

Off-Policy Evaluationの基礎と Open Bandit Dataset & Pipelineの紹介



Yuta Saito¹, Shunsuke Aihara²,

Megumi Matsutani², and Yusuke Narita³

¹Tokyo Institute of Technology ²ZOZO Technologies, Inc. ³Yale University.

目次

- 予測精度の評価よりも意思決定を評価することを考える
- 意思決定の評価方法: Off-Policy Evaluation
- Off-Policy Evaluationにおける標準的な推定量
- Off-Policy Evaluation研究の課題
- Open Bandit Dataset & Pipelineの公開 (our work!)

導入

機械学習を予測ではなく意思決定のために使う例

「機械学習による意思決定」イメージを掴むため簡単な例を考える

- 商品AかBのどちらかを推薦することで、**売上を最大化**したい
- 両方の商品の定価は1,000円で、推薦枠は1つしかないとする
- 購入確率はユーザーに非依存
- 推薦枠に入れることで、商品の購入確率が変化することがある

$$\text{期待売上} = \text{商品の定価} \times \text{商品の購入確率}$$

機械学習を予測ではなく意思決定のために使う例

推薦有無によって購入確率が変化するような人工的設定

	商品の 定価	推薦有時の 購入確率	推薦無時の 購入確率	購入確率 上昇幅
商品A	1,000	5.0%	1.0%	4.0%
商品B	1,000	5.5%	2.0%	3.5%

推薦という介入が購入確率に対して持っている因果効果

機械学習を予測ではなく意思決定のために使う例

- 商品Aを推薦した場合

	商品の 定価	推薦有時の 購入確率	推薦無時の 購入確率	購入確率 上昇幅	期待 売上
商品A	1,000	5.0%	1.0%	4.0%	50
商品B	1,000	5.5%	2.0%	3.5%	20

合計期待売上 = 50 + 20 = 70 円

機械学習を予測ではなく意思決定のために使う例

- 商品Bを推薦した場合

	商品の 定価	推薦有時の 購入確率	推薦無時の 購入確率	購入確率 上昇幅	期待 売上
商品A	1,000	5.0%	1.0%	4.0%	10
商品B	1,000	5.5%	2.0%	3.5%	55

合計期待売上 = 10 + 55 = 65 円

商品A or Bの出し分けをすることで売り上げを最大化

- 商品Aを推薦することで期待売上を最大化できる
＝推薦による購入確率上昇幅が大きい商品を推薦

	商品の 定価	推薦有時の 購入確率	推薦無時の 購入確率	購入確率 上昇幅
商品A	1,000	5.0%	1.0%	4.0%
商品B	1,000	5.5%	2.0%	3.5%

とあるデータサイエンティストの物語

- 推薦担当のデータサイエンティストは機械学習を使って
購入確率上昇幅を予測し、推薦施策を作ることにした

	商品の 定価	推薦有時の 購入確率	推薦無時の 購入確率	購入確率 上昇幅
商品A	1,000	5.0%	1.0%	4.0%
商品B	1,000	5.5%	2.0%	3.5%

データサイエンティストが施策を作るためにとった手順

データサイエンティストは次ような手順をとることにした

1. 訓練用ログデータを使って商品AとBを推薦した場合
の購入確率上昇幅を予測する機械学習モデルを2つ学習する
2. 検証用ログデータを使って学習した2つのモデルの予測精度を
オフライン評価する
3. オフライン評価において予測精度がよかった方のモデルに基づいて
推薦施策を作り、実戦投入する

データサイエンティストが施策を作るためにとった手順

データサイエンティストは次ような手順をとることにした

1. 訓練用ログデータを使って商品AとBを推薦した場合の購入確率上昇幅を予測する機械学習モデルを2つ学習する
2. **検証用ログデータを使って学習した2つのモデルの予測精度をオフライン評価する(本日の着目点！)**
3. オフライン評価において予測精度がよかった方のモデルに基づいて推薦施策を作り、実戦投入する

機械学習のオフライン評価に潜む罠

データサイエンティストは次ようなオフライン評価の結果を得た

	購入確率上昇幅に対する 予測誤差の評価
機械学習モデルa	15%
機械学習モデルb	30%



実践投入！

* 真の上昇幅が0.05で予測誤差が15%ならば[0.0425, 0.0575]の範囲の予測が可能

機械学習のオフライン評価に潜む罠

モデルaは予測の精度は良い

	商品Aの 購入確率上昇幅	商品Bの 購入確率上昇幅
真の値	4.0%	3.5%
モデルa の予測値	3.5%	3.8%
モデルb の予測値	5.0%	2.5%

予測誤差15%

予測誤差30%

機械学習のオフライン評価に潜む罠

モデルaは予測の精度は良いものの、意思決定には失敗している

	商品Aの 購入確率上昇幅	商品Bの 購入確率上昇幅	推薦意思決定
真の値	4.0%	3.5%	商品Aを推薦
モデルa の予測値	3.5%	3.8%	商品Bを推薦
モデルb の予測値	5.0%	2.5%	商品Aを推薦

なぜオフライン評価で間違いが起こったのか？

データサイエンティストがやっていたこと

あくまで中間生産物
に過ぎない

「機械学習による購入確率上昇幅の予測」に基づいて

「商品AとBのどちらの商品を推薦するかを決定」

売上に響く意思決定の部分

なぜオフライン評価で間違いが起こったのか？

予測誤差の方向(過小 or 過大評価)によって意思決定の性能は変わる



「機械学習による購入確率上昇幅の予測」に基づいて

「商品AとBのどちらの商品を推薦するかを決定」

売上に響くのは推薦の意思決定の部分にも関わらず予測精度を
評価してしまったがために、モデル選択に失敗していた

予測精度のオフライン評価結果はわかりにくい...

先ほどの話に加えて、

「モデルaのMSEは0.1で、モデルbのMSEは0.3です」

という評価を得るよりも

「モデルaによる期待売上は70円でモデルbによる

期待売上は65円」という評価をできた方がわかりやすい

機械学習の予測を意思決定のために利用している例

以下のような例では**予測値をそのままではなく意思決定のために使っている**ので**最終的な意思決定の性能を評価すべき**

- クリック率予測に基づいてどの検索結果を提示するかを決める
- 個別的因果効果予測に基づいてクーポンを配るか否かを決める
- レイティング予測に基づいてどのアイテムを推薦するか決める
- 購入確率 (CVR) 予測に基づいてオークション入札額を決める

予測誤差の方向(過小 or 過大評価)によって意思決定の性能は変わる

ここまでのまとめ

- 特にテック企業における機械学習の応用では、予測値をそのまま使うというよりむしろ意思決定を作るために使うことが多い
- その場合に、単なる中間生産物である予測精度を評価してしまうと意思決定の意味では性能の悪いモデルを投入してしまう恐れが..

以降、意思決定の性能をオフライン評価する方法を考える

用いる記号

適宜補足を加えるが、主に次のnotationを用いる

- x : 特徴量ベクトル (contextとされることも)
- a : 選択する行動 (action)を表す離散変数
- $Y(a)$: 行動 a が選択された場合の潜在目的変数
- $\pi(x)$: x に対してどの行動を選択するか (**意思決定policy**)

過去に蓄積されたデータを用いて意思決定policyの性能を
オフライン評価したい = **Off-Policy Evaluation (OPE)**

意思決定policyのオフライン評価 (OPE) の流れ

意思決定policyの性能を次のように定義する

$$V(\pi) = \mathbb{E}_{(X,Y)} [Y(\pi(X))]$$

➡ π を適用することによって得られる目的変数の期待値

例) Yがクリック有無ならば性能はpolicyを導入したときのクリック率

意思決定policyのオフライン評価 (OPE) の流れ

OPEに使える過去の蓄積データは次のような形をしている

$$\mathcal{D} = \left\{ \left(x_i, a_i, Y_i^{obs} \right) \right\}_{i=1}^n$$

$$a_i = \pi_b(x_i)$$

過去の意思決定policy
(旧ロジック)による選択

$$Y_i^{obs} = Y_i(a_i)$$

選択枝aに対応した
潜在目的変数のみが観測

意思決定policyのオフライン評価 (OPE) の流れ

OPEに使える過去の蓄積データは次のような形をしている

$$\mathcal{D} = \left\{ \left(x_i, a_i, Y_i^{obs} \right) \right\}_{i=1}^n$$

x_i に a_i を施して Y_i^{obs} という結果を観測したという
過去の意思決定の結果の観測

意思決定policy学習の流れ: policyの性能を推定する

やりたいこと: 意思決定policyの真の性能を正確に推定

$$V(\pi) \approx \hat{V}(\pi; \mathcal{D})$$

観測データ \mathcal{D} を用いた π の性能の推定値

OPEの論文はどんな \hat{V} を使えばうまく真の性能が推定できるかを議論

意思決定policy学習の流れ: policyの性能を推定する

例) 商品A or Bを意思決定policyによってユーザーごとに個別推薦

旧ロジックが収集した 過去データ(D)			新たな意思決定policy を過去データ上で動作
ユーザー 特徴量	過去の 推薦	観測 目的変数	新policyによる推薦
x_1	商品A	Y(A)	商品A
x_2	商品B	Y(B)	商品A
x_3	商品A	Y(A)	商品A
x_4	商品B	Y(B)	商品A

検証用データに対し予測
をかけているイメージ



意思決定policy学習の流れ: policyの性能を推定する

新旧意思決定policyの選択が一致している時は結末がわかる

旧ロジックが収集した 過去データ(D)			新たな意思決定policy を過去データ上で動作
ユーザー 特徴量	過去の 推薦	観測 目的変数	新policyによる推薦
x_1	商品A	Y(A)	商品A
x_2	商品B	Y(B)	商品A
x_3	商品A	Y(A)	商品A
x_4	商品B	Y(B)	商品A

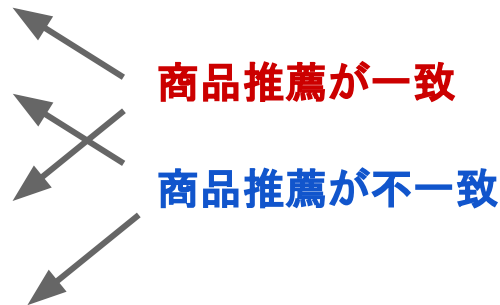


商品推薦が一致

意思決定policy学習の流れ: policyの性能を推定する

新旧意思決定policyの選択が一致していない時は結末が未観測

旧ロジックが収集した 過去データ(D)			新たな意思決定policy を過去データ上で動作
ユーザー 特徴量	過去の 推薦	観測 目的変数	新policyによる推薦
x_1	商品A	Y(A)	商品A
x_2	商品B	Y(B)	商品A
x_3	商品A	Y(A)	商品A
x_4	商品B	Y(B)	商品A



意思決定policy学習の流れ: policyの性能を推定する

やりたいこと: 意思決定policyの真の性能を正確に推定

$$V(\pi) \approx \hat{V}(\pi; \mathcal{D})$$

観測データ \mathcal{D} を用いた π の性能の推定値

過去のpolicyと評価したい新たなpolicyの選択の不一致を解決する必要

標準的な推定量

Off-Policy Evaluationの性能

OPEの性能(オフライン評価の正確さ)はMSEで測られる

$$\begin{aligned}MSE(\hat{V}) &= \mathbb{E}_{\mathcal{D}} \left[(V(\pi) - \hat{V}(\pi; \mathcal{D}))^2 \right] \\&= \underbrace{Bias(\hat{V})^2 + Var(\hat{V})}\end{aligned}$$

推定量のbiasとvarianceに分解できる

Direct Method (DM)

観測データから事前にYを予測するモデルを学習、それをOPEに用いる

$$\hat{V}_{DM}(\pi; \mathcal{D}) = \frac{1}{n} \sum_{i=1}^n \hat{\mu}(x_i, \pi(x_i); \mathcal{D})$$

目的変数の事前推定モデル

- データを用いて事前に推定

$$Y_i(a_i) \approx \hat{\mu}(x_i, a_i; \mathcal{D})$$

Direct Method (DM)

観測データから事前にYを予測するモデルを学習、それをOPEに用いる

$$\hat{V}_{DM}(\pi; \mathcal{D}) = \frac{1}{n} \sum_{i=1}^n \hat{\mu}(x_i, \pi(x_i); \mathcal{D})$$

目的変数の事前推定モデル

- 目的変数の推定 $\hat{\mu}$ にOPEの精度が大きく依存、biasが大きい
- 一方で、varianceは小さい

$$Y_i(a_i) \approx \hat{\mu}(x_i, a_i; \mathcal{D})$$

Inverse Probability Weighting (IPW)

過去の介入確率で目的変数を割ることでfeedback loopの影響を除去

$$\hat{V}_{IPW}(\pi; \mathcal{D}) = \frac{1}{n} \sum_{i=1}^n Y_i^{obs} \frac{\mathbb{I} \{ \pi(x_i) = a_i \}}{p_b(a_i | x_i)}$$

目的変数の重み付け平均

- 過去の意思決定policy(旧ロジック)による行動選択確率

$$p_b(a_i | x_i)$$

Inverse Probability Weighting (IPW)

過去の介入確率で目的変数を割ることでfeedback loopの影響を除去

$$\hat{V}_{IPW}(\pi; \mathcal{D}) = \frac{1}{n} \sum_{i=1}^n Y_i^{obs} \frac{\mathbb{I}\{\pi(x_i) = a_i\}}{p_b(a_i|x_i)}$$

目的変数の重み付け平均

- 過去の行動選択確率 (p_b) がわかっているならば不偏
- 一方で、varianceは大きい(特にpolicyの乖離が大きい場合)

Doubly Robust (DR)

DMをbaselineとしつつ、目的変数の推定誤差をIPWで補正

$$\hat{V}_{DR}(\pi; \mathcal{D}) = \underbrace{\hat{V}_{DM}(\pi; \mathcal{D})}_{\text{baseline}} + \underbrace{\frac{1}{n} \sum_{i=1}^n \left(Y_i^{obs} - \hat{\mu}(x_i, a_i) \right) \frac{\mathbb{I}\{\pi(x_i)=a_i\}}{p_a(a_i|x_i)}}_{\text{目的変数の推定誤差補正}}$$

- 過去の行動選択確率 (p_b) がわかっているならば不偏
- IPWに比べてvarianceも減少

$$Y_i(a_i) \approx \hat{\mu}(x_i, a_i; \mathcal{D})$$

その他の推定方法

- Self-Normalized IPW [[Swaminathan and Joachims 2015](#)]
- Switch Doubly Robust Estimator [[Wang+ 2017](#)]
- More Robust Doubly Robust Estimator [[Farajtabar+ 2018](#)]
- Hirano-Imbens-Ridder Estimator [[Narita+ 2019](#)]
- REG and EMP [[Kallus & Uehara 2019](#)]
- Double Machine Learning Estimator [[Narita+ 2020](#)]
- Doubly Robust with Shrinkage [[Su+ 2020](#)]

現在までに理論的知見が蓄積、強化学習設定だともっとたくさんある..

その他の推定方法

- Self-Normalized IPW [[Swaminathan and Joachims 2015](#)]
- Switch Doubly Robust Estimator [[Wang+ 2017](#)]
- More Robust Doubly Robust Estimator [[Farajtabar+ 2018](#)]
- Hirano-Imbens-Ridder Estimator [[Narita+ 2019](#)]
- REG and EMP [[Kallus & Uehara 2019](#)]
- Double Machine Learning Estimator [[Narita+ 2020](#)]
- Doubly Robust with Shrinkage [[Su+ 2020](#)]

本当に前進している？

現在までに理論的知見が蓄積、強化学習設定だともっとたくさんある..

Off-Policy Evaluation研究の課題

これらの論文の実験方法は次の2パターンにざっくり分類される

- 人工データを使う(非現実的)

or

- 非公開実データを使う(他者による再現不可能)

OPEの実験が可能な公開(大規模)実データは存在しない..

Our Work

*A Large-scale Open Dataset
for Bandit Algorithms*

Open Bandit Datasetの公開

大規模で、現実的で、再現可能なOPEの実験を実現すべく、

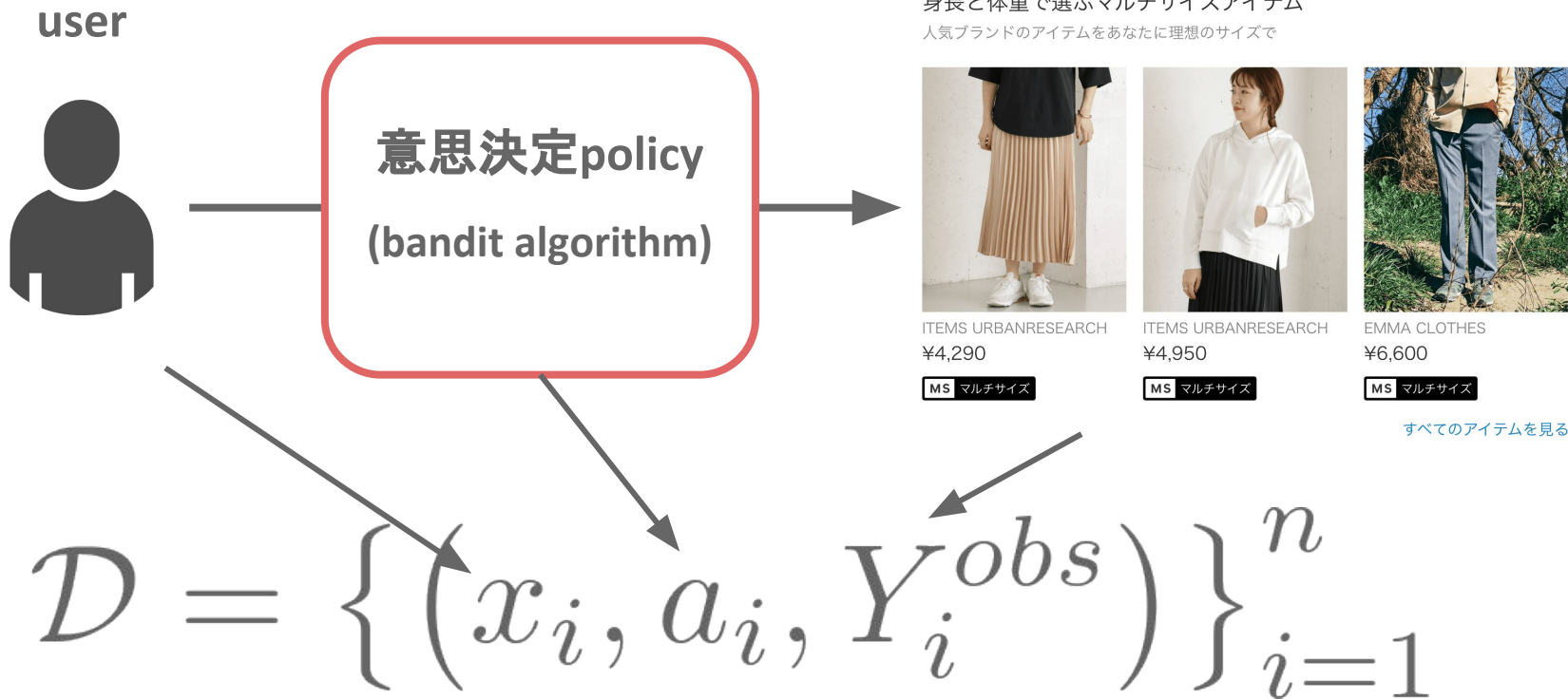
*Open Bandit Dataset*を公開し、研究利用可能に



**ZOZOTOWNのファッションアイテム
推薦枠を用いた大規模実験により収集**

Open Bandit Datasetの公開

recommendation



Open Bandit Datasetの特徴

	a_i		$p_b(a_i \mid x_i)$	Y_i^{obs}	x_i	
タイム スタンプ	アイテムid	推薦位置	行動 選択確率	クリック有無	特徴量	...
2019-11-xx	25	1	0.0002	0	e2500f3f	...
2019-11-xx	32	2	0.043	1	7c414ef7	...
2019-11-xx	11	3	0.167	0	60bd4df9	...
2019-11-xx	40	1	0.0011	0	7c20d9b5	...
...

Open Bandit Datasetの特徴


- 2500万以上のクリックログデータ(追加予定あり)
- 複数の意思決定policyによって収集されている
- データ収集に使われた意思決定policyの実装も公開されている
- データ中における行動の選択確率(p_b)が含まれている

OPEの評価(オフライン評価の正確さの評価)が可能
これまでにそのような公開実データはなし

データセットの使い方: オフライン評価 (OPE) の評価

policy Bが集めたデータ

policy Aが集めたデータ


$$\hat{V}(\pi_A; \mathcal{D}_B) \text{ と } V(\pi_A; \mathcal{D}_A) \text{ を比較}$$


推定量 \hat{V} による policy A の性能の推定値

意思決定 policy B を過去の意思決定 policy とみなして、
OPE 推定量を使って、意思決定 policy A の性能を推定

データセットの使い方: オフライン評価 (OPE)の評価

policy Bが集めたデータ

policy Aが集めたデータ


$$\hat{V}(\pi_A; \mathcal{D}_B) \text{ と } \underline{V(\pi_A; \mathcal{D}_A)} \text{ を比較}$$

意思決定policy Aの真の性能 (onpolicy推定)

$$V(\pi_A; \mathcal{D}_A) = \frac{1}{\mathcal{D}_A} \sum_i Y_i^{obs}$$

データセットの使い方(オフライン評価の正確さ評価)

Table 4: Comparing Relative-Estimation Errors of OPE Estimators (**Random** → **Bernoulli TS**)

Estimators	Campaigns		
	All	Men's	Women's
DM	0.23879 [0.22998, 0.24988]	0.24155 [0.22656, 0.25592]	0.22884 [0.22224, 0.23423]
IPW	0.03477 [0.01147, 0.06592]	0.09806 [0.07485, 0.12151]	0.03252 [0.01708, 0.04912]
SNIPW	0.03381 [0.01005, 0.06662]	0.08153 [0.05677, 0.10592]	0.03179 [0.01562, 0.04825]
DR	0.03487 [0.01094, 0.06784]	0.08528 [0.06186, 0.10876]	0.03224 [0.01605, 0.04843]

論文ではDM, IPW, DRをこの方法で比較している
現在このほかの推定量に関しても性能評価を実施中

ついでにOpen Bandit Pipelineも実装

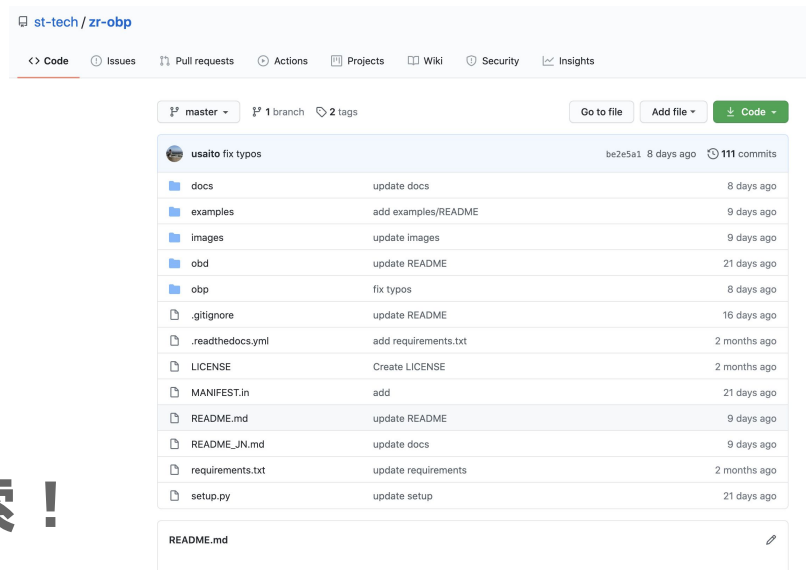
BanditやOPEの実験を容易にかつ統一された設定

で行うための基盤として**Open Bandit Pipeline**を実装



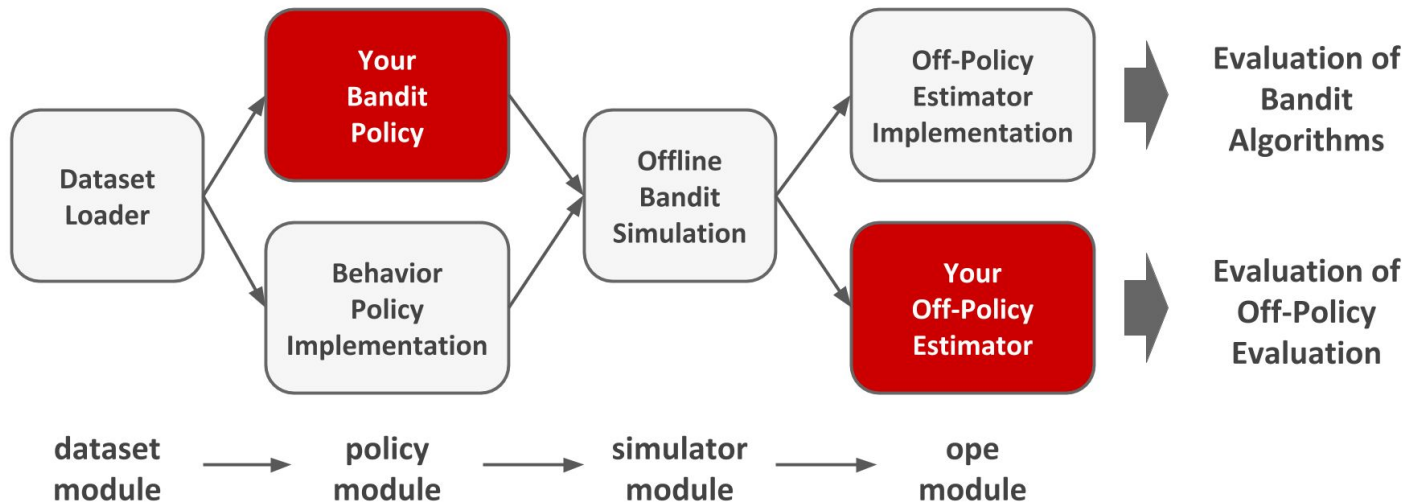
OPEN
BANDIT
PIPELINE™

「**zr-obp**」で検索！



Open Bandit Pipelineの構成

- 意思決定policyの実装
- オフライン評価 (OPE)



- データの読み込み
- 人工データの生成
- 意思決定policyの動作

Open Bandit Pipelineの活用方法

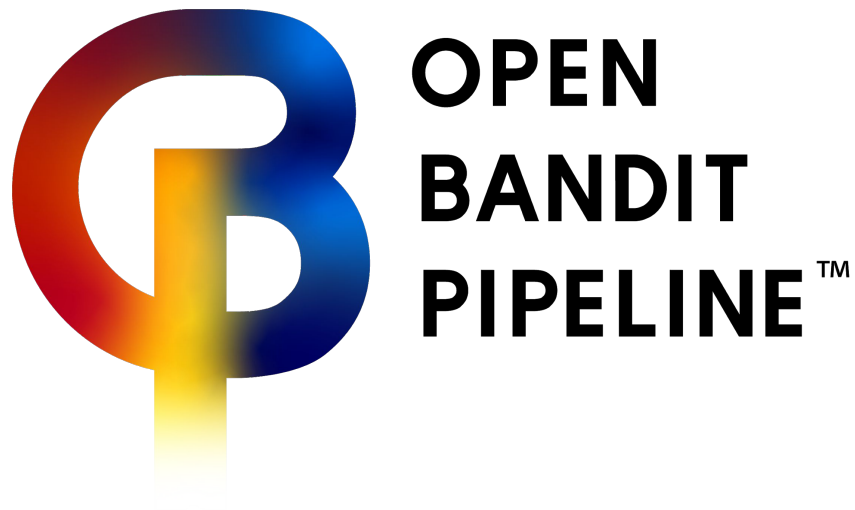
- 研究者

独自のpolicyやOPE推定量を実装に集中することですでに実装されたbaselineとの性能比較を行うことが可能

- 実践者(エンジニア、データサイエンティスト)

自社データによる意思決定policyのオフライン評価をすでに実装された標準的な推定量を使って行うことが可能

Open Bandit Pipelineの構成



使い方をquickstart exampleで見てみましょう

Open Bandit Pipelineの特徴

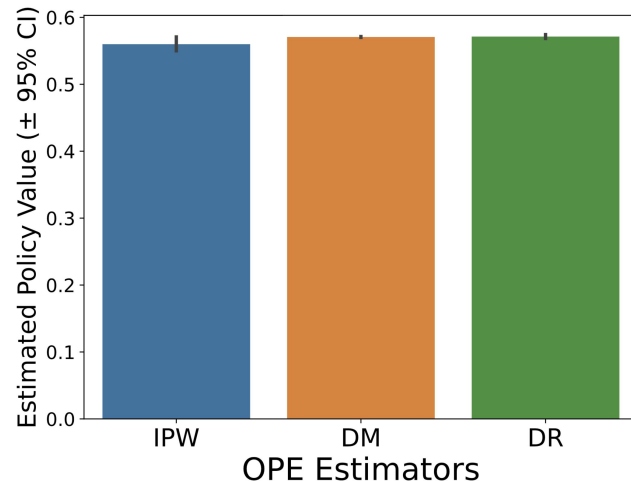
```
# a case for implementing OPE of the BernoulliTS policy using log data generated by the Random policy
from obp.dataset import OpenBanditDataset
from obp.policy import BernoulliTS
from obp.simulator import run_bandit_simulation
from obp.ope import OffPolicyEvaluation, ReplayMethod

# (1) Data loading and preprocessing
dataset = OpenBanditDataset(behavior_policy='random', campaign='women')
bandit_feedback = dataset.obtain_batch_bandit_feedback()

# (2) Offline Bandit Simulation
counterfactual_policy = BernoulliTS(n_actions=dataset.n_actions, len_list=dataset.len_list)
selected_actions = run_bandit_simulation(bandit_feedback=bandit_feedback, policy=counterfactual_policy)

# (3) Off-Policy Evaluation
ope = OffPolicyEvaluation(bandit_feedback=bandit_feedback, ope_estimators=[ReplayMethod()])
estimated_policy_value = ope.estimate_policy_values(selected_actions=selected_actions)

# estimated performance of BernoulliTS relative to the ground-truth performance of Random
relative_policy_value_of_bernoulli_ts = estimated_policy_value['rm'] / bandit_feedback['reward'].mean()
print(relative_policy_value_of_bernoulli_ts) # 1.120574...
```



お見せしたように数行のcodeで
新意思決定policyのオフライン評価が可能

Open Bandit Pipelineの特徴

obp
latest

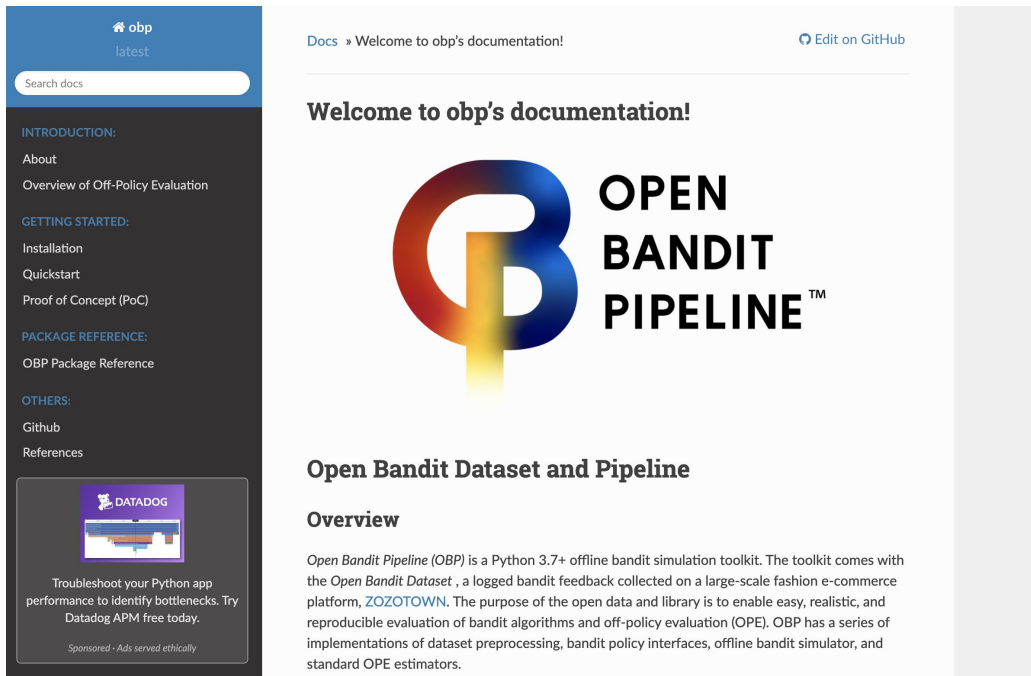
Search docs

INTRODUCTION:
About
Overview of Off-Policy Evaluation

GETTING STARTED:
Installation
Quickstart
Proof of Concept (PoC)

PACKAGE REFERENCE:
OBP Package Reference

OTHERS:
Github
References




Troubleshoot your Python app performance to identify bottlenecks. Try Datadog APM free today.

Sponsored - Ads served ethically

Docs » Welcome to obp's documentation! [Edit on GitHub](#)

Welcome to obp's documentation!



Open Bandit Dataset and Pipeline

Overview

Open Bandit Pipeline (OBP) is a Python 3.7+ offline bandit simulation toolkit. The toolkit comes with the *Open Bandit Dataset*, a logged bandit feedback collected on a large-scale fashion e-commerce platform, [ZOZOTOWN](#). The purpose of the open data and library is to enable easy, realistic, and reproducible evaluation of bandit algorithms and off-policy evaluation (OPE). OBP has a series of implementations of dataset preprocessing, bandit policy interfaces, offline bandit simulator, and standard OPE estimators.

ドキュメントもちゃんとあります

トップ国際会議のWorkshopで発表

Workshop on Real World Experiment Design and Active Learning at ICML 2020

Workshop on Real World Experiment Design and Active Learning at ICML 2020

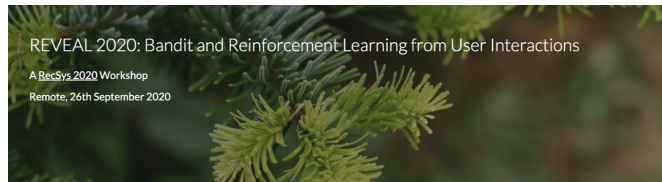
Virtual workshop, 18 July 2020 @ICML

This workshop aims to bring together researchers from academia and industry to discuss major challenges, outline recent advances, and highlight future directions pertaining to novel and existing large-scale real-world experiment design and active learning problems. We aim to highlight new and emerging research opportunities for the ML community that arise from the evolving needs to make experiment design and active learning procedures that are theoretically and practically relevant for realistic applications. Progress in this area has the potential to provide immense benefits in using experiment design and active learning algorithms in emerging high impact applications, such as material design, computational biology, algorithm configuration, AutoML, crowdsourcing, citizen science, robotics, and more.

Remark: In light of the COVID-19 situation, the workshop will be held virtually. For more information, please see the ICML conference [website](#).

Talks

REVEAL Workshop at RecSys 2020 (selected as oral presentation)



Important dates:

- Submission deadline: July 29th
- Author notification: August 21st
- Camera-ready deadline: September 4th
- Workshop: September 26th

State-of-the-art recommender systems are notoriously hard to design and improve upon, due to their interactive and dynamic nature, since they involve a multi-step decision-making process, where a stream of interactions occurs between the user and the system. Leveraging reward signals from these interactions and creating a scalable and performant recommendation inference model is a key challenge. Traditionally, to make the problem tractable, the interactions are often viewed as independent, but in order to improve recommender systems further, the models will need to take into account the delayed effects of each recommendation and start reasoning/planning for longer-term user satisfaction. To this end, our workshop invites contributions that enable recommender systems to adapt effectively to diverse forms of user feedback and to optimize the quality of each user's long-term experience.

Due to their interactive nature, recommender systems are also notoriously hard to evaluate. When evaluating their systems, practitioners often observe significant differences between a new algorithm's offline and online results, and therefore tend to mostly rely on online methods, such as A/B testing. This is unfortunate, since online evaluation is not always possible and often expensive. Offline evaluation, on the other hand, provides a scalable way of comparing recommender systems and enables the participation of academic research in an industry-relevant problem.

同分野の研究者からすでに興味関心を集める

Future Work

- ポジションバイアスは無さそうだが、隣に並んでいるアイテムの影響は無視できないのでリスト構造を考慮した推定量 (slate recommendation の設定)を実装、性能評価したい
- よりadvancedな推定量についてのベンチマーク作成
- その他、データの追加、細かい機能の追加などは継続して行う

まとめ

- 予測精度よりも意思決定の性能を評価しよう (OPE)
- OPEの理論研究はとても進んでいるものの、
実験は非現実的もしくは再現不可能な形で行われている
- Open Bandit Dataset & Pipelineによって、
特にOPEの身のある実験評価に広く貢献(したい..)

Thank you for listening!

- 論文 (arXiv): <https://arxiv.org/abs/2008.07146>
- github: <https://github.com/st-tech/zr-obp>
- dataset: <https://research.zozo.com/data.html>
- blog: <https://techblog.zozo.com/entry/openbanditproject>