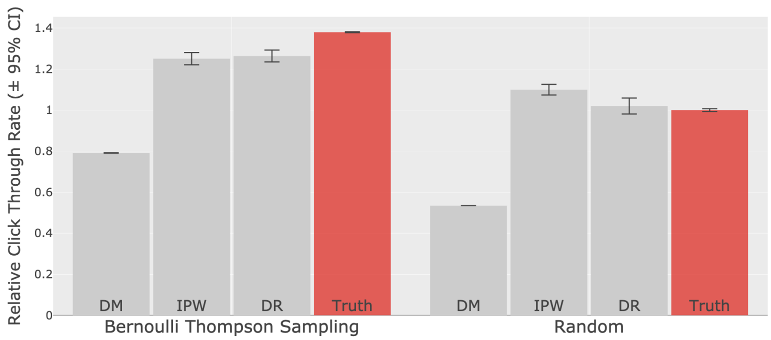

Figure 2: Comparing Off-Policy Evaluations with the Ground Truth

Notes: We report the estimated performances of Bernoulli Thompson Sampling and Random policies and their 95% confidence intervals (CI) in one of the three campaigns. IPW and DR predict the ground-truths well, while DM exhibits poor predictions.

Next, we perform a counterfactual policy search to find a new policy that would improve the CTR of the platform's fashion recommendation. The platform aims to improve the current context-free bandit algorithms (i.e., Bernoulli TS and Random) using their log data. For this purpose, we construct a counterfactual policy search space including 12 possible counterfactual policies consisting of different algorithms, hyperparameters, and context sets. We then evaluate their performance using the DR estimator, which performs best in the evaluation summarized in Empirical Result 1. This leads us to obtain the following bottomline:

> **Empirical Result 2.** The best hypothetical policy would statistically significantly improve the CTR by 56.7 % - 64.5 % compared to current best Bernoulli TS.

This result is reported in Figure 3, where the best hypothetical policy significantly outperforms Bernoulli TS in all campaigns. With an appropriate OPE technique, therefore, we provide the platform with valuable and reliable managerial input. Our pipeline implementation is flexible and can handle datasets other than our open data, thus allowing practitioners to follow our counterfactual policy search procedure and redesign their systems.

We believe that our open data and pipeline help researchers evaluate the empirical performance of their methods, thereby advancing ~~the~~ future OPE research. Our case study showcases how to use our data to compare different estimators and use an appropriate one to improve the bandit systems.

2 Setup

We consider a general multi-armed contextual bandit setting. Let $\mathcal{A} = \{0, ..., m\}$ be a finite set of $m + 1$ *actions* (equivalently, *arms* or *treatments*), that the decision maker can choose from. Let $Y(\cdot) : \mathcal{A} \to \mathbb{R}$ denote a potential reward function that maps actions into rewards or outcomes, where $Y(a)$ is the reward when action a is chosen (e.g., whether a fashion item as an action results in a click). Let X denote a *context* vector (e.g., the user's demographic profile and user-item interaction history) that the decision maker observes when picking an action. We denote the finite set of possible contexts by $\mathcal{X}$. We think of $(Y(\cdot), X)$ as a random vector with unknown distribution G. Given a vector of $(Y(\cdot), X)$, we define the mean reward function $\mu : \mathcal{X} \times \mathcal{A} \to \mathbb{R}$ as $\mu(x, a) = \mathbb{E}[Y(a)|X = x]$.

We call a function $\pi : \mathcal{X} \to \Delta(\mathcal{A})$ a *policy*, which maps each context $x \in \mathcal{X}$ into a distribution over actions, where $\pi(a|x)$ is the probability of taking action a given a context vector x. Let

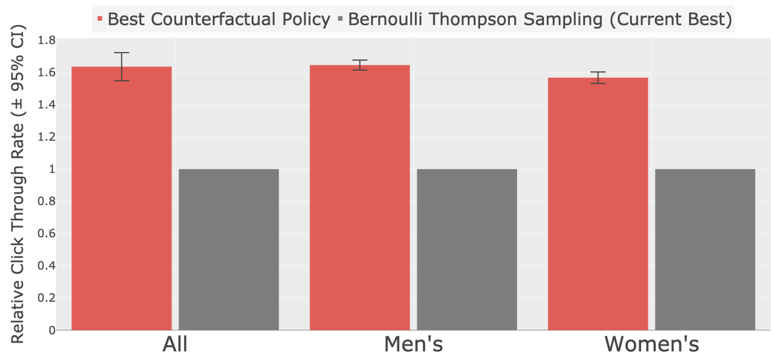

Figure 3: Comparing Counterfactual Policy and Bernoulli Thompson Sampling

Notes: This figure shows the best counterfactual policies among 12 candidate policies for three campaigns (All, Men's, and Women's). We report the performances relative to Bernoulli Thompson Sampling (the current best) and their 95% confidence intervals (CI). The result shows that the best policies significantly outperform the current best in all campaigns.

$\{(Y_t, X_t, D_t)\}_{t=1}^T$ be historical logged bandit feedback with T rounds of observations. $D_t := (D_{t0}, ..., D_{tm})'$, where D_{ta} is a binary variable indicating whether action a is chosen in round t. If a is chosen in round t, $D_{ta} = 1$, otherwise $D_{ta} = 0$. $Y_t := \sum_{a=0}^m D_{ta} Y_t(a)$ and X_t denote the reward and the context observed in round t, respectively. We assume that a logged bandit feedback is generated by a *behavior policy* π_b as follows: (i) In each round $t = 1, ..., T$, $(Y_t(\cdot), X_t)$ is i.i.d. drawn from distribution G., (ii) Given X_t, an action is randomly chosen based on $\pi_b(\cdot|X_t)$, creating the action choice D_t and the associated reward Y_t. Suppose that π_b is fixed for all rounds, and thus D_t is i.i.d. across rounds. Because $(Y_t(\cdot), X_t)$ is i.i.d. across rounds and $Y_t = \sum_{a=0}^m D_{ta} Y_t(a)$, each observation (Y_t, X_t, D_t) is i.i.d. across rounds. Note that D_t is independent of $Y_t(\cdot)$ conditional on X_t. We describe examples of bandit algorithms in Appendix A.

3 Off-Policy Evaluation

3.1 Prediction Target

We are interested in using the historical logged bandit data to estimate the following *policy value* of any given *counterfactual policy* π which might be different from π_b:

$$V^\pi := \mathbb{E}_{(Y(\cdot),X)\sim G}[\sum_{a=0}^m Y(a)\pi(a|X)] = \mathbb{E}_{(Y(\cdot),X)\sim G,\ D\sim\pi_b}[\sum_{a=0}^m Y(a) D_a \frac{\pi(a|X)}{\pi_b(a|X)}] \tag{1}$$

where the last equality uses the independence of D and $Y(\cdot)$ conditional on X and the definition of $\pi_b(\cdot|X)$. We allow the counterfactual policy π to be degenerate, i.e., it may choose a particular action with probability 1. Estimating V^π before implementing π in an online environment is valuable because π may perform poorly and damage user satisfaction. Additionally, it is possible to select a counterfactual policy that maximizes the policy value by comparing their estimated performances.

3.2 Benchmark Estimators

Direct Method (DM). There are several approaches to estimate the value of the counterfactual policy. A widely-used method, DM (Beygelzimer & Langford, 2009), first learns a supervised machine learning model, such as random forest, ridge regression, and gradient boosting, to predict the mean reward function. DM then uses it to estimate the policy value as

$$\hat{V}_{DM}^\pi = \frac{1}{T}\sum_{t=1}^T \sum_{a=0}^m \pi(a|X_t)\hat{\mu}(a|X_t).$$

where $\hat{\mu}(a|x)$ is the estimated reward function. If $\hat{\mu}(a|x)$ is a good approximation to the mean reward function, this estimator accurately predicts the policy value of the counterfactual policy V^π. If

$\hat{\mu}(a|x)$ fails to approximate the mean reward function well, however, the final estimator is no longer consistent. The model misspecification issue is problematic because the extent of misspecification cannot be easily quantified from data (Farajtabar et al., 2018).

Inverse Probability Weighting (IPW). To alleviate the issue with DM, researchers often use another estimator called IPW (Precup et al., 2000; Strehl et al., 2010). IPW re-weights the rewards by the ratio of the counterfactual policy and behavior policy as

$$\hat{V}^{\pi}_{IPW} = \frac{1}{T}\sum_{t=1}^{T}\sum_{a=0}^{m} Y_t D_{ta} \frac{\pi(a|X_t)}{\pi_b(a|X_t)}.$$

When the behavior policy is known, the IPW estimator is unbiased and consistent for the policy value. However, it can have a large variance, especially when the counterfactual policy significantly deviates from the behavior policy.

Doubly Robust (DR). The final approach is DR (Dudík et al., 2014), which combines the above two estimators as

$$\hat{V}^{\pi}_{DR} = \frac{1}{T}\sum_{t=1}^{T}\sum_{a=0}^{m} \left\{ (Y_t - \hat{\mu}(a|X_t)) D_{ta} \frac{\pi(a|X_t)}{\pi_b(a|X_t)} + \pi(a|X_t)\hat{\mu}(a|X_t) \right\}.$$

DR mimics IPW to use a weighted version of rewards, but DR also uses the estimated mean reward function 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 *semiparametric efficient* (Narita et al., 2019) when the mean reward estimator is correctly specified. On the other hand, when it is wrong, this estimator can have larger asymptotic mean-squared-error than IPW (Kallus & Uehara, 2019) and perform poorly in practice (Kang et al., 2007).

4 Open Bandit Dataset and Pipeline

We apply and evaluate the above methods by using real-world data. Our data is logged bandit feedback data we call the *Open Bandit Dataset*. The dataset is provided by ZOZO, Inc.[2], the largest Japanese fashion e-commerce company with over 5 billion USD market capitalization (as of May 2020). The company recently started using context-free multi-armed bandit algorithms to recommend fashion items to users in their large-scale fashion e-commerce platform called ZOZOTOWN. We present examples of displayed fashion items as actions in Figure 4.

We collected the data in a 7-day~~s~~ experiment in late November 2019 on three "campaigns," corresponding to "all", "men's", and "women's" items, respectively. Each campaign randomly uses either the Random algorithm or the Bernoulli Thompson Sampling (Bernoulli TS) algorithm for each user impression. In the notation of our bandit setups, action a is one of the possible fashion items, while reward Y is a click indicator. We describe some statistics of the dataset in Table 1. The data is large and contains many millions of recommendation instances. The number of actions is also sizable, so this setting is challenging for bandit algorithms and their OPE.

To facilitate the usage of the Open Bandit Dataset, we also build a toolkit called the *Open Bandit Pipeline*. Our pipeline contains implementations of dataset preprocessing, behavior policy simulators, and evaluation of OPE estimators. This pipeline allows researchers to focus on building their OPE estimator and easily compare it with other methods in realistic and reproducible ways. To our knowledge, our real-world dataset and pipeline are the first to include multiple behavior policies, their implementations used in production, and their ground-truth policy values. These features enable the evaluation of OPE for the first time. We describe a related work and dataset in Appendix C and our pipeline package in Appendix D.

5 Experiments

We utilize our Open Bandit Dataset and apply the off-policy estimators in Section 3.2 to empirically search for a counterfactual policy that maximizes the CTR as our reward. Specifically, we aim to find

[2]https://corp.zozo.com/en/about/profile/

Figure 4: Fashion items as actions displayed in ZOZOTOWN. Three fashion items are simultaneously presented to a user in each recommendation.

Table 1: Statistics of the Open Bandit Dataset

Campaigns	Behavior Policies	#Data	#Items	Average Age	CTR (V^{π}) ±95% CI	Relative-CTR
ALL	Random	1,374,327	80	37.93	0.35% ±0.010	1.00
	Bernoulli TS	12,168,084			0.50% ±0.004	1.43
Men's	Random	452,949	34	37.68	0.51% ±0.021	1.48
	Bernoulli TS	4,077,727			0.67% ±0.008	1.94
Women's	Random	864,585	46	37.99	0.48% ±0.014	1.39
	Bernoulli TS	7,765,497			0.64% ±0.056	1.84

Notes: Bernoulli TS stands for Bernoulli Thompson Sampling. **#Data** is the total number of user impressions observed during the 7-day experiment. **#Items** is the total number of items having a positive probability of being recommended by each behavior policy. **Average Age** is the average age of users in each campaign. **CTR** is the percentage of a click being observed in log data, and this is the ground-truth performance of behavior policies in each campaign. 95% confidence interval (CI) of CTR is calculated based on a normal approximation of Bernoulli sampling. **Relative-CTR** is CTR relative to that of the Random policy for the "All" campaign.

a combination of a context set and a contextual bandit algorithm that would improve the CTR of the current context-free algorithm.

5.1 Selecting Off-Policy Estimator

We first select the best off-policy estimator among DM, IPW, and DR. For this purpose, we empirically evaluate their performance as follows (for each campaign separately):

1. For each of the Random and Bernoulli TS policies, randomly split the data collected by that policy into training (70%) and test (30%) sets.
2. Estimate the ground-truth value of each policy π by the empirical mean of clicks in the test set collected by that policy: $V^{\pi} = (T^{\pi}_{test})^{-1} \sum_{t=1}^{T^{\pi}_{test}} Y_t$, where T^{π}_{test} is the size of the test set of policy π.
3. Estimate the policy value of each policy by DM, IPW, and DR with the training set collected by the other policy.
4. Repeat the above process $K = 15$ times by sampling different training sets.
5. Compare the ground-truth and policy value estimated by the bagging prediction (Breiman, 1996).

Table 2: Comparing Relative-Estimation Errors of Alternative Off-policy Estimators

	Campaigns and Behavior Policies (Prediction Target)					
	All		**Men's**		**Women's**	
Methods	**Bernoulli TS**	**Random**	**Bernoulli TS**	**Random**	**Bernoulli TS**	**Random**
DM	0.64162	0.08482	0.42645	0.46560	0.62527	0.02357
IPW	**0.04556**	**0.07532**	**0.09352**	**0.09940**	**0.06473**	**0.00942**
DR	**0.04512**	**0.02063**	**0.08410**	**0.01997**	**0.06538**	**0.00744**

Notes: The relative-estimation errors of the three estimators are reported. The **red fonts** and **blue fonts** represent the best and the second best OPE estimators for each prediction target.

We measure each estimator's performance with the *Relative-Estimation Error* defined below:

$$\textit{Relative-Estimation Error} \text{ of } \hat{V}^{\pi} = \left| \frac{\left(K^{-1} \sum_{k=1}^{K} \hat{V}_k^{\pi}\right) - V^{\pi}}{V^{\pi}} \right|.$$

where V^{π} is a ground-truth policy value of π in a test set. Table 1 presents V^{π} for each pair of behavior policies and campaigns, and the small confidence intervals ensure that the ground-truth estimation is accurate. $K^{-1} \sum_{k=1}^{K} \hat{V}_k^{\pi}$ is a bagging prediction where $\hat{V}_k^{\pi}$ is an estimated policy value with the k-th bootstrapped samples. $K = 15$ is the number of folds.

For IPW and DR, we compute the true behavior policy by Monte Carlo simulation of the beta distribution used in Bernoulli TS. For DM and DR, we need to obtain a reward estimator $\hat{\mu}$. We do so by using LightGBM (Ke et al., 2017) implemented in *scikit-learn*[3] and training it with the whole training set. This procedure follows the method to obtain the regression function of DR in the literature (Narita et al., 2020).

The results of off-policy estimator selection are given in Table 2 and Figure 2. First, DM fails to predict the policy values in all settings. The failure of DM likely comes from the bias of the regression model. We observe that the prediction by LightGBM does not improve upon a naive prediction using the mean CTR for every prediction. Specifically, the improvements of the regression model over the naive prediction are only 1.51%-7.04% in the binary cross-entropy measure[4].

The problem with DM makes us expect that IPW and DR may perform better, because the two methods do not rely on the correct specification of the regression model. We confirm this expectation in Table 2, where IPW and DR drastically outperform DM. In particular, DR performs best in five out of the six scenarios. A possible reason for the best performance of DR is that DR is robust to the bias of the nuisance estimation, which is known as the *Neyman orthogonality* (Narita et al., 2020). These results motivate us to use DR to search for an optimal counterfactual policy below.

5.2 Selecting Context and Algorithm

Our final goal is to find a policy better than the context-free policy currently used by the company. We do so by using the DR estimator, which performs best in our off-policy estimator selection above. The counterfactual policy search space is as follows:

- **Algorithms**[5]: LOGISTIC ϵ-GREEDY with three different values of the exploration hyperparameter ($\epsilon = 0.01, 0.05, 0.1$), *logistic Thompson sampling (*LOGISTIC TS*)* (Chapelle & Li, 2011), *logistic Upper Confidence Bound (*LOGISTIC UCB*)* (Mahajan et al., 2012) with two different values of the exploration hyperparameter ($\alpha = 0.1, 1.0$)

[3]https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.HistGradientBoostingClassifier.html

[4]We present the performance of the regression model in Table 4 in Appendix B.

[5]We follow Algorithm 3 of (Chapelle & Li, 2011) to train logistic regression models for these algorithms.

Table 3: Comparing Relative-CTR of Alternative Counterfactual Policies.

Counterfactual Policies		Campaigns		
Algorithms	**Context Sets**	**All**	**Men's**	**Women's**
LOGISTIC ϵ-GREEDY ($\epsilon = 0.01$)	CONTEXT SET 1	0.9124 [0.8617, 0.9617]	0.7230 [0.6788, 0.7643]	0.9451 [0.8961, 0.9965]
	CONTEXT SET 2	0.8355 [0.7867, 0.8840]	1.1587 [1.1058, 1.2128]	1.1929 [1.1425, 1.2438]
LOGISTIC ϵ-GREEDY ($\epsilon = 0.05$)	CONTEXT SET 1	1.1074 [1.0049, 1.2121]	0.8957 [0.8528, 0.9443]	0.9991 [0.8987, 1.0940]
	CONTEXT SET 2	**1.2221 [1.1254, 1.3269]**	1.3359 [1.2324, 1.4345]	1.2195 [1.1367, 1.2979]
LOGISTIC ϵ-GREEDY ($\epsilon = 0.1$)	CONTEXT SET 1	0.8951 [0.8609, 0.9300]	1.0193 [0.9679, 1.0658]	0.7639 [0.7051, 0.8163]
	CONTEXT SET 2	1.0979 [1.0185, 1.1805]	1.2165 [1.1451, 1.2869]	**1.5253 [1.3731, 1.6912]**
LOGISTIC TS	CONTEXT SET 1	1.0162 [0.9595, 1.0767]	1.0094 [0.9419, 1.0822]	1.1996 [1.1271, 1.2945]
	CONTEXT SET 2	1.0933 [0.9575, 1.2091]	**1.4192 [1.3046, 1.5415]**	1.2064 [1.1179, 1.2920]
LOGISTIC UCB ($\alpha = 0.1$)	CONTEXT SET 1	1.2184 [1.0945, 1.3572]	1.1222 [1.0647, 1.1857]	0.8889 [0.8360, 0.9422]
	CONTEXT SET 2	**1.6381 [1.3333, 2.0067]**	**1.6459 [1.5257, 1.7685]**	**1.5676 [1.4307, 1.7049]**
LOGISTIC UCB ($\alpha = 1.0$)	CONTEXT SET 1	0.5000 [0.2873, 0.7243]	0.9535 [0.8659, 1.0401]	0.7823 [0.6471, 0.9043]
	CONTEXT SET 2	0.4763 [0.2667, 0.6957]	1.1306 [0.8896, 1.3026]	0.9141 [0.7386, 1.0664]

Notes: This table presents the estimated CTRs relative to the ground-truth policy value of Bernoulli Thompson Sampling (the current best performing policy). The averaged relative-CTRs and their 95% confidence intervals induced by nonparametric bootstrap-like procedure are reported. We present the method to obtain the confidence intervals in Appendix B. The **red fonts** and **blue fonts** represent the best and the second best policies for each campaign.

- **Context Sets**: CONTEXT SET 1 (user features only, such as age, gender, and length of membership), CONTEXT SET 2 (Context Set 1 plus user-item affinity induced by the number of past clicks observed between each user-item pair)

The search space consists of six possible contextual bandit algorithms and two possible context sets, which results in 12 possible counterfactual policies.[6]

The off-policy evaluation of these potential policies is reported in Table 3 and Figure 3. We compare candidate counterfactual policies by their predicted policy values (CTRs) relative to that of context-free Bernoulli TS (current best policy):

$$\textit{relative-CTR of } \pi = \frac{K^{-1}\sum_{k=1}^{K} \hat{V}^{\pi}_{DR_k}}{V^{\pi_{\text{Bernoulli TS}}}},$$

where the numerator is the bagging prediction of the performance of a counterfactual policy π. The denominator is the ground-truth performance of the current best policy, which is estimated by the empirical mean of clicks using the test sets of data collected by Bernoulli TS as in Section 5.2.

Table 3 reports the resulting performance of several counterfactual policies. Several findings emerge. First, using only Context Set 1 (user features) is worse than the current policy using no feature. Naively incorporating more contextual features may therefore damage user satisfaction. The choice of hyperparameters also has a big impact on policy performance. For example, logistic UCB with $\alpha = 0.1$ and Context Set 2 outperforms the current policy by over 64% in men's campaign. On the other hand, logistic UCB with $\alpha = 1.0$ and Context Set 2 *under*performs it by over 42%.

Overall, the best counterfactual policy is the one to combine Context Set 2 and logistic UCB ($\alpha = 0.1$). This policy outperforms the current algorithm by 63.8% in campaign "all", 64.5% in campaign "men's", and 56.5% in campaign "women's." This finding leads the company to change its algorithm as suggested by the finding.

6 Conclusion and Future Work

To enable realistic and reproducible evaluation of off-policy evaluation of bandit algorithms, we have publicized the Open Bandit Dataset–a benchmark logged bandit dataset collected on a large-scale fashion e-commerce platform. The data comes with the Open Bandit Pipeline, a collection of

[6]We modify the contextual bandit algorithms to adjust to our top-3 recommendation setting shown in Figure 4. For example, modified logistic TS selects three actions with the three highest sampled rewards.

implementations that makes it easy to evaluate and compare different OPE estimators. We expect them to facilitate the understanding of the empirical properties of the OPE techniques and address experimental inconsistencies in the literature.

In addition to developing the Open Bandit Dataset and Pipeline, we have presented a case study using the dataset. We first performed an off-policy estimator selection to find the best estimator among DM (Direct Method), IPW (Inverse Probability Weighting), and DR (Doubly Robust). We then conducted a counterfactual policy search to find a better combination of a context set and a contextual bandit algorithm. Through the exercise, OPE finds a new policy that would significantly improve on the current policy, generating valuable managerial conclusions. Practitioners can replicate the counterfactual policy search analysis with their own datasets by using our pipeline and implementations.

In the near future, we plan to publicize the performance of the selected counterfactual policy in an online environment. Such an evaluation will produce additional log data generated by the contextual policy (while the current open dataset contains only log data generated by the old context-free policy). We aim to constantly expand and improve the Open Bandit Dataset to include more data and tasks.

References

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*, pp. 129–138, 2009.

Léon Bottou, Jonas Peters, Joaquin Quiñonero-Candela, Denis X Charles, D Max Chickering, Elon Portugaly, Dipankar Ray, Patrice Simard, and Ed Snelson. Counterfactual Reasoning and Learning Systems: The Example of Computational Advertising. *Journal of Machine Learning Research*, 14 (1):3207–3260, 2013.

Leo Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.

Olivier Chapelle and Lihong Li. An empirical evaluation of thompson sampling. In *Advances in neural information processing systems*, pp. 2249–2257, 2011.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Miroslav Dudík, Dumitru Erhan, John Langford, and Lihong Li. Doubly Robust Policy Evaluation and Optimization. *Statistical Science*, 29:485–511, 2014.

Vijay Prakash Dwivedi, Chaitanya K Joshi, Thomas Laurent, Yoshua Bengio, and Xavier Bresson. Benchmarking graph neural networks. *arXiv preprint arXiv:2003.00982*, 2020.

Mehrdad Farajtabar, Yinlam Chow, and Mohammad Ghavamzadeh. More robust doubly robust off-policy evaluation. In *Proceedings of the 35th International Conference on Machine Learning*, pp. 1447–1456, 2018.

Alexandre Gilotte, Clément Calauzènes, Thomas Nedelec, Alexandre Abraham, and Simon Dollé. Offline a/b testing for recommender systems. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, pp. 198–206, 2018.

Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 580–587, 2014.

Alois Gruson, Praveen Chandar, Christophe Charbuillet, James McInerney, Samantha Hansen, Damien Tardieu, and Ben Carterette. Offline evaluation to make decisions about playlistrecommendation algorithms. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pp. 420–428, 2019.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: Datasets for machine learning on graphs. *arXiv preprint arXiv:2005.00687*, 2020.

Alex Irpan, Kanishka Rao, Konstantinos Bousmalis, Chris Harris, Julian Ibarz, and Sergey Levine. Off-policy evaluation via off-policy classification. In *Advances in Neural Information Processing Systems*, 2019.

Nan Jiang and Lihong Li. Doubly robust off-policy value evaluation for reinforcement learning. In *Proceedings of the 33rd International Conference on Machine Learning*, pp. 652–661, 2016.

Nathan Kallus and Masatoshi Uehara. Intrinsically efficient, stable, and bounded off-policy evaluation for reinforcement learning. In *Advances in Neural Information Processing Systems*, 2019.

Joseph DY Kang, Joseph L Schafer, et al. Demystifying double robustness: A comparison of alternative strategies for estimating a population mean from incomplete data. *Statistical science*, 22(4):523–539, 2007.

Masahiro Kato, Masatoshi Uehara, and Shota Yasui. Off-policy evaluation and learning for external validity under a covariate shift. *arXiv preprint arXiv:2002.11642*, 2020.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems*, pp. 3146–3154, 2017.

Damien Lefortier, Adith Swaminathan, Xiaotao Gu, Thorsten Joachims, and Maarten de Rijke. Large-scale validation of counterfactual learning methods: A test-bed. *arXiv preprint arXiv:1612.00367*, 2016.

Lihong Li, Wei Chu, John Langford, and Robert E Schapire. A Contextual-bandit Approach to Personalized News Article Recommendation. In *Proceedings of the 19th International Conference on World Wide Web*, pp. 661–670. ACM, 2010.

Lihong Li, Wei Chu, John Langford, and Xuanhui Wang. Unbiased Offline Evaluation of Contextual-bandit-based News Article Recommendation Algorithms. In *Proceedings of the Fourth ACM International Conference on Web Search and Data Mining*, pp. 297–306, 2011.

Lihong Li, Wei Chu, John Langford, Taesup Moon, and Xuanhui Wang. An Unbiased Offline Evaluation of Contextual Bandit Algorithms with Generalized Linear Models. In *Journal of Machine Learning Research: Workshop and Conference Proceedings*, volume 26, pp. 19–36, 2012.

Yao Liu, Omer Gottesman, Aniruddh Raghu, Matthieu Komorowski, Aldo A Faisal, Finale Doshi-Velez, and Emma Brunskill. Representation balancing mdps for off-policy policy evaluation. In *Advances in Neural Information Processing Systems*, pp. 2644–2653, 2018.

Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3431–3440, 2015.

Dhruv Kumar Mahajan, Rajeev Rastogi, Charu Tiwari, and Adway Mitra. Logucb: an explore-exploit algorithm for comments recommendation. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, pp. 6–15, 2012.

Travis Mandel, Yun-En Liu, Sergey Levine, Emma Brunskill, and Zoran Popovic. Offline policy evaluation across representations with applications to educational games. In *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*, pp. 1077–1084, 2014.

Susan A Murphy, Mark J van der Laan, James M Robins, and Conduct Problems Prevention Research Group. Marginal mean models for dynamic regimes. *Journal of the American Statistical Association*, 96(456):1410–1423, 2001.

Yusuke Narita, Shota Yasui, and Kohei Yata. Efficient counterfactual learning from bandit feedback. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 4634–4641, 2019.

Yusuke Narita, Shota Yasui, and Kohei Yata. Safe counterfactual reinforcement learning. *arXiv preprint arXiv:2002.08536*, 2020.

Doina Precup, Richard S. Sutton, and Satinder Singh. Eligibility Traces for Off-Policy Policy Evaluation. In *Proceedings of the 17th International Conference on Machine Learning*, pp. 759–766, 2000.

Alex Strehl, John Langford, Lihong Li, and Sham M Kakade. Learning from Logged Implicit Exploration Data. In *Advances in Neural Information Processing Systems*, pp. 2217–2225, 2010.

Adith Swaminathan and Thorsten Joachims. Batch Learning from Logged Bandit Feedback through Counterfactual Risk Minimization. *Journal of Machine Learning Research*, 16:1731–1755, 2015a.

Adith Swaminathan and Thorsten Joachims. The Self-normalized Estimator for Counterfactual Learning. In *Advances in Neural Information Processing Systems*, pp. 3231–3239, 2015b.

Adith Swaminathan, Akshay Krishnamurthy, Alekh Agarwal, Miro Dudik, John Langford, Damien Jose, and Imed Zitouni. Off-policy Evaluation for Slate Recommendation. In *Advances in Neural Information Processing Systems*, pp. 3635–3645, 2017.

Philip Thomas and Emma Brunskill. Data-efficient Off-policy Policy Evaluation for Reinforcement Learning. In *Proceedings of the 33rd International Conference on Machine Learning*, pp. 2139–2148, 2016.

Philip Thomas, Georgios Theocharous, and Mohammad Ghavamzadeh. High confidence policy improvement. In *Proceedings of the 32th International Conference on Machine Learning*, pp. 2380–2388, 2015a.

Philip S Thomas, Georgios Theocharous, and Mohammad Ghavamzadeh. High-Confidence Off-Policy Evaluation. *AAAI*, pp. 3000–3006, 2015b.

Nikos Vlassis, Aurelien Bibaut, Maria Dimakopoulou, and Tony Jebara. On the design of estimators for bandit off-policy evaluation. In *International Conference on Machine Learning*, pp. 6468–6476, 2019.

Cameron Voloshin, Hoang M Le, Nan Jiang, and Yisong Yue. Empirical study of off-policy policy evaluation for reinforcement learning. *arXiv preprint arXiv:1911.06854*, 2019.

Yu-Xiang Wang, Alekh Agarwal, and Miroslav Dudik. Optimal and Adaptive Off-policy Evaluation in Contextual Bandits. In *Proceedings of the 34th International Conference on Machine Learning*, pp. 3589–3597, 2017.

Tengyang Xie, Yifei Ma, and Yu-Xiang Wang. Towards optimal off-policy evaluation for reinforcement learning with marginalized importance sampling. In *Advances in Neural Information Processing Systems*, pp. 9665–9675, 2019.

A Examples

Our setup allows for many popular multi-armed bandit algorithms, as the following examples illustrate.

Example 1 (Random A/B testing)**.** *We always choose each action uniformly at random:* $\pi_b(\cdot|X) = \frac{1}{m+1}$ *always holds for any* $a \in \mathcal{A}$ *and* $X \in \mathcal{X}$*.*

Example 2 (Bernoulli Thompson Sampling)**.** *When the context* X_t *is given, we sample the potential reward* $\tilde{Y}(a)$ *from the beta distribution* $Beta(S_{ta} + \alpha, F_{ta} + \beta)$ *for each action, where* $S_{ta} = \sum_{t'=1}^{t-1} Y_{t'} D_{t'a}, F_{ta} = (t-1) - S_{ta}$*.* α, β *are the parameters of the prior Beta distribution. We then choose the action with the highest sampled potential reward,* $\underset{a' \in \mathcal{A}}{\operatorname{argmax}} \tilde{Y}(a')$*. As a result, this algorithm chooses actions with the following probabilities:*

$$\pi(a|X_t) = \Pr\{a = \underset{a' \in \mathcal{A}}{\operatorname{argmax}} \tilde{Y}(a')\}.$$

Table 4: Prediction Accuracy of the Regression Model ($\hat{\mu}$)

Campaigns	Behavior Policies	Average Metrics $\pm$StdDev	
		Area Under the ROC Curve (AUC)	**Relative Cross Entropy (RCE)**
ALL	RANDOM	0.7083 $\pm$0.0062	0.0704 $\pm$0.0127
	BERNOULLI TS	0.6363 $\pm$0.0015	0.0253 $\pm$0.0007
MEN'S	RANDOM	0.6988 $\pm$0.0184	0.0454 $\pm$0.0184
	BERNOULLI TS	0.6476 $\pm$0.0034	0.0208 $\pm$0.0017
WOMEN'S	RANDOM	0.6689 $\pm$0.0105	0.0230 $\pm$0.0061
	BERNOULLI TS	0.6147 $\pm$0.0022	0.0151 $\pm$0.0010

Notes: This table presents the prediction performance of the regression model on validation sets. The averaged metrics and their standard deviations (StdDev) over 15 different train-validation splits are reported. **RCE** is the improvement of a prediction relative to the naive prediction, which predicts the mean CTR for every data. **AUC** is the probability that positive samples are ranked higher than negative items by a classifier under consideration. Note that, for both metrics, large value is better. We describe the formal definitions of these metrics in Appendix B.

B Additional Experimental Settings and Results

B.1 Prediction Accuracy of the Regression Model

We evaluate the performance of the regression model by using the following two evaluation metrics in classification.

Relative Cross Entropy (RCE): RCE is defined as the improvement of a prediction relative to the naive prediction, which predicts the mean CTR for every data. We calculate this metric using a size n of validation samples $\{(x_i, y_i)\}_{i=1}^{n}$ as:

$$\textit{RCE of } \hat{\mu} = \frac{\sum_{i=1}^{n} y_i \log(\hat{\mu}(x_i)) + (1-y_i)\log(1-\hat{\mu}(x_i))}{\sum_{i=1}^{n} y_i \log(\hat{\mu}_{naive}) + (1-y_i)\log(1-\hat{\mu}_{naive})} - 1$$

where $\hat{\mu}_{naive} = n^{-1}\sum_{i=1}^{n} y_i$ is the naive prediction. A larger value of RCE means better performance of a predictor.

Area Under the ROC Curve (AUC): AUC is defined as the probability that positive samples are ranked higher than negative items by a classifier under consideration.

$$\textit{AUC of } \hat{\mu} = \frac{1}{n^{\mathrm{pos}} n^{\mathrm{neg}}} \sum_{i=1}^{n^{\mathrm{pos}}} \sum_{j=1}^{n^{\mathrm{neg}}} \mathbb{I}\{\hat{\mu}(x_i^{\mathrm{pos}}) > \hat{\mu}(x_j^{\mathrm{neg}})\}$$

where $\mathbb{I}\{\cdot\}$ is the indicator function. $\{x_i^{\mathrm{pos}}\}_{i=1}^{n^{\mathrm{pos}}}$ and $\{x_j^{\mathrm{neg}}\}_{j=1}^{n^{\mathrm{neg}}}$ are sets of positive and negative samples in the validation set, respectively. A larger value of AUC means better performance of a predictor.

Table 4 presents the performance of the regression model on validation sets.

B.2 Estimating Confidence Intervals

To estimate confidence intervals in Table 3, we first construct an empirical cumulative distribution function $\hat{F}_K$ by $\{\hat{V}_k^{\pi}\}_{k=1}^{K}$. Then, we draw bootstrap samples $\hat{V}_1^{\pi,\star}, \ldots, \hat{V}_K^{\pi,\star}$ from $\hat{F}_K$ and compute its empirical mean by $\hat{V}^{\pi,\star} = K^{-1}\sum_{k=1}^{K} \hat{V}_k^{\pi,\star}$. We iterated this process $B = 10,000$ times and construct an empirical cumulative distribution function $\hat{F}_B$ by $\{\hat{V}_b^{\pi,\star}\}_{b=1}^{B}$. Finally, we estimate the 95% confidence intervals by using 2.5 and 97.5 percentiles of $\hat{F}_B$.

C Related Work and Dataset

Our work is most closely related to (Lefortier et al., 2016). (Lefortier et al., 2016) introduces a large-scale logged bandit feedback data (Criteo dataset) from a leading company in the display advertising, Criteo. The data contains context vectors of user impressions, advertisements (ads) as actions, and click indicators as reward. It also provides the ex ante probability of each ad being selected by the behavior policy. Therefore, this data can be used to compare different *off-policy learning* methods, which aim to learn a new bandit policy using only log data generated by a behavior policy. However, the Criteo data has limitations, which we overcome as follows:

- The Criteo dataset does not provide the code (production implementation) of their behavior policy. Moreover, the data was collected by running only a single behavior policy. As a result, this data cannot be used for evaluation and comparison of different OPE estimators.

  $\rightarrow$ In contrast, we provide the code of our behavior policies (i.e., Bernoulli TS and Random), which allows researchers to re-run the same behavior policies on the log data. Our open data also contains logged bandit feedback data generated by *multiple* behavior policies. It enables the evaluation and comparison of different OPE estimators, as we demonstrated in Section 5.2. This is the first large-scale bandit dataset that enables such evaluation of OPE with the ground-truth policy value of behavior policies.

- Lefortier et al. (2016) does not provide a pipeline implementation to handle their data. Researchers have to re-implement the experimental environment by themselves before implementing their own methods. This may lead to inconsistent experimental conditions across different studies, potentially causing reproducibility issues.

  $\rightarrow$ We implement the Open Bandit Pipeline to simplify and standardize the experimental processing of bandit algorithms and OPE using our open data. This tool contributes to the reproducible and transparent use of our data.

We summarize these key differences between the Criteo Dataset and our Open Bandit Dataset in Table 5.

Table 5: Comparison of Currently Available Large-scale Bandit Datasets

	Datasets	
	Criteo Dataset (Lefortier et al., 2016)	**Open Bandit Dataset** (ours)
Domain	Display Advertising	Fashion E-Commerce
#Data	>= 103M	>= 26M (will increase)
#Behavior Policies	1	2 (will increase)
Random A/B Test Data	✗	✔
Behavior Policy Code	✗	✔
Evaluation of Bandit Algorithms	✔	✔
Evaluation of OPE	✗	✔
Pipeline Implementation	✗	✔

Notes: **#Data** is the total number of samples included in the data. **#Behavior Policies** is the number of behavior policies that were used to collect the data. **Random A/B Test Data** is whether the data contains a subset of data generated by the uniform random policy. **Behavior Policy Code** is whether the code (production implementation) of behavior policies is publicized along with the data. **Evaluation of Bandit Algorithms** is whether it is possible to use the data to evaluate a new bandit algorithm. **Evaluation of OPE** is whether it is possible to use the data to evaluate a new OPE estimator. **Pipeline Implementation** is whether a pipeline tool to handle the data is available.

D Open Bandit Pipeline (OBP) Package

As described in Section 4, *Open Bandit Pipeline* contains implementations of dataset preprocessing, behavior policy simulator, and evaluation of OPE estimators.

Below, we show an example of conducting an offline evaluation of the performance of BernoulliTS using Inverse Probability Weighting as an OPE estimator and the Random policy as a behavior policy. We see that only ten lines of code are sufficient to complete OPE from scratch (Code Snippet 1).

```
# a case for implementing OPE of the BernoulliTS policy
# using log data generated by the Random policy
>>> from obp.dataset import OpenBanditDataset
>>> from obp.policy import BernoulliTS
>>> from obp.simulator import OfflineBanditSimulator

# (1) Data loading and preprocessing
>>> dataset = OpenBanditDataset(behavior_policy="random", campaign="all")
>>> train, test = dataset.split_data(test_size=0.3, random_state=42)

# (2) Offline Bandit Simulation
>>> simulator = OfflineBanditSimulator(train=train)
>>> counterfactual_policy = BernoulliTS(
>>>   n_actions=dataset.n_actions,
>>>   len_list=dataset.len_list,
>>>   random_state=42)
>>> simulator.simulate(policy=counterfactual_policy, train=train)

# (3) Off-Policy Evaluation
>>> estimated_policy_value = simulator.inverse_probability_weighting()

# estimated performance of BernoulliTS relative to the ground-truth performance of
    Random
>>> relative_policy_value_of_bernoulli_ts = estimated_policy_value /
    test["reward"].mean()
>>> print(relative_policy_value_of_bernoulli_ts) # 1.21428...
```

Code Snippet 1: **Overall Flow of using OBP**

In the following subsections, we explain some important features in the example flow.

D.1 Data Loading and Preprocessing

We prepare easy-to-use data loader for Open Bandit Dataset. The dataset class will then download, preprocess, split the data in a standardized manner. Users can also implement their own feature engineering in `OpenBanditDataset` class easily. Our implementation is general and can be used to handle future datasets other than our open data.

```
# Load and preprocess raw data in "All" campaign collected by the Random policy
>>> dataset = OpenBanditDataset(behavior_policy="random", campaign="all")
# Split the data into 70% training and 30% test sets
>>> train, test = dataset.split_data(test_size=0.3, random_state=0)
```

Code Snippet 2: **Data Loading and Preprcessing, and Splitting**

D.2 Bandit Simulation

After preparing our data, we now run an offline bandit simulation on the logged bandit feedback as follows.

```
# Define a simulator object
>>> simulator = OfflineBanditSimulator(train=train)
# Define a counterfacutal policy, which is the Bernoulli TS policy here
>>> counterfactual_policy = BernoulliTS(
>>>   n_actions=dataset.n_actions,
>>>   len_list=dataset.len_list,
>>>   random_state=42)
# Run an offline bandit simulation on the training set
>>> simulator.simulate(policy=counterfactual_policy, train=train)
```

Code Snippet 3: **Offline Bandit Simulation**

The simulation takes `BanditPolicy` class and `train` (a dictionary) as inputs and runs offline bandit simulation of a given bandit policy (Bernoulli TS here).

D.3 Off-Policy Evaluation

Our final step is OPE, which attempts to estimate the performance of bandit algorithms using log data generated by offline bandit simulations. Our pipeline also provides an easy procedure for doing OPE as follows.

```
# Estimate the policy value of BernoulliTS based on actions selected by that policy
>>> estimated_policy_value = simulator.inverse_probability_weighting()

# Comapre the estimated performance of BernoulliTS (counterfactual policy)
# with the ground-truth performance of Random (behavior policy)
>>> relative_policy_value_of_bernoulli_ts = estimated_policy_value /
    test["reward"].mean()
# Our OPE procedure estimates that BernoulliTS improves Random by 21.4%
>>> print(relative_policy_value_of_bernoulli_ts) # 1.21428...
```

Code Snippet 3: **Off-Policy Evaluation by IPW**

Researchers can easily implement their own OPE estimators as a method of `OfflineBanditSimulator` class. In the package, we provide implementations of methods described in Section 3.2 as examples in the `estimator` module.