Modern Recommendation for Advanced Practitioners.

Part I. Recommendation via maximizing the likelihood of reward.

Presenters: **Flavian Vasile, Amine Benhalloum**

Criteo AI Lab

September 11, 2019

criteo

About the Speakers:

Flavian Vasile

- **Flavian Vasile** is part of the Criteo AI Lab where he works as the *ML Recommendations Solutions Architect*, with his main focus being on the development of Deep Learning-based Recommendation Systems and on introducing aspects of Causal Inference to Recommendation.
- Before joining Criteo, he worked as a Senior Researcher in the Twitter Advertising Science and as part of the Yahoo! Research Lab where he mostly focused on Content Understanding problems.
- His current research interests include *Deep Sequential Models for Recommendation* and understanding *Causality in Recommendation*.
- Among his recent research publications, the work on *Causal Embeddings for Recommendation* received the *best paper award at RecSys 2018* and he is the co-organizer of the *Workshop on Offline Evaluation for Recommender Systems* at *RecSys 2019*.

criteo

About the Speakers:

Amine Benhalloum

- **Amine Benhalloum** is a Senior Machine Learning Engineer at Criteo, working on building large scale representation learning and retrieval systems for recommendation, applying Deep learning to personalize billions of daily display ads, reaching billions of users and connecting them with millions of products.
- His areas of expertise are: large scale machine learning, natural language processing, information retrieval and data intensive systems.
- Before joining Criteo, Amine worked on a variety of topics ranging from Natural Language processing to fraud detection. He holds a master's degree in Applied Mathematics.

criteo

Collaborators:

Special thanks to our colleagues:

- **That co-authored this course:** David Rohde, Martin Bompaire, Olivier Jeunen
- **That developed the methods behind it:** Stephen Bonner, Dmytro Mykhaylov, Elvis Dohmatob, Louis Faury, Ugo Tanielian

criteo

criteo

AI Lab

http://cail.criteo.com

Who we are

Criteo is a leading ad-tech company, connecting more than a billion users to products they might be interested in. With more than 120TB of extra data per day, we need to seriously weigh the algorithmic choices we make on a daily basis...

The Criteo AI Lab is an industrial R&D Lab focusing on Computational Advertising. Our research focuses on Machine Learning, Recommender Systems, Reinforcement Learning, and much more!

criteo

About the Course:

- **Part I. Recommendation via maximizing likelihood** approaches
  - I.1. Classic vs. Modern: Recommendation as autocomplete vs. recommendation as intervention policy
  - I.2. ERM and Likelihood models for optimal effect recommendations
  - I.3. Shortcomings of ERM/likelihood-based models for recommendation
- **Part II. Recommendation as policy learning** approaches
  - II.1. Policy Learning: Concepts and Notations
  - II.2. Fixing ERM using Policy Learning
    - *Fixing Covariate Shift: From ERM to Counterfactual Risk Minimization*
    - *Fixing Optimizer's Curse: From ERM to Distributional Robust Optimization*
    - Reco-specific Policy Learning methods: Using Organic Feedback
  - II.3. Recap and Conclusions

criteo

Getting started with RecoGym and Google Colaboratory

- Open your favourite browser, and go to the course repository: `https://github.com/bamine/recsys-summer-school`
- You will find the notebooks and the slides for the course
- You can access the notebooks directly on Google Collab in the following manner:
  - `https://colab.research.google.com/github/bamine/recsys-summer-school/blob/master/notebooks/0%20Getting%20Started.ipynb`
- Alternatively, clone the repository and run the notebooks locally.
- . . . you're all set!

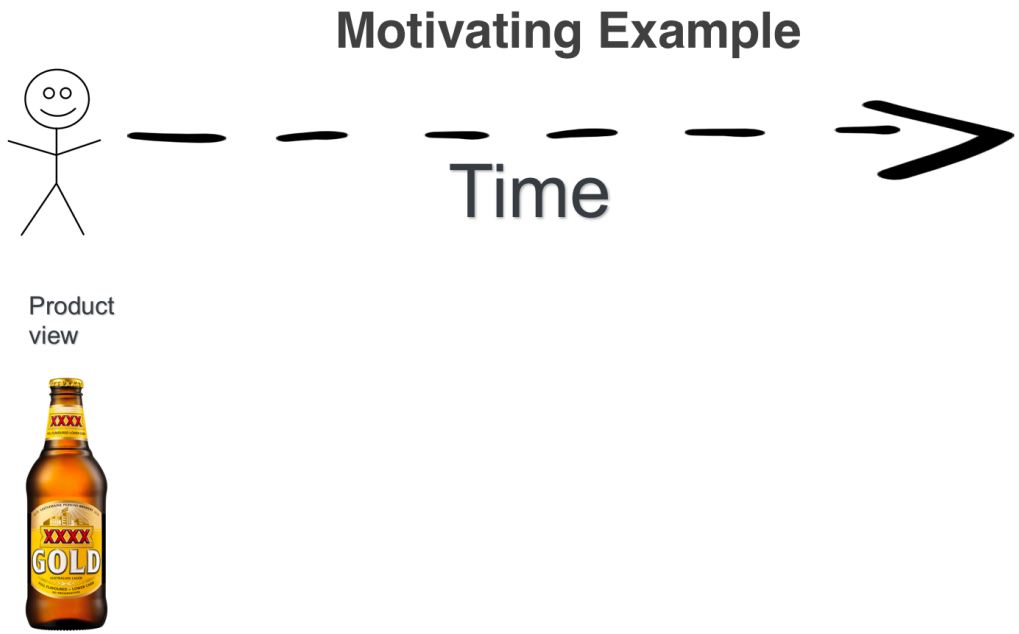
criteo

Section 1. Classic vs. Modern Recommendation

What makes a Recommender System modern?

criteo

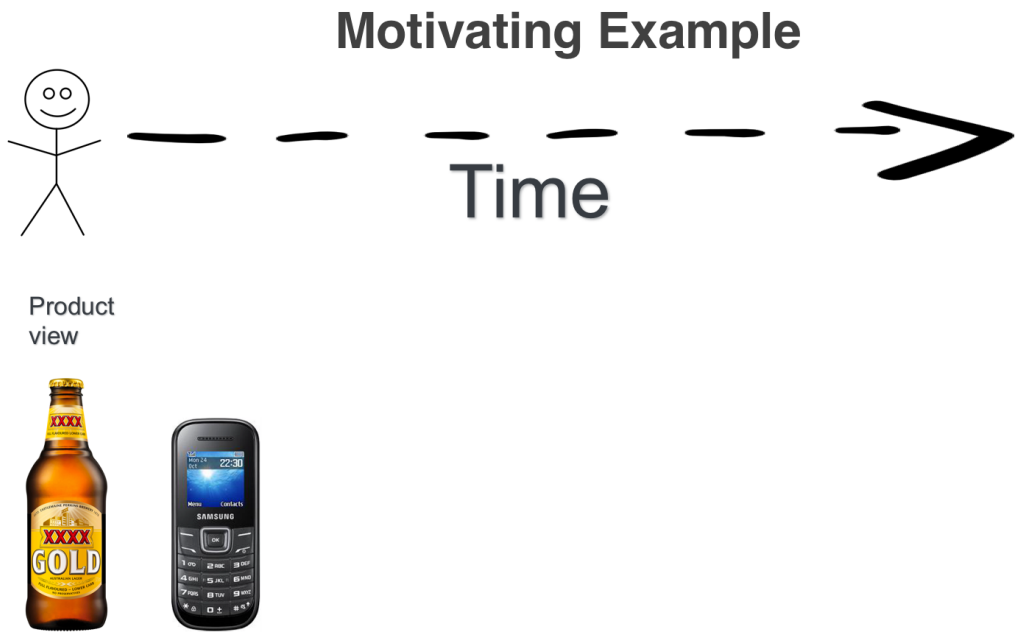
Classic Reco



$$P(V_1 = \text{beer})$$

criteo

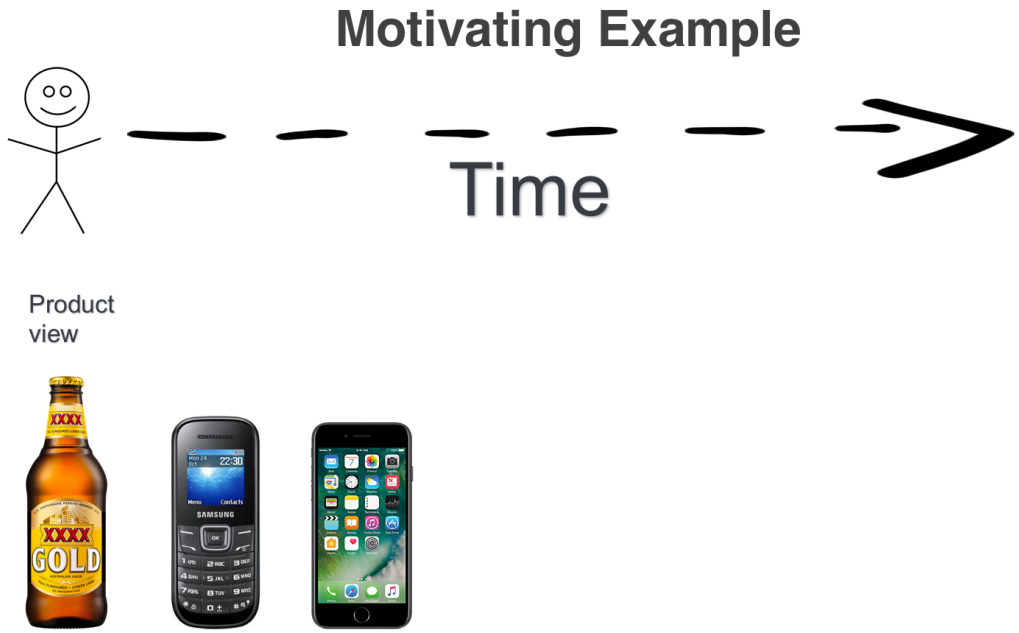
Classic Reco



$$P(V_2 = \text{phone A} | V_1 = \text{beer})$$

criteo

Classic Reco



$$P(V_3 = \text{phone B} | V_1 = \text{beer}, V_2 = \text{phone A})$$

criteo

Next Item Prediction, hold out V_3

$$
\begin{aligned}
P(V_3 = \text{phone A} | V_1 = \text{beer}, V_2 = \text{phone A}) &= 0.27 \\
P(V_3 = \text{phone B} | V_1 = \text{beer}, V_2 = \text{phone A}) &= 0.24 \\
P(V_3 = \text{rice} | V_1 = \text{beer}, V_2 = \text{phone A}) &= 0.10 \\
P(V_3 = \text{couscous} | V_1 = \text{beer}, V_2 = \text{phone A}) &= 0.09 \\
P(V_3 = \text{beer} | V_1 = \text{beer}, V_2 = \text{phone A}) &= 0.30
\end{aligned}
$$

What actually happened?

criteo

Next Item Prediction, hold out V_3

$$
\begin{aligned}
P(V_3 = \text{phone A} | V_1 = \text{beer}, V_2 = \text{phone A}) &= 0.27 \\
P(V_3 = \text{phone B} | V_1 = \text{beer}, V_2 = \text{phone A}) &= 0.24 \\
P(V_3 = \text{rice} | V_1 = \text{beer}, V_2 = \text{phone A}) &= 0.10 \\
P(V_3 = \text{couscous} | V_1 = \text{beer}, V_2 = \text{phone A}) &= 0.09 \\
P(V_3 = \text{beer} | V_1 = \text{beer}, V_2 = \text{phone A}) &= 0.30
\end{aligned}
$$

What actually happened? $V_3 = \text{phone B}$

criteo

Next Item Prediction, hold out V_3

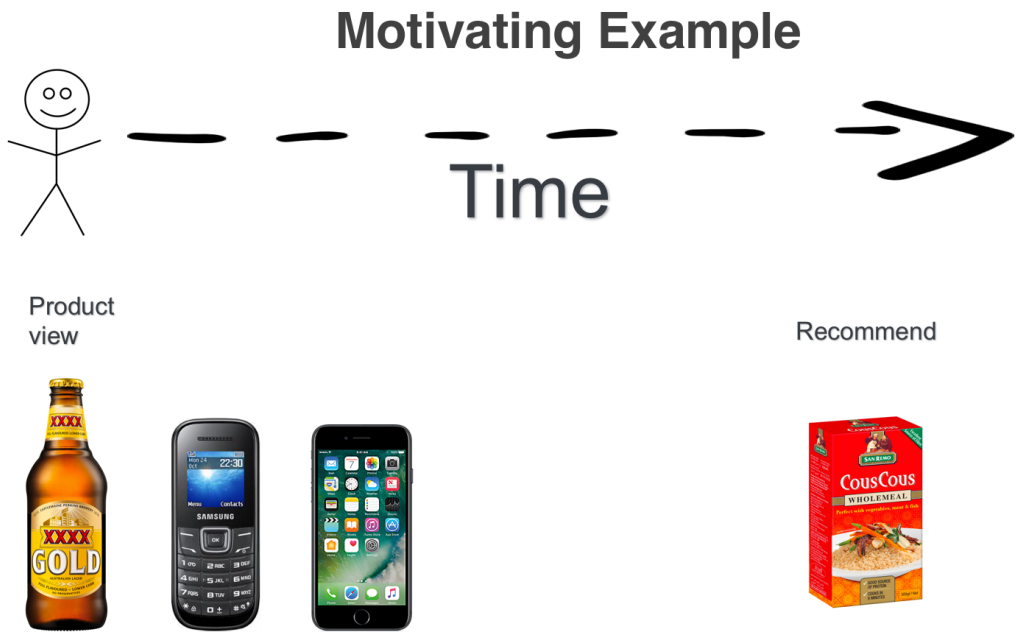
$$
\begin{aligned}
P(V_3 = \text{phone A} | V_1 = \text{beer}, V_2 = \text{phone A}) &= 0.27 \\
P(V_3 = \text{phone B} | V_1 = \text{beer}, V_2 = \text{phone A}) &= 0.24 \\
P(V_3 = \text{rice} | V_1 = \text{beer}, V_2 = \text{phone A}) &= 0.10 \\
P(V_3 = \text{couscous} | V_1 = \text{beer}, V_2 = \text{phone A}) &= 0.09 \\
P(V_3 = \text{beer} | V_1 = \text{beer}, V_2 = \text{phone A}) &= 0.30
\end{aligned}
$$

What actually happened? $V_3 = \text{phone B}$

Our model assigned: $P(V_3 = \text{phone B} | V_1 = \text{beer}, V_2 = \text{phone A}) = 0.24$

criteo

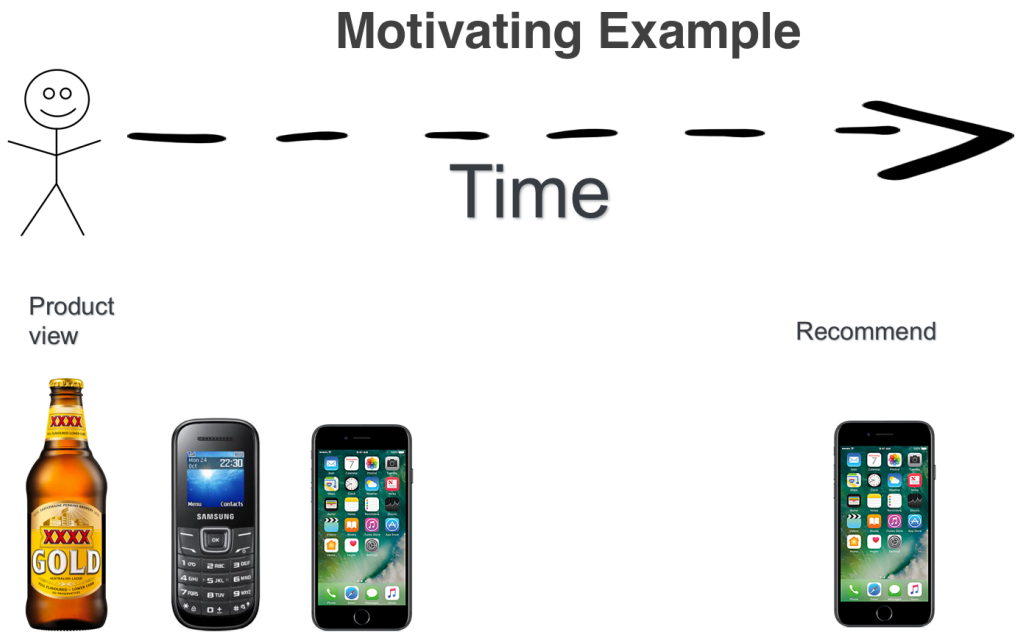
The Real Reco Problem



$$P(C_4 = 1 | A_4 = \text{couscous}, V_1 = \text{beer}, V_2 = \text{phone A}, V_3 = \text{phone B})$$

criteo

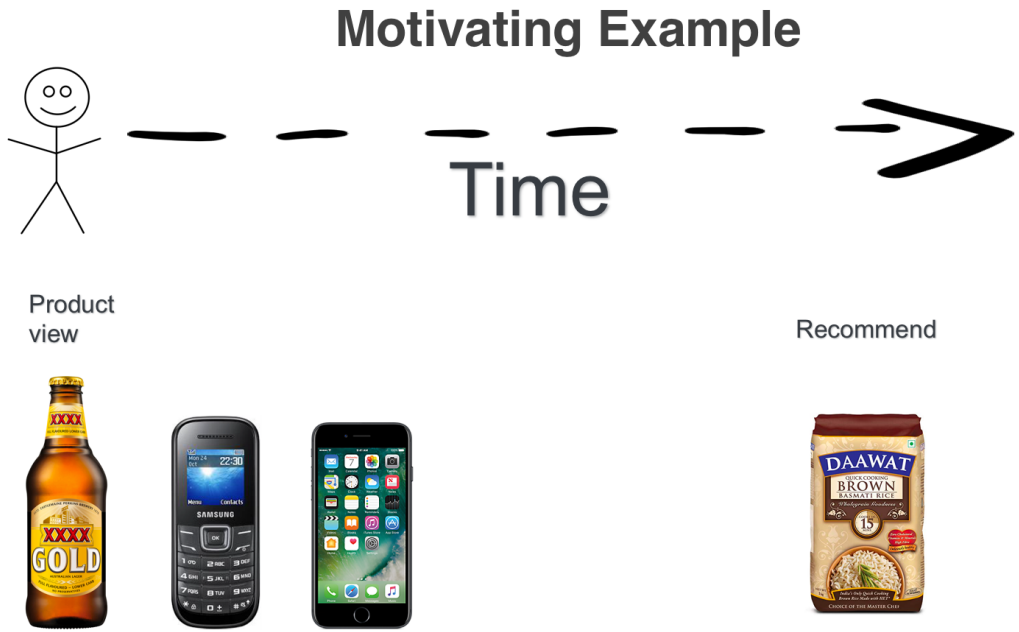
The Real Reco Problem



$$P(C_4 = 1 | A_4 = \text{phone B}, V_1 = \text{beer}, V_2 = \text{phone A}, V_3 = \text{phone B})$$

criteo

The Real Reco Problem



$$P(C_4 = 1 | A_4 = \text{rice}, V_1 = \text{beer}, V_2 = \text{phone A}, V_3 = \text{phone B})$$

criteo

What we have, and what we want:

Our model:

$$
\begin{aligned}
P(V_4 = \text{phone A} | V_1 = \text{beer}, V_2 = \text{phone A}, V_3 = \text{phone A}) &= 0.27 \\
P(V_4 = \text{phone B} | V_1 = \text{beer}, V_2 = \text{phone A}, V_3 = \text{phone A}) &= 0.28 \\
P(V_4 = \text{rice} | V_1 = \text{beer}, V_2 = \text{phone A}, V_3 = \text{phone A}) &= .12 \\
P(V_4 = \text{couscous} | V_1 = \text{beer}, V_2 = \text{phone A}, V_3 = \text{phone A}) &= .14 \\
P(V_4 = \text{beer} | V_1 = \text{beer}, V_2 = \text{phone A}, V_3 = \text{phone A}) &= 0.19
\end{aligned}
$$

Which is a vector of size P that sums to 1.

We want:

criteo

What we have, and what we want:

Our model:

$$P(V_4 = \text{phone A} | V_1 = \text{beer}, V_2 = \text{phone A}, V_3 = \text{phone A}) = 0.27$$
$$P(V_4 = \text{phone B} | V_1 = \text{beer}, V_2 = \text{phone A}, V_3 = \text{phone A}) = 0.28$$
$$P(V_4 = \text{rice} | V_1 = \text{beer}, V_2 = \text{phone A}, V_3 = \text{phone A}) = .12$$
$$P(V_4 = \text{couscous} | V_1 = \text{beer}, V_2 = \text{phone A}, V_3 = \text{phone A}) = .14$$
$$P(V_4 = \text{beer} | V_1 = \text{beer}, V_2 = \text{phone A}, V_3 = \text{phone A}) = 0.19$$

Which is a vector of size P that sums to 1.

We want:

$$P(C_4 = 1 | A_4 = \text{phone A}, V_1 = \text{beer}, V_2 = \text{phone A}, V_3 = \text{phone A}) = ?$$
$$P(C_4 = 1 | A_4 = \text{phone B}, V_1 = \text{beer}, V_2 = \text{phone A}, V_3 = \text{phone A}) = ?$$
$$P(C_4 = 1 | A_4 = \text{rice}, V_1 = \text{beer}, V_2 = \text{phone A}, V_3 = \text{phone A}) = ?$$
$$P(C_4 = 1 | A_4 = \text{couscous}, V_1 = \text{beer}, V_2 = \text{phone A}, V_3 = \text{phone A}) = ?$$
$$P(C_4 = 1 | A_4 = \text{beer}, V_1 = \text{beer}, V_2 = \text{phone A}, V_3 = \text{phone A}) = ?$$

Which is a vector of size P where each entry is between 0 and 1.

criteo

An implicit assumption:

Our model predicts:

$$
\begin{aligned}
P(V_4 = \text{phone A} | V_1 = \text{beer}, V_2 = \text{phone A}, V_3 = \text{phone A}) &= 0.27 \\
P(V_4 = \text{phone B} | V_1 = \text{beer}, V_2 = \text{phone A}, V_3 = \text{phone A}) &= 0.28 \\
P(V_4 = \text{rice} | V_1 = \text{beer}, V_2 = \text{phone A}, V_3 = \text{phone A}) &= .12 \\
P(V_4 = \text{couscous} | V_1 = \text{beer}, V_2 = \text{phone A}, V_3 = \text{phone A}) &= .14 \\
P(V_4 = \text{beer} | V_1 = \text{beer}, V_2 = \text{phone A}, V_3 = \text{phone A}) &= 0.19
\end{aligned}
$$

So let's recommend phone B, it has the highest next item probability.

criteo

An implicit assumption:

If:

$$P(V_n = a | V_1 = v_1 .. V_{n-1} = v_{n-1}) > P(V_n = b | V_1 = v_1 .. V_{n-1} = v_{n-1})$$

Assume:

$$\begin{aligned} &P(C_n = 1 | A_n = a, V_1 = v_1 .. V_{n-1} = v_{n-1}) \\ &\qquad > P(C_n = 1 | A_n = b, V_1 = v_1 .. V_{n-1} = v_{n-1}) \end{aligned}$$

Vast amounts of academic Reco work assumes this implicitly!

criteo

Types of approaches to Recommendation

- **Classical Approach: Recommendation as auto-complete**:
  - Typically leverage *organic* user behaviour information (e.g. item views, page visits)
  - Frame the problem either as *missing link prediction/MF* problem, either as a *next event prediction, sequence prediction* problem
- **Modern approach: Recommendation as intervention policy**:
  - Typically leverage *bandit* user behaviour information (e.g. ad clicks)
  - Frame the problem either as *click likelihood* problem, either as a *contextual bandit/policy learning* problem

criteo

1.1. Advantages of the classical approach

criteo

Academia

- An already established framework
- Standard datasets and metrics
- Easier to publish and compare methods !

criteo

Real-world

- Data arises naturally from the use of the application (before any recommendation system in place)
- Methods have standard efficient and well tested implementations
- Very good initial system !

criteo

1.2. Limitations of the classical approach

What is wrong with the classical formulation?

We are operating under the assumption that the best recommendation policy is in some sense the **optimal auto-complete of natural user behavior**

criteo

Solving the wrong problem

From the point of view of business metrics, **learning to autocomplete behavior is a great initial recommendation policy**, especially when no prior user feedback is available.

However, after a first golden age, where all offline improvements turn into positive A/B tests, the *naive recommendation* optimization objective and the business objective will start to diverge.

Solving the wrong problem: Are the classic datasets logs of RecSys?

- MovieLens: no, explicit feedback of movie ratings
- Netflix Prize: no, explicit feedback of movie ratings
- Yoochoose (RecSys competition '15): no, implicit session based behavior
- 30 Music: no, implicit session based behavior
- Yahoo News Feed Dataset: yes!
- Criteo Dataset for counterfactual evaluation of RecSys algorithms: yes!

criteo

Solving the wrong problem: Are the classic datasets logs of RecSys?

- MovieLens: no, explicit feedback of movie ratings
- Netflix Prize: no, explicit feedback of movie ratings
- Yoochoose (RecSys competition '15): no, implicit session based behavior
- 30 Music: no, implicit session based behavior
- Yahoo News Feed Dataset: yes!
- Criteo Dataset for counterfactual evaluation of RecSys algorithms: yes!

The Criteo Dataset shows a log of recommendations and if they were successful in getting users to click.

criteo

Solving the wrong problem: Do the classical offline metrics evaluate the quality of recommendation?

Classical offline metrics:

- Recall@K, Precision@K, HR@K: How often is an item in the top k - no, evaluates next item prediction
- DCG: Are we assigning high score to an item - no, evaluates next item prediction

Online metrics and their offline simulations:

- AB Test: i.e. run a randomized control trial live - yes, but expensive + the academic literature has no access to this
- Inverse Propensity Score estimate of click through rate: - to be explained later. - yes! (although it is often noisy)

criteo

Solving the wrong problem: Do the classical offline metrics evaluate the quality of recommendation?

Bottom line: If the dataset does not contain a log of recommendations and if they were successful, you cannot really compute metrics of the recommendation quality!

criteo

1.3. Modern Reco: Solving the problems of classical approach

criteo

Aligning the Recommendation objective with the business objectives

- Of course, we could start incorporating user feedback that we collected while running the initial recommendation policies
- We should be able to continue bringing improvements using feedback that is now aligned with our business metrics (ad CTR, post click sales, dwell time, number of videos watched, ...)

Better offline evaluation

criteo

Offline evaluation that predicts an AB test result

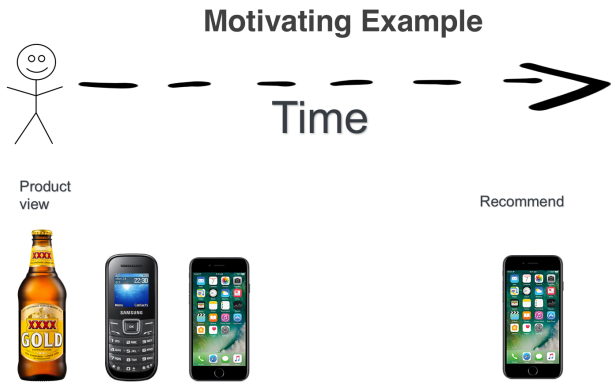
Let's follow the previous notation (A action/recommended item, V organic item view/interaction) and denote π the recommendation policy (that assign probabilities to items conditionally on user past).

Offline evaluation that predicts an AB test result

Let's follow the previous notation (A action/recommended item, V organic item view/interaction) and denote π the recommendation policy (that assign probabilities to items conditionally on user past). Imagine we have a new recommendation policy $\pi_t(A_n = a_n | V_1 = v_1, ..., V_n = v_{n-1})$, can we predict how well it will perform if we deploy it? We collected logs from a different policy: $\pi_0(A_n = a_n | V_1 = v_1, ..., V_n = v_{n-1})$

Let's examine some hypothetical logs...

Offline evaluation IPS



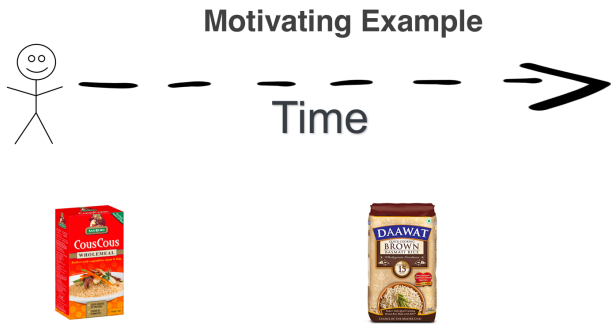
Imagine the user clicks

$$\pi_t(A_4 = \text{phone B} | V_1 = \text{beer}, V_2 = \text{phone A}, V_3 = \text{phone B}) = 1$$
$$\pi_0(A_4 = \text{phone B} | V_1 = \text{beer}, V_2 = \text{phone A}, V_3 = \text{phone B}) = 0.8$$

The new policy will recommend "phone B" the $\frac{1}{0.8} = 1.25\times$ as often as the old policy.

criteo

Offline evaluation IPS



criteo

Imagine the user clicks

$$\pi_t(A_2 = \text{rice} | V_1 = \text{couscous}) = 1$$
$$\pi_0(A_2 = \text{rice} | V_1 = \text{couscous}) = 0.01$$

The new policy will recommend "rice" the $\frac{1}{0.01} = 100\times$ as often as the old policy.

criteo

Offline evaluation IPS

Let's denote c_n the feedback on the n^{th} recommendation, i.e. 1 if the recommendation got clicked else 0.

criteo

Offline evaluation IPS

Let's denote c_n the feedback on the n^{th} recommendation, i.e. 1 if the recommendation got clicked else 0.
Let's re-weight each click by *how much more likely the recommendation will appear under the new policy π_t compared the logging policy π_0)*

Offline evaluation IPS

Let's denote c_n the feedback on the n^{th} recommendation, i.e. 1 if the recommendation got clicked else 0.
Let's re-weight each click by *how much more likely the recommendation will appear under the new policy π_t compared the logging policy π_0)*

$$\text{CTR estimate} = \mathbb{E}_{\pi_0}\left(\frac{c \cdot \pi_t(a|X)}{\pi_0(a|X)}\right)$$

$$\approx \frac{1}{N}\sum_{n}^{N} \frac{c_n \pi_t(A_n = a_n | V_1 = v_1 .. V_n = v_n)}{\pi_0(A_n = a_n | V_1 = v_1 .. V_n = v_n)}$$

criteo

Offline evaluation

This estimator is unbiased

$$\text{CTR estimate} = \mathbb{E}_{\pi_0}\left(\frac{c \cdot \pi_t(a|X)}{\pi_0(a|X)}\right) = \mathbb{E}_{\pi_t}(c)$$

criteo

Offline evaluation IPS

We only look at the clicks i.e. $c_n = 1$ (otherwise the contribution is 0).

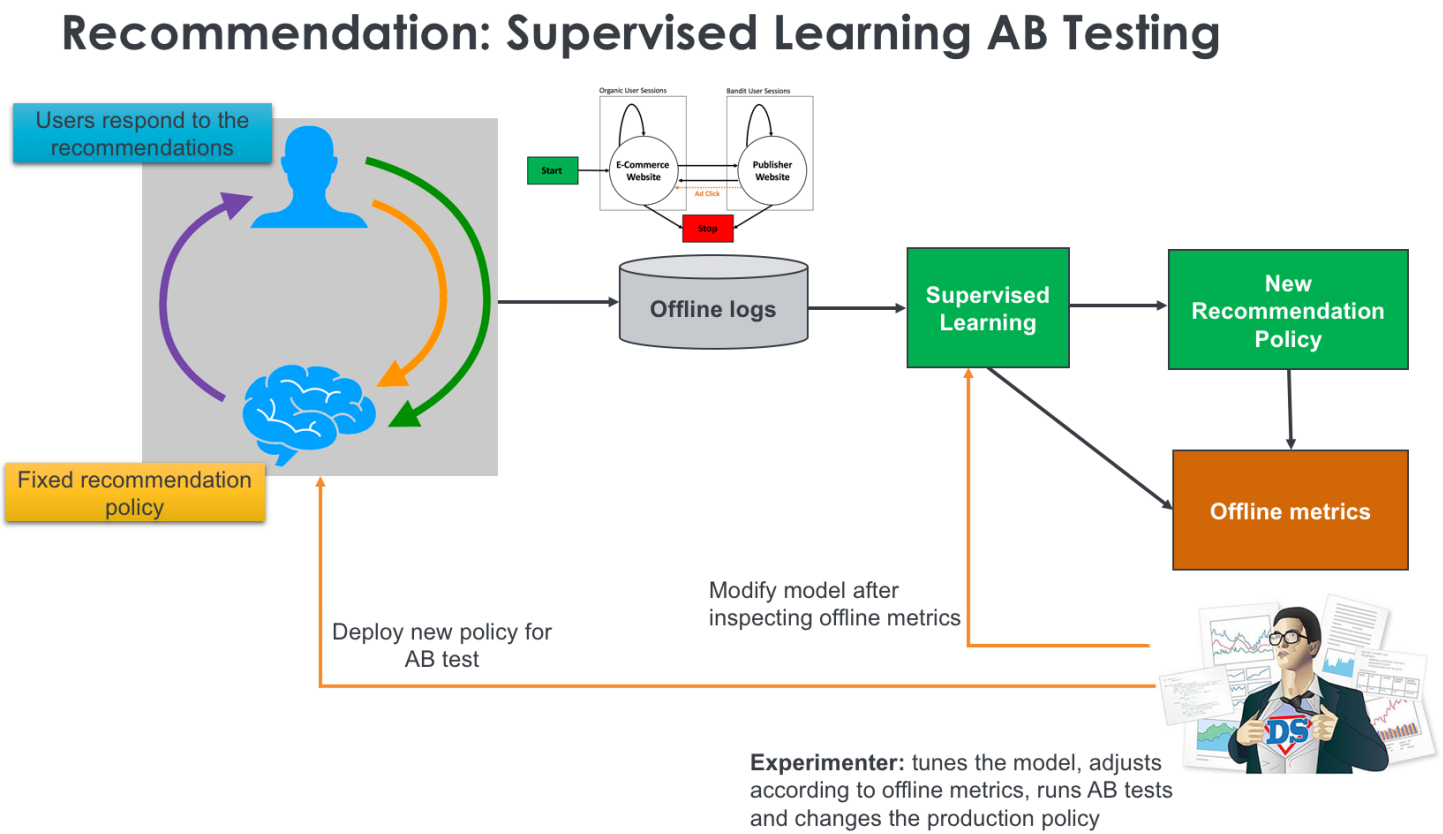
When the new policy differs markedly from the old, the weights become very high (to compensate for the fact that these are rare examples in your sample). These rare high values contribute a lot to the variance of the estimator.

IPS actually attempts to answer a counterfactual question (what would've happened ?), but it often has less than spectacular results...

Improvement of real-world Reco Systems

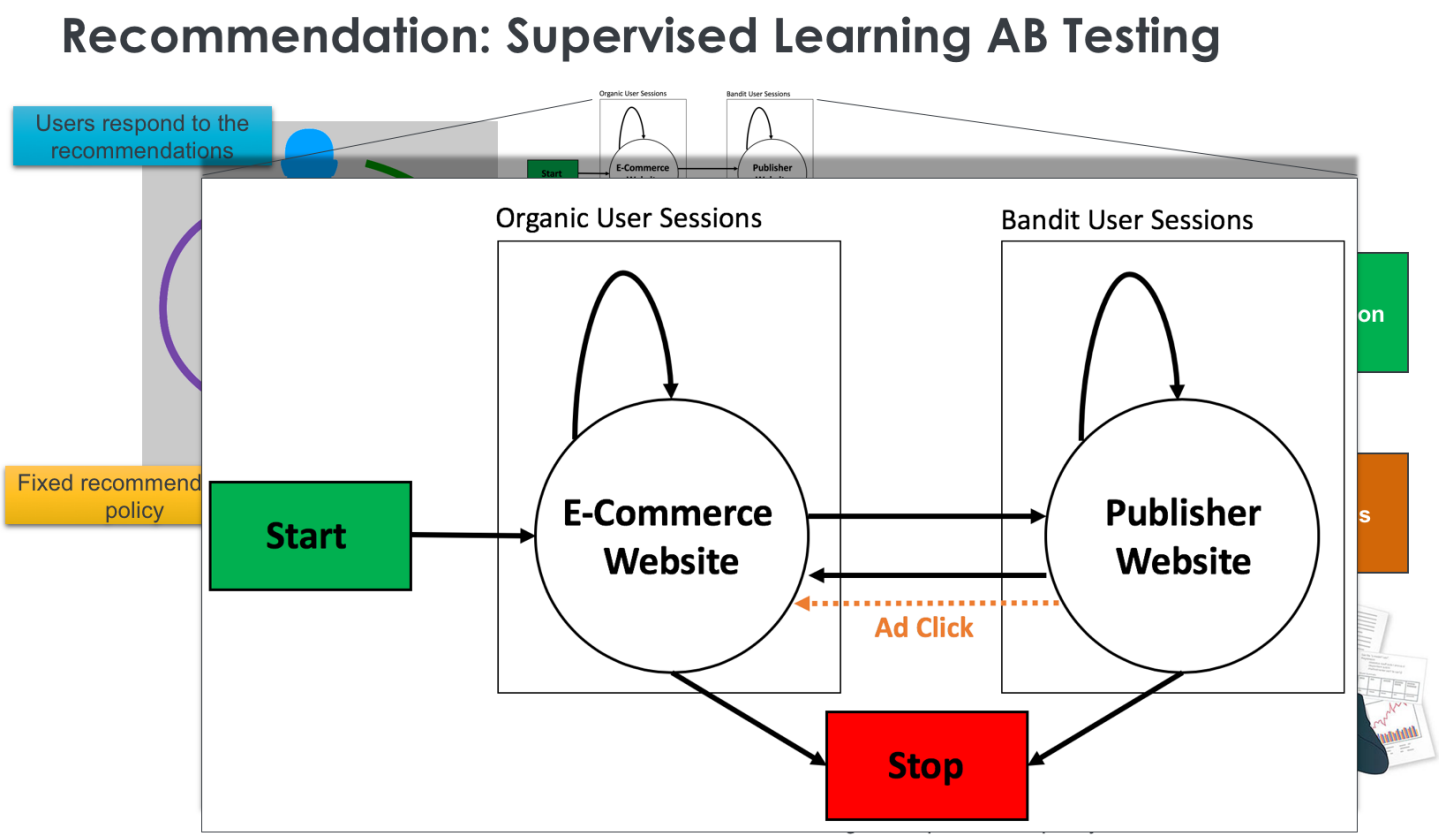
criteo

Recommendation as Supervised Learning and AB Testing



criteo

Recommendation as Supervised Learning and AB Testing



Let's take a step back...

How are we improving large-scale Recommender Systems in the Real World

- Learn a supervised model from past user activity
- Evaluate offline and decide whether to A/B test
- A/B test
- If positive and scalable, roll-out
- If not, try to understand what happened and try to create a better model of the world using the same past data
- Repeat

criteo

The relationship between Reinforcement Learning and Recommendation

- We are doing Reinforcement Learning by hand!
- Furthermore, we are trying to do RL using the Supervised Learning Framework
- Standard test data sets do not let us explore this aspect of recommender systems

.. but how do you evaluate offline a reinforcement learning algorithm?

criteo

1.4. RecoGym: An offline simulator for Recommendation

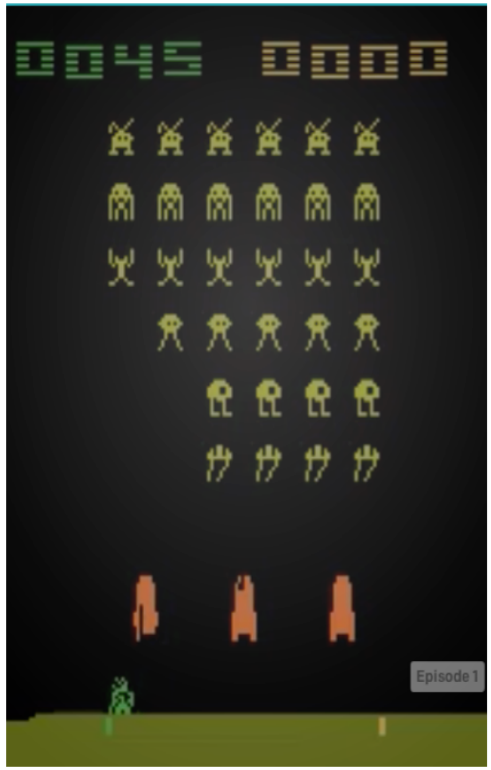
criteo

Open AI Gym

Problem: The reinforcement learning community lacked a common set of tools to allow comparison of reinforcement learning algorithms.

Open AI Gym: 2016 (Brockman et al.): Software standard (Python api) that allows comparison of the performance of reinforcement learning algorithms.

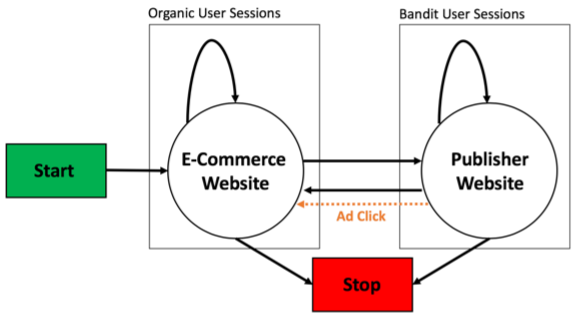
Core idea: environments (the problem) and agents (the RL algorithm) interact.



criteo

Introducing RecoGym

RecoGym: Simulates User Behavior (both Organic and Bandit)



- Allows online evaluation of new recommendation policies in a simulated environment
- Holistic view of user (organic and bandit) provides a framework for categorizing recommendation algorithms
- Key feature: adjustable parameter that changes the effectiveness of “Pure Organic” or next item prediction algorithms.

Algorithm 1: A simple Simulator

Input : $S \in \mathcal{R}^{3\times3}$ transition matrix between organic and bandit, $\Gamma \in \mathcal{R}^{P\times K}$ organic embeddings, μ_Γ organic popularity $\boldsymbol{\beta} \in \mathcal{R}^{P\times K}$ bandit embeddings, μ_β non-personalised ctr contribution $f(\cdot)$ montonic increasing function accounting for ad fatigue $\boldsymbol{m}$, U number of users, P number of products.

Output: Sequence of organic and bandit events

for $u \in 1..U$ **do**
  $t \leftarrow 0$
  $z_{u,0} \leftarrow$ organic
  $r_{u,0} \leftarrow$ undef
  $c_{u,0} \leftarrow$ undef
  $\omega_{u,0} \sim \mathcal{N}(0_{K\times1}, I_K)$
  $v_{u,0} \sim$ Categorical(softmax($\Gamma\omega_{u,0}$))
  while $z_{u,t} \neq$ *stop* **do**
    $t \leftarrow t+1$
    $\omega_{u,t} \sim \mathcal{N}(\omega_{u,t-1}, \sigma_\omega^2 I_K)$
    if $c_{u,t-1} \neq 1$ **then**
      $z_{u,t} \sim$ Categorical($S_{z_{u,t-1},\text{organic}}, S_{z_{u,t-1},\text{bandit}}, S_{z_{u,t-1},\text{stop}}$)
    else
      $z_{u,t} =$ organic
    end
    if $z_{u,t} =$ *organic* **then**
      $v_{u,t} \sim$ Categorical(softmax($\Gamma\omega_{u,t} + \mu_\Gamma$))
      $r_{u,t} \leftarrow$ undef
      $c_{u,t} \leftarrow$ undef
    end
    if $z_{u,t} =$ *bandit* **then**
      $r_{u,t}$ is geneated from the policy
      $c_{u,t} \sim$ Bernoulli($[f(\boldsymbol{\beta}\omega_{u,t} + \mu_\beta)]_{r_{u,t}}$)
    end
  end
end

criteo

f is an increasing function

$$f : \mathbb{R} \to (0, 1)$$

$$v_0 \sim \text{categorical}\left(\text{softmax}\left(\begin{pmatrix}\Gamma_{1,1} & \cdot & \Gamma_{1,K}\\ \cdot & \cdot & \cdot\\ \cdot & \cdot & \cdot\\ \Gamma_{P,1} & \cdot & \Gamma_{P,K}\end{pmatrix}\begin{pmatrix}\omega_{0,1}\\ \cdot\\ \omega_{0,K}\end{pmatrix} + \begin{pmatrix}\mu_1\\ \cdot\\ \cdot\\ \mu_P\end{pmatrix}\right)\right)$$

$$v_1 \sim \text{categorical}\left(\text{softmax}\left(\begin{pmatrix}\Gamma_{1,1} & \cdot & \Gamma_{1,K}\\ \cdot & \cdot & \cdot\\ \cdot & \cdot & \cdot\\ \Gamma_{P,1} & \cdot & \Gamma_{P,K}\end{pmatrix}\begin{pmatrix}\omega_{1,1}\\ \cdot\\ \omega_{1,K}\end{pmatrix} + \begin{pmatrix}\mu_1\\ \cdot\\ \cdot\\ \mu_P\end{pmatrix}\right)\right)$$

$$v_2 \sim \text{categorical}\left(\text{softmax}\left(\begin{pmatrix}\Gamma_{1,1} & \cdot & \Gamma_{1,K}\\ \cdot & \cdot & \cdot\\ \cdot & \cdot & \cdot\\ \Gamma_{P,1} & \cdot & \Gamma_{P,K}\end{pmatrix}\begin{pmatrix}\omega_{2,1}\\ \cdot\\ \omega_{2,K}\end{pmatrix} + \begin{pmatrix}\mu_1\\ \cdot\\ \cdot\\ \mu_P\end{pmatrix}\right)\right)$$

$$c_3|r_3 \sim \text{Bernoulli}\left(\left[f\left(\begin{pmatrix}\beta_{1,1} & \cdot & \beta_{1,K}\\ \cdot & \cdot & \cdot\\ \cdot & \cdot & \cdot\\ \beta_{P,1} & \cdot & \beta_{P,K}\end{pmatrix}\begin{pmatrix}\omega_{3,1}\\ \cdot\\ \omega_{3,K}\end{pmatrix} + \begin{pmatrix}\mu_1^*\\ \cdot\\ \cdot\\ \mu_P^*\end{pmatrix}\right)\right]_{r_3}\right)$$

$$c_4|r_4 \sim \text{Bernoulli}\left(\left[f\left(\begin{pmatrix}\beta_{1,1} & \cdot & \beta_{1,K}\\ \cdot & \cdot & \cdot\\ \cdot & \cdot & \cdot\\ \beta_{P,1} & \cdot & \beta_{P,K}\end{pmatrix}\begin{pmatrix}\omega_{4,1}\\ \cdot\\ \omega_{4,K}\end{pmatrix} + \begin{pmatrix}\mu_1^*\\ \cdot\\ \cdot\\ \mu_P^*\end{pmatrix}\right)\right]_{r_4}\right)$$

$$v_5 \sim \text{categorical}\left(\text{softmax}\left(\begin{pmatrix}\Gamma_{1,1} & \cdot & \Gamma_{1,K}\\ \cdot & \cdot & \cdot\\ \cdot & \cdot & \cdot\\ \Gamma_{P,1} & \cdot & \Gamma_{P,K}\end{pmatrix}\begin{pmatrix}\omega_{5,1}\\ \cdot\\ \omega_{5,K}\end{pmatrix} + \begin{pmatrix}\mu_1\\ \cdot\\ \cdot\\ \mu_P\end{pmatrix}\right)\right)$$

time

criteo

The RecoGym Session

In [1]:
```
import gym, reco_gym

# env_0_args is a dictionary of default parameters (i.e. number of products)
from reco_gym import env_1_args, Configuration

# You can overwrite environment arguments here:
env_1_args['random_seed'] = 41

# Initialize the gym for the first time by calling .make() and .init_gym()
env = gym.make('reco-gym-v1')
env.init_gym(env_1_args)


env.reset() # we call request to move to the first user
```

In [2]:
```
observation, reward, done, info = env.step(None)
# We specify None because we have need to learn about the user before we can act
```

In [3]:
```
observation.current_sessions
```

Out[3]:
```
[{'t': 0, 'u': 0, 'z': 'pageview', 'v': 4},
 {'t': 1, 'u': 0, 'z': 'pageview', 'v': 4},
 {'t': 2, 'u': 0, 'z': 'pageview', 'v': 4},
 {'t': 3, 'u': 0, 'z': 'pageview', 'v': 0}]
```

In [4]:
```
# We see the user is interested in product id 4 (they viewed it 3 times) and product id 0 (they viewed it once)
```

criteo

The RecoGym Session

```
In [5]: observation, reward, done, info = env.step(0)

In [6]: reward # the user does not click on our recommendation of product 0
Out[6]: 0

In [7]: observation.current_sessions # the user does not view any more products, so we learn nothing more about them
Out[7]: []
```

criteo

The RecoGym Session

```
In [8]: observation, reward, done, info = env.step(0) # we recommend product 0 again

In [9]: reward # they do not click again, ctr are usually low - this is not surprising
Out[9]: 0

In [10]: observation.current_sessions # again no additional product views the bandit session continues
Out[10]: []
```

criteo

The RecoGym Session

```
In [11]: observation, reward, done, info = env.step(4) # we now recommend product 4

In [12]: reward # they clicked!  We must have done something right on that last recommendation
Out[12]: 1

In [13]: observation.current_sessions # the user moved to the retailer website and viewed the product
Out[13]: [{'t': 7, 'u': 0, 'z': 'pageview', 'v': 4}]
```

criteo

Unlike RL we continue to use logs

Let's start with a random logging policy (a crazy thing to do, but a useful theoretical concept).

```
In [17]: env.generate_logs(100)
```

Out[17]:

	a	c	ps	ps-a	t	u	v	z
0	NaN	NaN	NaN	None	0	0	2.0	organic
1	NaN	NaN	NaN	None	1	0	4.0	organic
2	NaN	NaN	NaN	None	2	0	4.0	organic
3	NaN	NaN	NaN	None	3	0	4.0	organic
4	NaN	NaN	NaN	None	4	0	2.0	organic
5	NaN	NaN	NaN	None	5	0	4.0	organic
6	NaN	NaN	NaN	None	6	0	5.0	organic
7	NaN	NaN	NaN	None	7	0	4.0	organic
8	NaN	NaN	NaN	None	8	0	4.0	organic
9	NaN	NaN	NaN	None	9	0	4.0	organic
10	3.0	0.0	0.1	[0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, ...	10	0	NaN	bandit
11	1.0	0.0	0.1	[0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, ...	11	0	NaN	bandit
12	2.0	0.0	0.1	[0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, ...	12	0	NaN	bandit
13	1.0	0.0	0.1	[0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, ...	13	0	NaN	bandit
14	8.0	0.0	0.1	[0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, ...	14	0	NaN	bandit
15	2.0	0.0	0.1	[0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, ...	15	0	NaN	bandit
16	5.0	0.0	0.1	[0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, ...	16	0	NaN	bandit
17	1.0	0.0	0.1	[0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, ...	17	0	NaN	bandit
18	7.0	0.0	0.1	[0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, ...	18	0	NaN	bandit
19	7.0	0.0	0.1	[0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, ...	19	0	NaN	bandit
20	5.0	0.0	0.1	[0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, ...	20	0	NaN	bandit

criteo

Let's get started

An introductory notebook to RecoGym can be found *here* We will be using Google Collab, so you have nothing to install !

criteo

5. Exercise no.1: Organic Best Of vs Bandit Best Of

Exercise - The best of the best

Go to the notebook *1. Organic vs Bandit Best Ofs*

- Examine the logs, to start with we will look at non-personalized behavior only.
- Plot a histogram of organic product popularity. What is the most popular product?
- Plot the (non-personalized) click through rate of each product (with an error analysis). What is the most clicked-on product?
- Plot popularity against click through rate. How good a proxy is popularity for click through rate?
- Simulate an AB test comparing a simple recommender system (agent) that always recommends the highest ctr product with a recommender system that always recommends the organically most popular product.

Quick Recap of diffs. between classic and modern reco

Classic vs. Modern Reco

	Classic Reco	**Last Click Sales**
Approach	Autocomplete	Intervention
Feedback	Organic	Bandit
Loss	Softmax	Sigmoid
Metrics	Offline ranking and regression	Online ABTest / Offline simulation of ABTest
Offline evaluation setup	Datasets	Propensity datasets / Simulators
Literature	RecSys	ComputationalAdvertising/ContextualBandits/RL
RealWorld deployment	Initial Solution	Advanced Solution

criteo

Section 2. Likelihood Based Agents

Likelihood Based Agent

How can we build an agent/recommender from historical recommendation data i.e. (X_n, a_n, c_n) ?

criteo

Likelihood Based Agent

How can we build an agent/recommender from historical recommendation data i.e. (X_n, a_n, c_n) ? The most straighforward way is to first frame the task as a reward/click prediction task.

Likelihood Based Agent

$$c_n \sim \text{Bernoulli}\left(\sigma\left(\Phi([\boldsymbol{X}_n\ \boldsymbol{a}_n])^T\beta\right)\right)$$

where

- $\boldsymbol{X}$ represent context or user features
- $\boldsymbol{a}$ represent action or product features
- β are the parameters;
- $\sigma(\cdot)$ is the logistic sigmoid;
- $\Phi(\cdot)$ is a function that maps $\boldsymbol{X}, \boldsymbol{a}$ to a higher dimensional space and includes some interaction terms between $\boldsymbol{X}_n$ and $\boldsymbol{a}_n$.

Likelihood Based Agent

$$c_n \sim \text{Bernoulli}\left(\sigma\left(\Phi([\boldsymbol{X}_n\ \boldsymbol{a}_n])^T\beta\right)\right)$$

where

- $\boldsymbol{X}$ represent context or user features
- $\boldsymbol{a}$ represent action or product features
- β are the parameters;
- $\sigma(\cdot)$ is the logistic sigmoid;
- $\Phi(\cdot)$ is a function that maps $\boldsymbol{X}, \boldsymbol{a}$ to a higher dimensional space and includes some interaction terms between $\boldsymbol{X}_n$ and $\boldsymbol{a}_n$. *Why?*

Modeling interaction between user/context and product/action

- $\Phi([\boldsymbol{X}_n\ \boldsymbol{a}_n]) = \boldsymbol{X}_n \otimes \boldsymbol{a}_n$ Kronecker product, this is somewhat what we will be doing in the example notebook, these will model pairwise interactions between user and product features.
- $\Phi([\boldsymbol{X}_n\ \boldsymbol{a}_n]) = \nu(\boldsymbol{X}_n) \cdot \mu(\boldsymbol{a}_n)$ Another more "modern" approach, is to build an embedded representation of the user and product features and consider the dot product between both (in this case the parameters of the model are carried in the embedding functions ν and μ and we have no parameter vector β)

criteo

Likelihood Based Agent

Then it's just a classification problem right ? We want to maximize the log likelihood :

$$\hat{\beta}_{\text{lh}} = \text{argmax}_{\beta} \sum_n c_n \log \sigma \left(\Phi \left([X_n \ a_n] \right) \right)^T \beta \Big) + (1 - c_n) \log \left(1 - \sigma \left(\Phi \left([X_n \ a_n] \right)^T \beta \right) \right)$$

criteo

Other possible tasks

Imagine we recommend a group of items (a banner with products for instance) and one of them gets clicked, an appropriate task would be to rank the clicked items higher than non clicked items.
Example: pairwise ranking loss

$$\sum_n \log \sigma \left(\Phi \left([\boldsymbol{X}_n \ \boldsymbol{a}_n] \right)^T \beta - \Phi \left([\boldsymbol{X}_n \ \boldsymbol{a'}_n] \right)^T \beta \right)$$

Where $\boldsymbol{a}_n$ represent a clicked item from group n and $\boldsymbol{a'}_n$ a non clicked item.
Multitude of possible other tasks (listwise ranking, top-1, ...)

criteo

2.1. Using maximum likelihood models as a policy

criteo

Context, Action, Policy

We first need to introduce the contextual bandit setting. It is a particularly useful version of the multi-armed bandit setting, where the agent observes before pulling an arm, a context vector on which to reward distribution relates.

Context, Action, Policy

Let us introduce the necessary vocabulary and notation:

- $x \in$ arbitrary *contexts* drawn from an unknown distribution ν
- $y \in$ denote the *actions* available to the decision maker
- A *policy* is a mapping $\pi :\rightarrow ()$ from the space of contexts to probabilities in the action space. For a given (context, action) pair (x, y), the quantity $\pi(y|x)$ denotes the probability of the policy π to take the action y when presented with the context x.

Policy Risk

- True action reward of y in the context x: $\delta(x, y)$ (This was
- Cost/loss: $c(x, y) \triangleq -\delta(x, y)$
- Policy Risk:

$$R(\theta) \triangleq_{x \sim \nu, y \sim \pi_\theta(\cdot|x)} [c(x, y)] \tag{1}$$

  If we replace by P the joint distribution over (states,actions) (x,y) pairs, we have that: $R(\theta) \triangleq_{(x,y) \sim P} [l(x, y)]$

Empirical Risk Minimization

- Empirical Risk Estimator:

$$\hat{R}_n(\theta) \triangleq \frac{1}{n} \sum_{i=1}^{n} c_i \tag{2}$$

- **Empirical Risk Minimization (ERM)** / Sample Average Aproximation (SAA):

$$\hat{\theta}_n^{ERM} \triangleq \underset{\theta \in \Theta}{} \hat{R}_n(\theta)$$

criteo

Value-based models

To choose which action to take, we need to evaluate all possible actions and take the one maximizing the predicted reward, this is a *Value based model*.

Likelihood model $\rightarrow$ Policy

Our max likelihood click predictor can be converted into a (deterministic) policy by just taking the action with the highest predicted click probability.

Sidepoint: Scaling argmax in the case of large action spaces

- Imagine we mapped the context/user features $\boldsymbol{X}$ into a dense representation (embedding) $\boldsymbol{u}$
- Same thing for the action/product features $\boldsymbol{a}$ into $\boldsymbol{v}$
- Such that $P(c = 1|\boldsymbol{X}, \boldsymbol{a}) = \sigma(\boldsymbol{u} \cdot \boldsymbol{v})$
- Choosing for a given context the reward maximizing action is a *Maximum Inner Product Search* problem, which is a very well studies problem for which approximate solutions have a high quality

Sidepoint: Scaling argmax in the case of large action spaces

- Several method families
  - Hashing based: LSH
  - Tree based: Random Projection Trees
  - Graph based: NSW, HSNW
- Bottom line: use and abuse them !

criteo

7. Exercise no.2: Pure bandit model

Pure Bandit Feedback

- Build a model using feature engineering using logistic regression (or a deep model)

criteo

Pure Bandit Feedback

- Build a model using feature engineering using logistic regression (or a deep model)
- Finally you evaluate your performance against production.

criteo

Pure Bandit Feedback

- Build a model using feature engineering using logistic regression (or a deep model)
- Finally you evaluate your performance against production.

Please look at notebook “2. Likelihood Agent.ipynb” Click here

criteo

Section 3. Shortcomings of Value-based Recommendations

criteo

3.1. The Covariate Shift Problem

criteo

Issue #1: The Covariate Shift Problem

The problem: We train our value model on one distribution of (user,action) / (x,y) pairs, but we test on another!

- Train on: $c(x, y) \times \pi_0(y|x)$
- Test on: $c(x, y) \times \pi_{new}(y|x)$

Since by definition we are learning a model of the rewards in order to do better recommendations, π_0 and π_{new} will always be different!

criteo

Covariate Shift. An Example

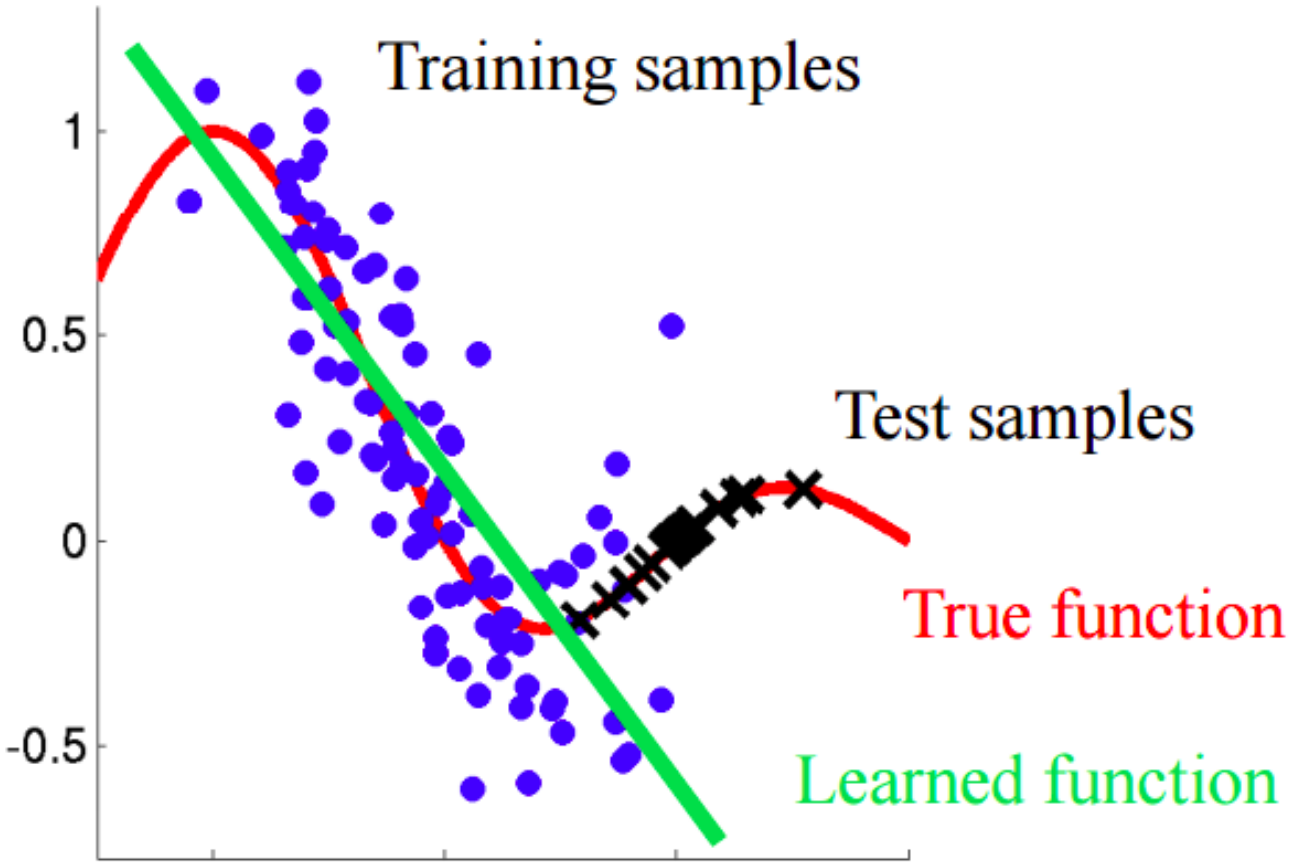


Figure 1: Train vs. Test: Blue vs. Black

criteo

Covariate Shift and Model Misspecification

Covariate Shift and Model Underfitting

- If we have a perfect predictor for $P(y|x)$, why should we care if $P(x)$ changes?
- Of course, $P(y|x)$ is not perfect and it is usually designed to underfit (to insure generalization through capacity constraints - limited degrees of freedom)
- In this case, we should allocate the model capacity on the portions of the $P(X)$ where the model will be tested

Covariate Shift and Recommendation

- **Note:** In the case of using value-based models for decision making (and in our case, Recommendation) we will evaluate all actions for all contexts so we will need to be good everywhere!

In Part 2, we will see how we are going to solve this, but first let's talk about the second issue!

3.2. The Optimizer's Curse

criteo

Issue #2: The Optimizer's Curse. A Story

Day1: Offline Evaluation Time (Predicted Reward)

- Imagine a decision problem, handed to us by our CEO, where we need to choose between **3 actions** and where, for each one of them, we have samples of historical rewards.
- In order to choose, we build empirical reward estimators for each of the 3 actions, we find the one that has the best historical performance and we go back to our CEO with the recommendation! After work, we congratulate ourselves and go to sleep happy!

criteo

Issue #2: The Optimizer's Curse. A Story

Day2: Online Evaluation Time (True Reward)

- However, the next day we meet with the CEO only to find out this was just a test of our decision-making capabilities, and that all of the sampled rewards for the 3 actions were **all drawn independently from the same distribution**.
- Due to the finite nature of our samples, each one of our 3 reward estimators made errors, some of them more negative (pessimistic), and some of them more positive (optimistic).
- Because when we decided, we selected the action with the highest estimate, we obviously favored the most optimistic of the 3, making us believe we optimized our decision, when we were just overfitting the historical returns. This effect is known at the **Optimizer's Curse**.

criteo

Issue #2: The Optimizer's Curse. A Story

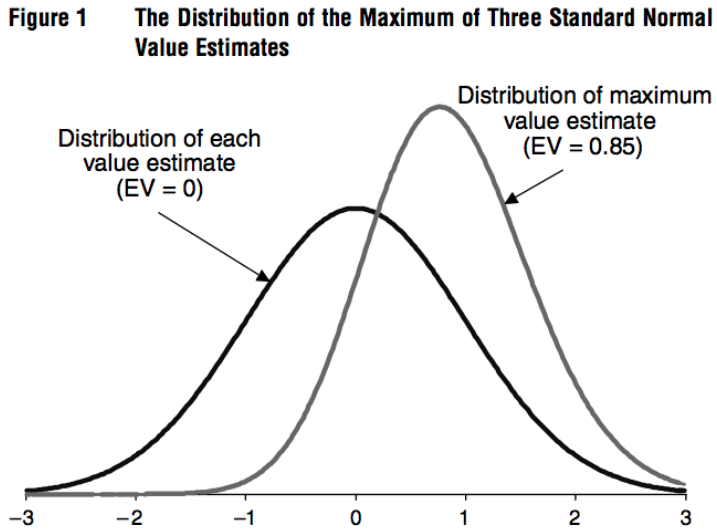


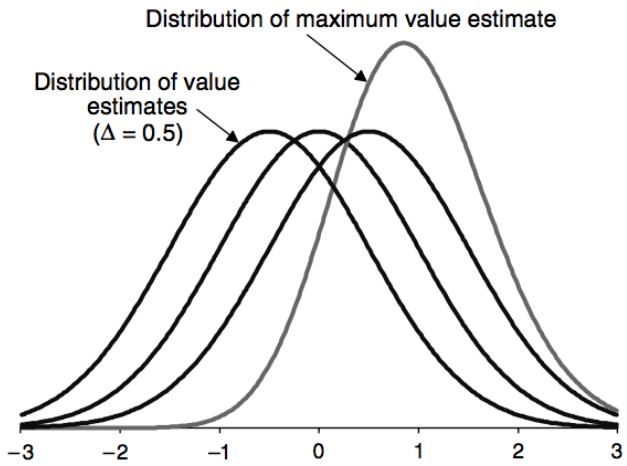
Figure 2: The distribution of rewards of argmax of 3 identical actions

Img Credits: The Optimizers Curse: Skepticism and Postdecision Surprise in Decision Analysis [?]

criteo

Issue #2: The Optimizer's Curse. A Story

The Distribution of Maximum Value Estimates with Separation Between Alternatives



Δ	Expected disappointment	Probability of correct choice
0.0	0.85	0.33
0.2	0.66	0.42
0.4	0.51	0.50
0.6	0.39	0.59
0.8	0.30	0.66
1.0	0.22	0.73
1.2	0.17	0.78
1.4	0.12	0.83
1.6	0.10	0.87
1.8	0.07	0.90
2.0	0.05	0.92
2.2	0.03	0.94
2.4	0.02	0.95
2.6	0.01	0.97
2.8	0.01	0.98
3.0	0.00	0.98

Figure 3: The distribution of rewards of argmax of 3 actions with delta separation

Img Credits: The Optimizers Curse: Skepticism and Postdecision Surprise in Decision Analysis [?]

criteo

The Optimizer's Curse in Recommendation

- The *Optimizer's Curse* mainly appears when the **reward estimates are based on small sample sizes** and **the deltas between the true mean rewards** of the different actions **are relatively small** such that the variance of the empirical estimators can lead to overoptimistic expectations or even erroneous ranking of actions.
- **This is not a corner case!** In real recommendation applications, the space of possible context-action pairs is extremely large, so there are always areas of the distribution where the current system is deciding based on very low sample sizes.

criteo

Fixing Covariate Shift and Optimizer's Curse with Policy Learning

We saw that value-based approaches for decision-making suffer from two issues **when using ERM-based models**.

Tomorrow we will show how to solve both issues !

criteo

Example 3. Showcasing the Optimizer's curse effect in our likelihood agent

criteo

Demo: Expected CTR vs. Observed CTR

We will be using our likelihood model to showcase optimizer's curse, where we overstimate the number of clicks we will be getting from showing particular products.
The notebook is "3 Optimizer's curse"

criteo

Beyond Optimizer's Curse: A Quick Overview of Decision-Making in the presence of Uncertainty

criteo

Recommender Systems as Decision-Making Systems

- With the *Optimizer's Curse*, we saw that Recommender Systems can be better described as policies and that a pure supervised ML formulation does not directly answer the true questions behind optimal RecSys...
- However, there is another body of literature that has been extremely active and that could answer some of our questions, which is the field of **Decision Making under Uncertainty**.

Decision Making under Uncertainty

- One of the key applications for the field was and remains **Finance** and within it, the area of **Portofolio Optimization**
- For this reason we will use it as our illustrative example in order to:
  - Cover the concepts of **Expected Value and Expected Utility**
  - Talk about the relationship between **Utility, Risk Aversiveness and Diminishing Returns**
  - Define the class of **Coherent Risk Measures** and transform the decision-making objective into a minimization problem
  - Finally, using the duality theorem, make the link between **Worst-case Risk Minimization and Distributional Robust Optimization**

criteo

Expected Value vs. Expected Utility

criteo

Motivating Example: Portofolio Optimization

Problem: Given a set of assets(stocks) I, what fraction of my current wealth W should I invest in each to maximize return?
Let R_i be the absolute return of each asset i. R_i is positive and stochastic and its associated rate of return r_i is in: $[-1, \infty)$. Let x_i be the fraction of the initial wealth W invested in i.
Constraint: Invest all money and no shorting: $\sum_{i \in I} x_i = 1, x \geq 0$.

Q: What is the optimal solution if we want to maximize expected value after one timestep?

A: Invest all money in the stock with the highest expected R_i!

criteo

The Expected Value Principle

- Optimizing the allocation to drive maximum expected value is known as the **Expected Value Principle.**
- But, is this really the best way to allocate our wealth?
- Let's look at another potential objective...

criteo

Motivating Example: Portofolio Optimization over multiple timesteps

Unlike in the previous example, we want to optimize our portofolio over *a series of timesteps* in the future.

What is different in this case is that, after each timestep we can reinvest our new wealth in the different stocks, leading to a compound growth of our wealth.

So let's define our new objective (also known as the **Kelly's Criterion**): We want to **find the allocation that maximizes the growth of our expected wealth at timestep n (denoted as W_n) versus our start wealth** W_0:

$$x^* = argmax_x \frac{E[W_n]}{W_0} \tag{3}$$

criteo

Portofolio Growth Optimization

Example: For simplicity, we assume a toy example situation, where we need to allocate between risk-free cash and a single stock that returns the time series of stochastic returns r_i coming from a known distribution with mean return μ.

Then at timestep 1, for a fractional allocation x, we have that our wealth W_1 is equal to:

$$W_1 = (1 - x)W_0 + xW_0(1 + r_1) = W_0(1 + xr_1) \tag{4}$$

By induction we have that W_n is equal to the product of the multiple timestep returns:

$$W_n = W_0 \prod_{i=1}^{n} (1 + xr_i) \tag{5}$$

Portofolio Growth Optimization

Plugging it back in our growth optimization objective, we have that:

$$x^* = argmax_x \frac{W_n}{W_0} = argmax_x \frac{W_0 \prod_{i=1}^{n}(1 + xr_i)}{W_0} = argmax_x \prod_{i=1}^{n}(1 + xr_i) \tag{6}$$

Now, let's assume that the return rate r_i can take only two values, namely a positive rate of return r^+ with probability P, and a negative rate of return r^- with probability (1-P). Then by CLT, as n goes to ∞, we have that:

$$x^* = argmax_x \prod_{i=1}^{n}(1 + xr_i) \sim argmax_x (1 + xr^+)^{nP}(1 + xr^-)^{n(1-P)} \tag{7}$$

criteo

Portofolio Growth Optimization

Furthermore, since taking the log maintains the argmax solution (since its monotonically increasing) we have that:

$$\begin{aligned} x^* &= argmax_x \log((1 + xr^+)^{nP}(1 + xr^-)^{n(1-P)}) = \\ &= argmax_x P \log(1 + xr^+) + (1 - P) \log(1 + xr^-) \end{aligned} \tag{8}$$

This means, that instead of maximizing expected return, we now want to **maximize expected log return** $g = log(1 + r)$**!**

criteo

Portofolio Growth Optimization

Differentiating the logarithmic expression and solving for x gives us:

$$x^* = -\frac{P}{r^-} - \frac{1-P}{r^+} \tag{9}$$

We observe that *once we take into account the longer horizon impact of our local decisions*, **the optimal allocation changes from putting all mass on the stock with the best return to a mix between risk-free and risky assets**.

Portofolio Growth Optimization

However, this is not very clear why this is the case.
For this to become painfully obvious we need to undergo a transformation of our objective...

criteo

Volatility drag: Expressing the impact of stock volatility over compounded wealth growth

We can use the **Taylor expansion** to approximate the log returns around their mean:

$$ln(1+r) \approx ln(1+\mu) + \frac{r-\mu}{1+\mu} + \frac{(r-\mu)^2}{2(1+\mu)^2} + \frac{(r-\mu)^3}{3(1+\mu)^3} - \frac{(r-\mu)^4}{4(1+\mu)^4} + ... \tag{10}$$

Taking the expectation and dropping higher order terms we have that the log of the *Expected Geometric Return* of the portofolio is:

$$E[ln(1+g)] \approx ln(1+\mu) - \frac{E[(r-\mu)^2]}{2(1+\mu)^2} \tag{11}$$

Therefore the *Expected Geometric Return* depends on the log arithmetic mean of the risky asset μ and its volatility σ:

$$g \approx \exp\left\{ln(1+\mu) - \frac{\sigma^2}{2(1+\mu)^2}\right\} - 1 \tag{12}$$

criteo

Volatility drag

We found that the geometric mean return is a function of the arithmetic mean return and variance, with variance reducing the growth rate, which is known as the *volatility drag on growth*.

Volatility drag

Example: On the first day, the portofolio value improves by 100%; on the second day it loses 50%. The arithmetic mean of the two returns is 25%, yet after both periods, the true compound return is 0%, which mean no portfolio growth: $100 \times (1 + 100\%) \times (1 - 50\%) = 100$

- The arithmetic mean of the return rate: $\mu = (1 - 0.5)/2 = 0.25\%$
- The geometric mean of the return rate:
  $g = \sqrt{(1+1) \times (1-0.5)} - 1 = \sqrt{1} - 1 = 0\%$
- The arithmetic mean of the log return rate:
  $g' = (\log 2 + \log 0.5)/2 = 0\%$

The effect is clear: **volatility causes a divergence between the arithmetic and geometric mean.**

Optimizer's Curse, Kelly's Criterion...

We see now two examples where maximizing expected reward is not the right answer.

- It looks like (short-term/myopic) Expected Value is only a *means to an end*.
- But if maximizing expected value is not the best objective, then what?

What Expected Value is not measuring

- At larger scale, the agent taking the actions wants first to self-preserve and secondly to grow.
- This means that there is intrinsic value in guaranteeing that the agent will stay in the game (e.g. live to die another day).
- If we assume that getting to a state of zero reward (aka wealth, energy, health) means the *death* of the agent, there is value in bounding the risk of going to zero.
- This is exactly what we see in the growth portofolio strategy, **if current wealth is zero there is no way to grow in the future!**

criteo

What Expected Value is not measuring

- **There is intrinsic value in being risk adverse!**
- Mean does not differentiate between distributions where the outcomes frequently go to zero or become negative and that are compensated by a few huge positive outliers.
- **Note: Median however does!**

criteo

Utility, Risk-Aversiveness and Diminishing Returns

criteo

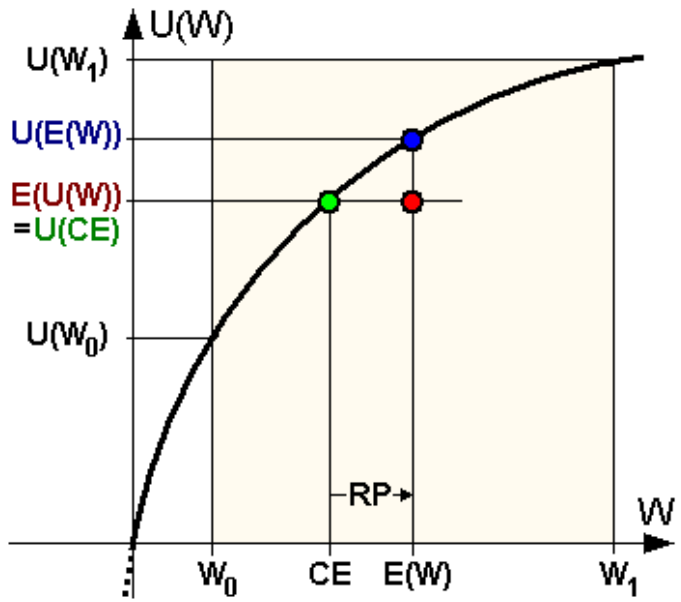
Maximizing Expected Utility of Reward

- We are moving from maximizing wealth/rewards directly to **maximizing a utility function that depends on them.**
- We want to maximize a function that grows with the increases of wealth, not with their absolute value, but with their relative value to the existing wealth.
- **For example**, starting from \$0 wealth, the first earned \$1000 are arguably more important (allowing one to buy food, shelter) than an additional \$1000 after already earning \$1M.

criteo

Diminishing Returns = Concave Utility Function

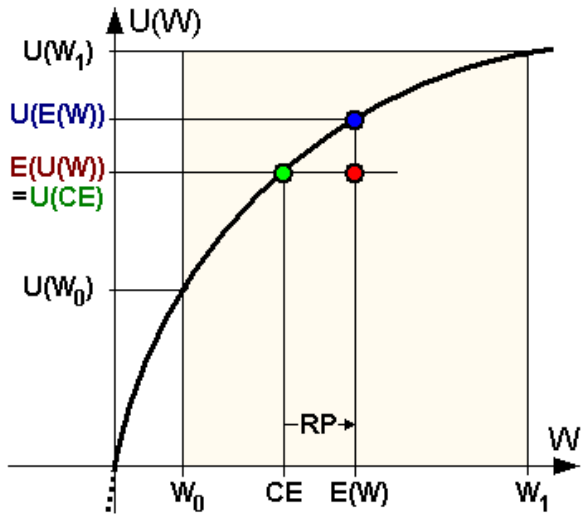
- If the utility function values more the first units of wealth than the later ones, it is exhibiting a *diminishing returns property*
- **Diminishing returns = Concave utility function** = Decreasing first moment of the utility function



criteo

Risk Premimum

- CE - Certainty equivalent
- E(U(W)) - Expected utility of the uncertain reward W
- E(W) - Expected value of the uncertain reward W
- U(CE) - Utility of the certainty equivalent
- U(E(W)) - Utility of the expected value of the uncertain reward
- **RP - Risk premium = E(W) - CE is the difference between expected reward and the certain reward that have the same utility according to the concave utility curve U.**



criteo

Connection with Risk Aversiveness

- If the utility function leads to a positive Risk Premium, $RP > 0$, this means that the agent is *Risk Averse*.
- This means that:
  Risk Aversiveness = Concave Utility = Diminishing Returns

criteo

From Maximizing Utility to Minimizing Risk

criteo

Traditional Portfolio Risk Measures

- **Mean-Variance:** Minimize the portofolio's variance/standard deviation (with a constrain on minimum acceptable expected returns). It was introduced by Harry Markowitz in [Markowitz, 1952]. The main idea is to diversify the stocks in the portofolio since the sum of independent stock variances is smaller than the variance of a big position in a single stock.
- **Value at Risk (VaR):** For a given portfolio, time horizon, and probability p, the VaR_p can be defined informally as the maximum possible loss during that time after we exclude all worse outcomes whose combined probability is at most p.

criteo

Mean-Variance. Diversification and Risk

In finance, **diversification is the process of allocating capital in a way that reduces the exposure to any one particular asset or risk.** A common path towards diversification is to reduce risk or volatility by investing in a variety of assets.

Mean-Variance. Diversification and Risk

For example, let asset X have stochastic return x and asset Y have stochastic return y, with respective return variances σ_x^2, σ_y^2.

If the fraction q of a one-unit (e.g. one-million-dollar) portfolio is placed in asset X and the fraction 1-q is placed in Y, the stochastic portfolio return is $qx + (1-q)y$.

If x and y are uncorrelated, the variance of portfolio return is $\text{var}(qx + (1-q)y) = q^2\sigma_x^2 + (1-q)^2\sigma_y^2$

The variance-minimizing value of q is $q = \sigma_y^2/[\sigma_x^2 + \sigma_y^2]$, which is strictly between 0 and 1.

criteo

Mean-Variance. Diversification and Risk

Using this value of q in the expression for the variance of portfolio return gives the latter as $\sigma_x^2\sigma_y^2/[\sigma_x^2+\sigma_y^2]$, **which is less than what it would be at either of the un-diversified values q=1 and q=0** (which respectively give portfolio return variance of σ_x^2 and σ_y^2.

Trading-off between mean return and risk variance.

An efficient portfolio:

- the portfolio that **maximizes the expected return** for a given amount of risk (standard deviation)
- or in the *dual representation*, the portfolio that **minimizes the risk** subject to a given expected return.

Mean-Variance. Minimizing risk given expected return

So, to calculate the efficient frontier of mean-variance trade-offs, we have to **minimize the risk (standard deviation) given some expected return.**

This is a *Quadratic Programming* problem, e.g. a problem of optimizing a quadratic function of several variables subject to linear constraints on these variables:

$$
\begin{aligned}
\text{Min} \quad & \sum_{i=1}^{n} \sum_{j=1}^{n} \sigma_{ij} x_i x_j \\
st \quad & \\
& \sum_{i=1}^{n} \mu_i x_i = \rho \\
& \sum_{i=1}^{n} x_i = 1 \\
& x_i \geq 0 \qquad\qquad i = 1, \ldots, n
\end{aligned}
$$

criteo

Value-at-Risk Objective

Let X be a profit and loss distribution (loss negative and profit positive).

p-VaR is defined such that **the probability of a loss greater than VaR is (at most) p** while the probability of a loss less than VaR is (at least) $1 - p$. A loss which exceeds the VaR threshold is termed a "VaR breach

Mathematically, $VaR_\alpha(X)$ is the $(1 - \alpha)$-quantile of Y, i.e.,

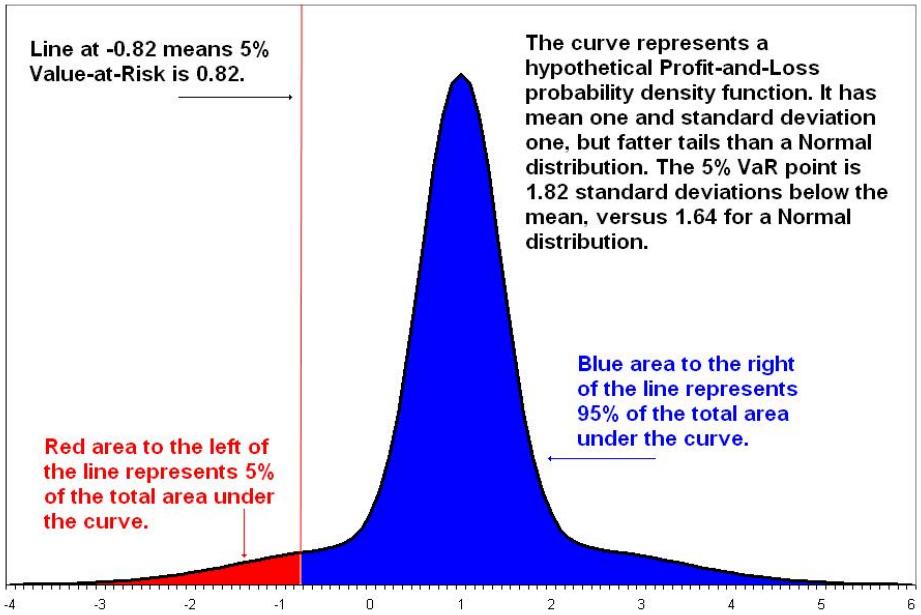
$$VaR_\alpha(X) = -\inf\left\{x \in \mathbb{R} : F_X(x) > \alpha\right\} = F_Y^{-1}(1 - \alpha).$$

criteo

Value-at-Risk Objective

Example: The $5\% - VaR$ is 0.82, meaning that:

- the probability of the future return being less than -0.82 is at most 5%
- the probability of the future return being more than -0.82 is at least 95%



criteo

From Traditional to Modern Risk Measures

Shortcomings of Traditional Risk Measures:

- **Mean-Variance:** The resulting portfolios are too sensitive to small changes in the model inputs. Also, stock returns are empirically asymmetric, so variance is not a good measure of risk (semivariance and VaR fix that).
- **VaR:** VaR is not subadditive, meaning VaR of a combined portfolio can be larger than the sum of the VaRs of its components.

From Traditional to Modern Risk Measures

With a lot of options for risk measures, is there a way to decide which one to optimize for?

In their seminal work [Artzner et al., 1999], the authors established an axiomatic framework that risk measures should satisfy in order to exhibit *coherent behaviour*.

criteo

Coherent Risk Measures. Axiomatic Properties

- (P1) *Sub-additivity:* $R(Z + Z') \leq R(Z) + R(Z')$. It means that the risk of the union of two portofolios should be lower than the sum of the separate risks. This is inline with the idea of diversification to lower overall return variance.
- (P2) *Positive homogeneity:* $R(\lambda Z) = \lambda R(Z)$. It means that doubling the allocation on a stock, translates in doubling its risk.
- (P3) *Monotonicity:* $R(Z) \leq R(Z')$ *when* $Z \leq Z'$, *meaning* $prob[Z > Z'] = 0$. If losses in portfolio Z' are larger than losses in portfolio Z for all possible scenarios, then the risk of portfolio Z' is higher than the risk of portfolio Z.
- (P4) *Translation invariance.* $R(Z + C) = R(Z) - c$ for constants C. This means that adding risk-free assets C to a portofolio reduces its risk.

criteo

Coherent Risk Measures. Revised Axiomatic Properties

In [Rockafellar and Uryasev, 2013], the authors revisit the axioms and propose a revised set of properties that works better with optimization theory.

- (R1) R is lower semicontinuous, possibly going to ∞
- (R2) R is convex
- (R3) R is monotonic
- (R4) R has risk-neutral elements $R(C) = C$
- (R4') Additionally, R is risk adverse: $R(X) > E(X)$ for stochastic X

Connection with the concavity of utility functions: The negative counterpart of a concave function is convex, so the axiomatic approach matches our previous observation on the concavity of the utility function!

Are traditional risk measures coherent?

Looking at the two we covered:

- Mean-Variance = Yes.
- VaR = No, since it is not-subadditive.

criteo

Coherent risk measures

Modern(coherent) risk measures:

- **Conditional Value at Risk (CVaR)** (also known as Expected Shortfall, Average Value at Risk, Expected Tail Loss , Tail Value at Risk)
- **Conditional Entropic Value at Risk (EVaR)**

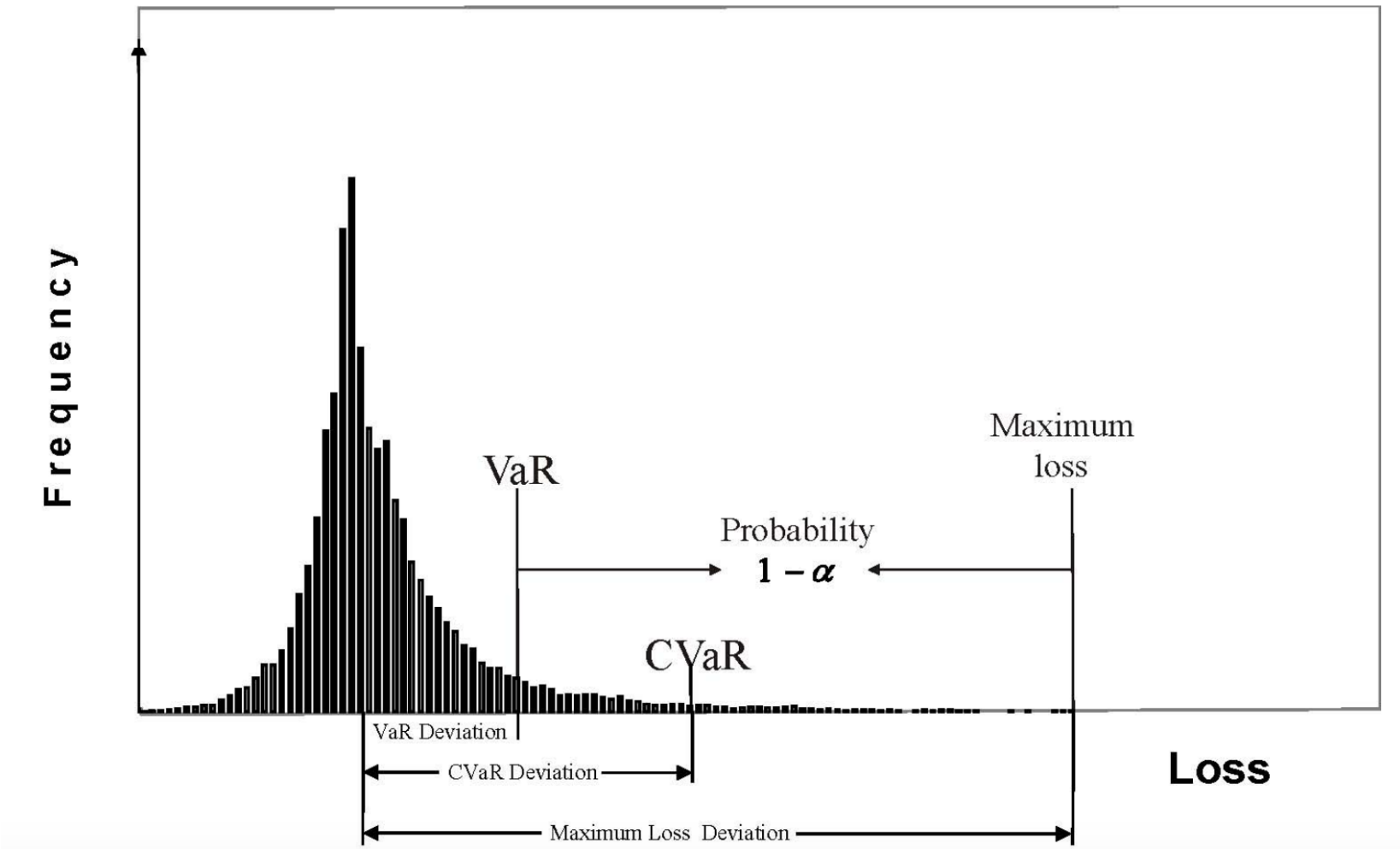
CVaR: Conditional Value at Risk

If $X \in L^p(\mathcal{F})$ (an Lp space) is the payoff of a portfolio at some future time and $0 < \alpha < 1$ then we define the expected shortfall as:

$$CVaR_\alpha = -\frac{1}{\alpha} \int_0^\alpha VaR_\gamma(X)\, d\gamma \tag{13}$$

where VaR_γ is the value at risk.

CVaR: Relationship with VaR



CVaR: Relationship with VaR

- CVaR has superior mathematical properties versus VaR
- Risk management with CVaR functions can be done very efficiently
- VaR does not control scenarios exceeding VaR
- CVaR accounts for losses exceeding VaR

criteo

The dual representation of coherent risk measures and Distributional Robust Optimization

criteo

Dual representation of coherent risk measures

A convex risk measure can be written in its *dual form*:

$$R(X) = sup_{Q \in M}(E_Q[-X] - \alpha(Q)) \tag{14}$$

where:

- M is a set of probability measures such that $E_Q[X]$ is well-defined for all $Q \in M$
- α is called the penalty function

Distributionally Robust Risk

The elements of the set M can be interpreted as possible probabilistic models, which are taken more or less seriously according to the size of the penalty $\alpha(Q)$.

The risk $R(X)$ is computed as the worst-case expectation taken over all models $Q \in M$ and penalized by $\alpha(Q)$.

Under this dual formulation, the risk $R(X)$ can be seen as **distributionally robust risk** estimator version of the normal risk estimator which is computed over the finite sample available at training time.

Distributionally Robust Risk

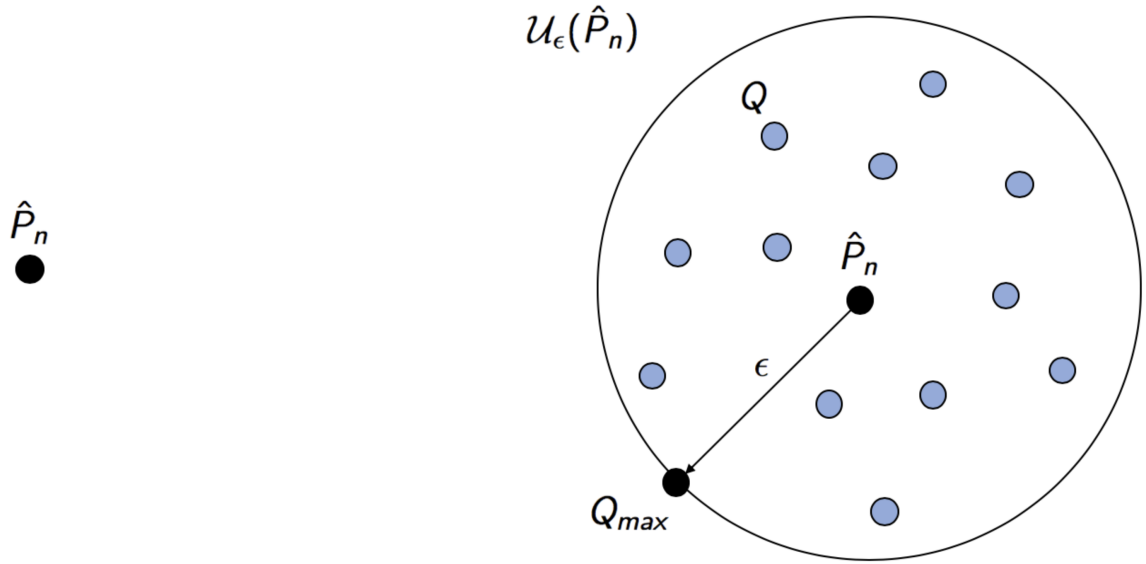


Figure 4: Empirical Risk vs. Empirical Robust Risk

criteo

Distributionally Robust Optimization (DRO)

By switching the definition of risk to its dual/robust representation, the original risk minimization problem becomes a minimax problem, e.g. **to minimize the (max)worst-case of risk under all of the Q probability distributions in M**:

$$X^* = argmin_X(sup_{Q \in M}(E_Q[-X] - \alpha(Q))) \tag{15}$$

The choice of the penalty function $\alpha(Q)$

- Since every coherent risk measure can be represented as a robust risk with an associated coherent penalty/deviation) function, the choice of penalty function becomes crucial
- In the context of probabilistic Qs, the penalty function can be interpreted as a divergence measure between distributions

F-Divergences

In probability theory, an **f-divergence** is a function $D_f(P||Q)$ that measures the difference between two probability distributions P and Q.

Let P and Q be two probability distributions over a space Ω such that P is absolutely continuous with respect to Q. Then, for a convex function f such that f(1) = 0, the f-divergence of P from Q is defined as:

$$D_f(P \parallel Q) \equiv \int_{\Omega} f\left(\frac{dP}{dQ}\right) dQ. \tag{16}$$

Intuition: Think of the F-divergence as an average, weighted by the function f, of the odds ratio given by P and Q. For the ones familiar with KL-divergence, this is a generalization of the concept.

F-Divergences

Example of F-divergences and their associated f(t) functions:

Divergence	f(t)
KL-divergence	$t \log t$
Reverse KL-divergence	$-\log t$
Hellinger distance	$(\sqrt{t}-1)^2, 2(1-t)$
Total variation distance	$\frac{1}{2}\lvert t-1\rvert$
Chi-squared distance	$(t-1)^2, t^2-1, t^2-t$

criteo

DRO with Coherent F-divergences

- The minimization of the robust risk based on **any coherent F-divergences** is *asymptotically equivalent* with the robust version of the **Mean-Variance** objective - see [Duchi et al., 2016]
- The minimization of the robust risk based on **Chi-squared distance** is *exactly equivalent* with the robust version of the **Mean-Variance** objective - see [Duchi et al., 2016]
- The minimization of the robust risk based on **KL distance** is *exactly equivalent* with the **EVaR (i.e. Entropic Value at Risk)** objective, which is closely related but a more computational tractable/easier to estimate alternative to CVaR - see [Ahmadi-Javid and Fallah-Tafti, 2019]

DRO with Coherent F-divergences

- Using the DRO framework we can recover as special cases some of the most important risk measures
- DRO provides a unifying framework for a lot of the previous work on Decision-Making under Uncertainty
- In Part 2, we will see how we can frame the Recommendation problem in the DRO framework and introduce computationally tractable optimization algorithms.

Wrapping-up Part 1 of our course

criteo

What have we learnt today?

- How the classical recommendation setting differs from real world recommendation
- Framing the problem in the contextual bandit vocabulary
- Using a simulator aka RecoGym to build and evaluate recommender agents
- Shortcomings of ERM based policies (covariate shift and optimizer's curse)
- The deeper connections between reward and uncertainty in decision making

criteo

About Part II happening tomorrow

- We will introduce an alternative to ERM to solve the covariate shift issue
- A framework to reason about optimizer's curse and solve the issue

criteo

Thank You!

criteo