RecSys: test & deploy

План:

1. A/B/.../N tests
2. Многорукие бандиты
3. Обзорные статьи по RL в RecSys
4. Общая архитектура для production

1. A/B тесты

Оценка качества рекомендательных алгоритмов

1. Offline тестирование модели на исторических данных с помощью ретро-тестов;

2. Тестирование готовых моделей с помощью A/B тестов;

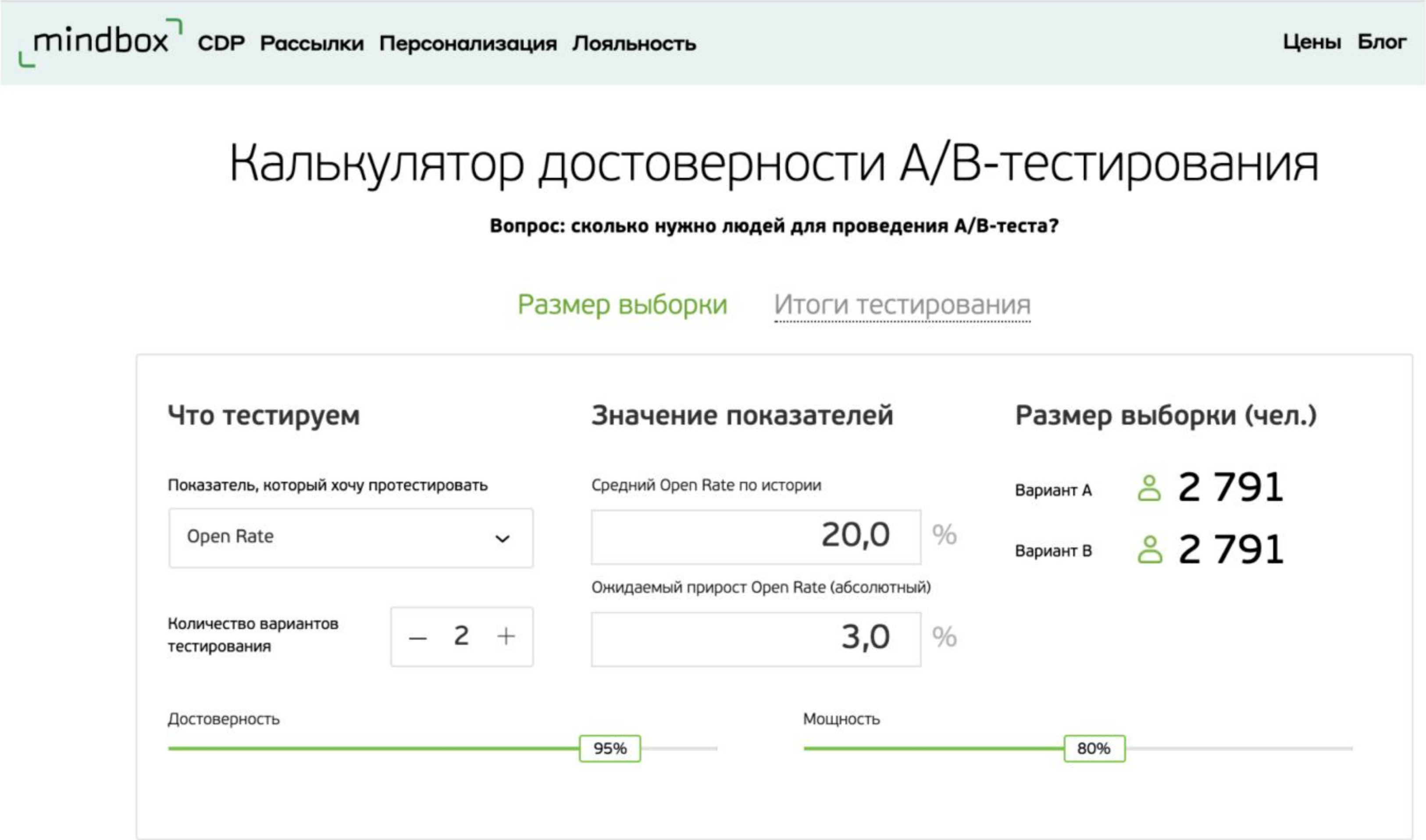
3. Online оценка изменения целевых метрик (отложенный reward)

A/B

1. Хотим убедиться, что наши рекомендации получились хорошими.
   - Есть оффлайн метрики (по оценке качества алгоритма на исторических данных)
   - Есть онлайн метрики:
     - бизнесовые таргетные метрики (например, CTR)
     - можно считать метрики для оценки качества на новых данных (NDCG@k, MNAP@k, etc.)

Этапы

1. Определить точку роста и метрику
2. Составить гипотезу (в идеале - с ожидаемым приростом нужной метрики)
3. Определить размер выборки для тестирования
4. Логировать нужные данные и метрики
5. Через N дней подсчитать результаты по метрикам и статистическую значимость



Калькулятор достоверности A/B-тестирования

Вопрос: сколько нужно людей для проведения A/B-теста?

Что тестируем

Показатель, который хочу протестировать

Open Rate

Количество вариантов тестирования: – 2 +

Значение показателей

Средний Open Rate по истории

20,0 %

Ожидаемый прирост Open Rate (абсолютный)

3,0 %

Размер выборки (чел.)

Вариант А 2 791

Вариант В 2 791

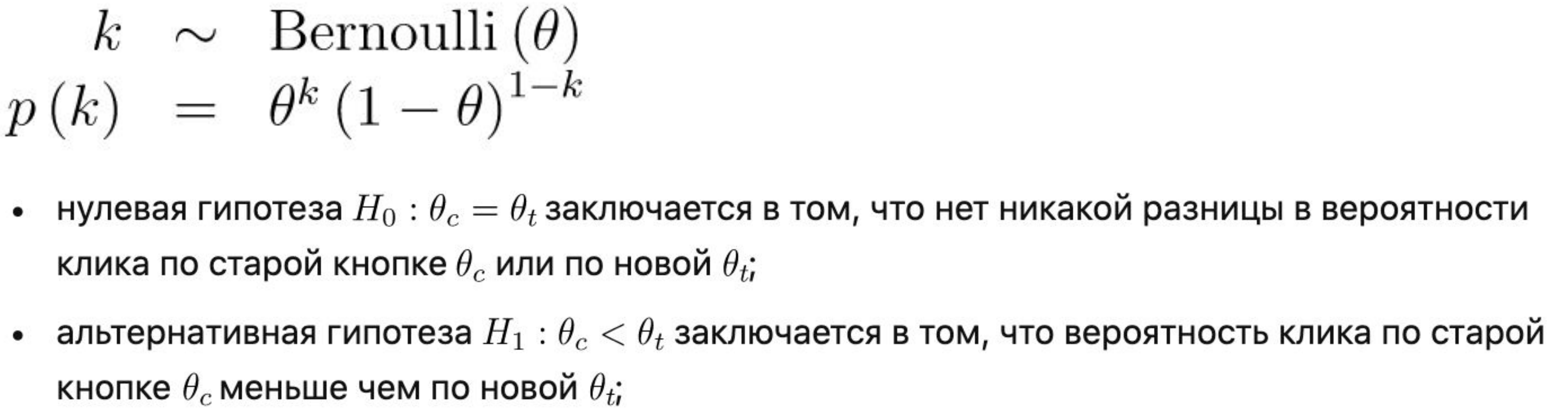
Достоверность 95%

Мощность 80%

Недостатки A/B тестов

- Долго
- Часть пользователей по дефолту увидит плохие рекомендации
- Дорого проводить
- Высокая цена ошибки при постановке теста
- Негибкость (нельзя менять состав групп/рекомендаций на ходу)

Пример



$$\begin{aligned} k &\sim \mathrm{Bernoulli}\left(\theta\right) \\ p\left(k\right) &= \theta^{k}\left(1-\theta\right)^{1-k} \end{aligned}$$

- нулевая гипотеза $H_0 : \theta_c = \theta_t$ заключается в том, что нет никакой разницы в вероятности клика по старой кнопке θ_c или по новой θ_t;
- альтернативная гипотеза $H_1 : \theta_c < \theta_t$ заключается в том, что вероятность клика по старой кнопке θ_c меньше чем по новой θ_t;

Пример

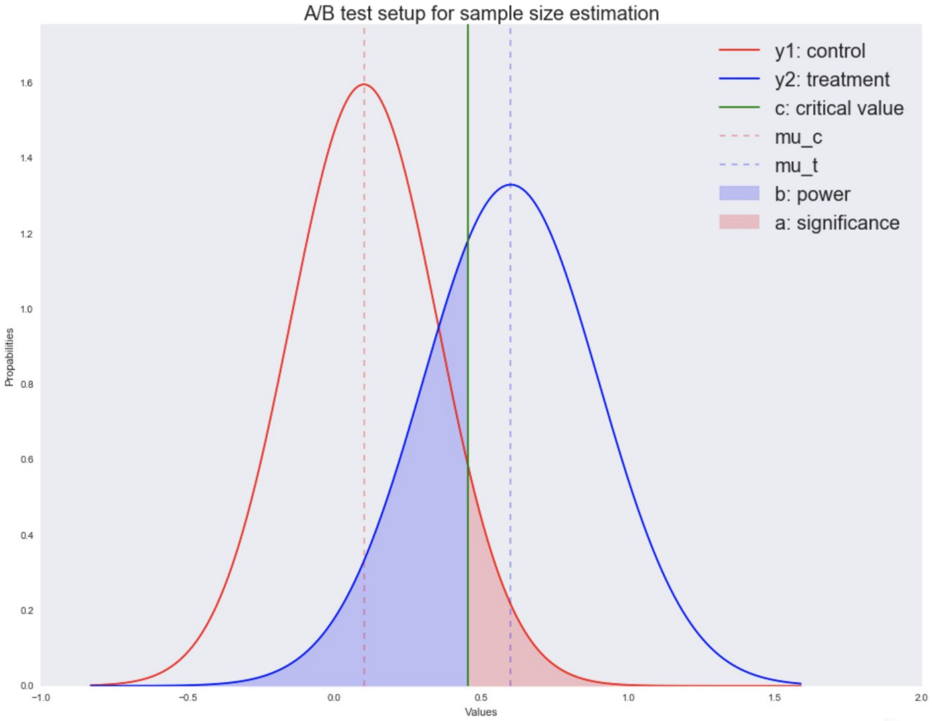
$$\alpha = P\left(H_1 \mid H_0\right),$$

$$\beta = P\left(H_0 \mid H_1\right).$$

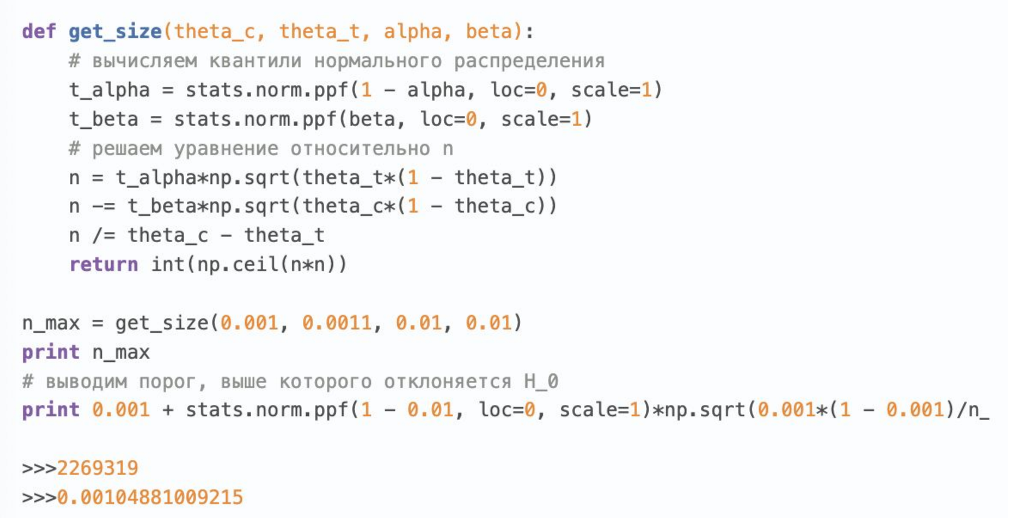
		Верная гипотеза	
		H_0	H_1
Ответ теста	H_0	H_0 принята	H_0 неверно принята (ошибка II рода)
	H_1	H_0 неверно отвергнута (ошибка I рода)	H_0 отвергнута

$$c = \mu + t\frac{\sigma}{\sqrt{n}}$$

$$t = P\left(X \le t\right), X \sim \mathcal{N}\left(0,1\right)$$



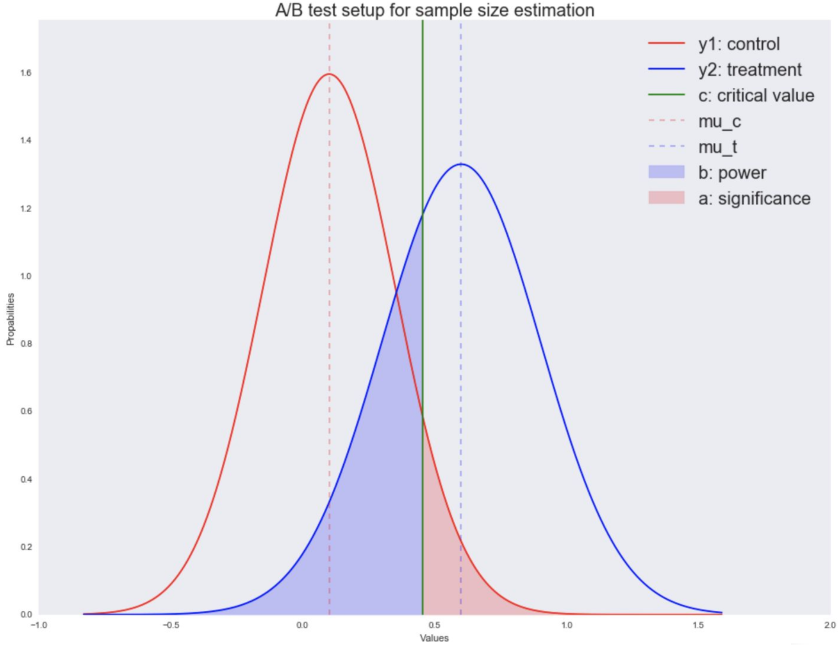
Пример



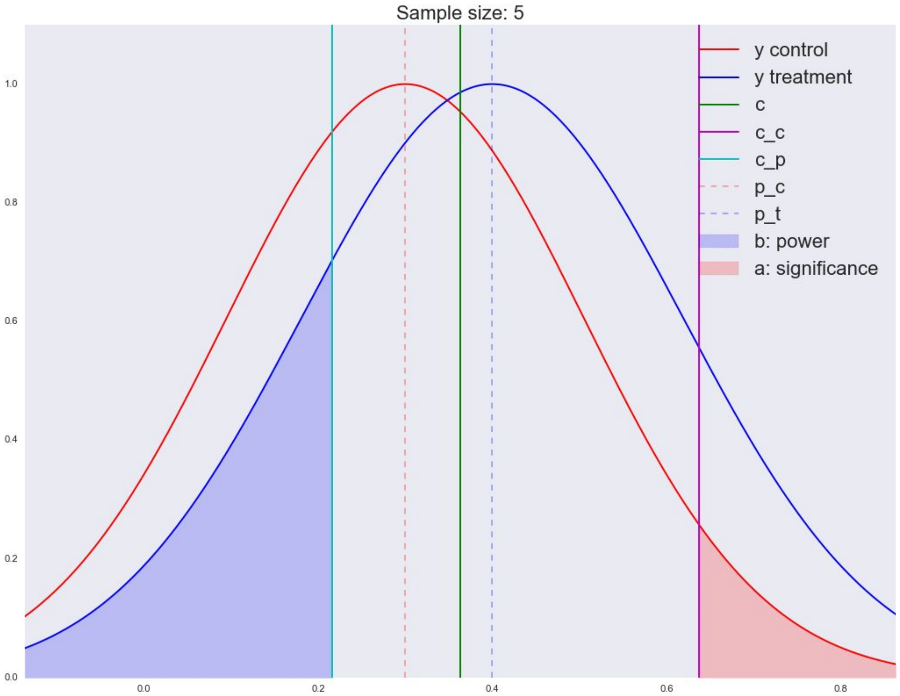
```
def get_size(theta_c, theta_t, alpha, beta):
    # вычисляем квантили нормального распределения
    t_alpha = stats.norm.ppf(1 - alpha, loc=0, scale=1)
    t_beta = stats.norm.ppf(beta, loc=0, scale=1)
    # решаем уравнение относительно n
    n = t_alpha*np.sqrt(theta_t*(1 - theta_t))
    n -= t_beta*np.sqrt(theta_c*(1 - theta_c))
    n /= theta_c - theta_t
    return int(np.ceil(n*n))

n_max = get_size(0.001, 0.0011, 0.01, 0.01)
print n_max
# выводим порог, выше которого отклоняется H_0
print 0.001 + stats.norm.ppf(1 - 0.01, loc=0, scale=1)*np.sqrt(0.001*(1 - 0.001)/n_

>>>2269319
>>>0.00104881009215
```



Пример


Sample size: 5
y control
y treatment
c
c_c
c_p
p_c
p_t
b: power
a: significance
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8

Поправка Холма-Бонферонни

$$\begin{aligned}
P\left(\text{хотя бы один результат значимый}\right) &= 1 - P\left(\text{все результаты незначимы}\right) \\
&= 1 - (1 - 0.05)^5 \\
&= 1 - 0.95^5 \\
&\approx 0.2262
\end{aligned}$$

Поправка Холма-Бонферонни

$$\begin{aligned}
P\,(\text{хотя бы один результат значимый}) &= 1 - P\,(\text{все результаты незначимы}) \\
&= 1 - (1 - 0.05)^5 \\
&= 1 - 0.95^5 \\
&\approx 0.2262
\end{aligned}$$

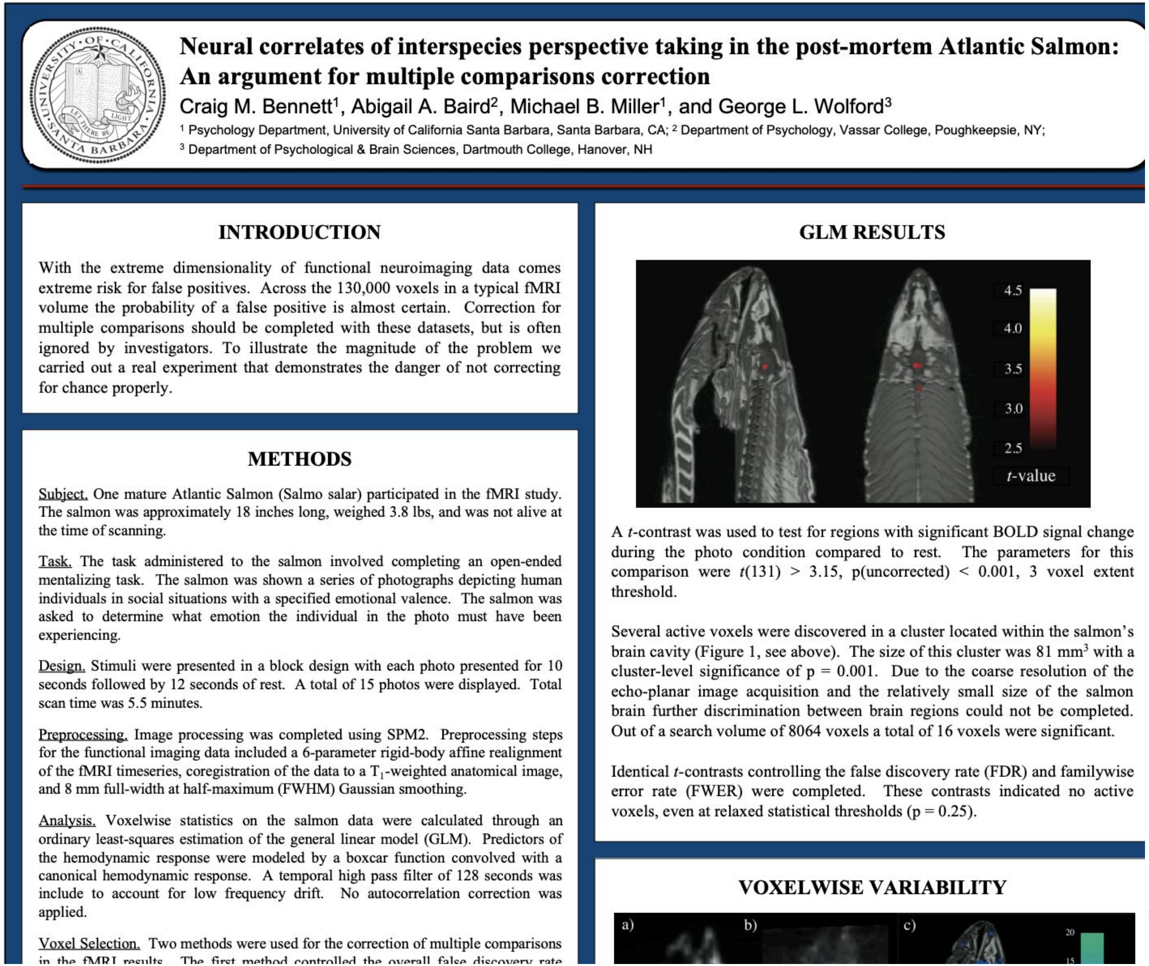
$$\begin{align}
P\left(\bigcup_{i=1}^{n}\left(p_i < \frac{\alpha}{n}\right)\right) &\leq \sum_{i=1}^{n} P\left(p_i < \frac{\alpha}{n}\right) \tag{1} \\
&\leq \sum_{i=1}^{n} \frac{\alpha}{n} \tag{2} \\
&= n\frac{\alpha}{n} \tag{3} \\
&= \alpha \tag{4}
\end{align}$$

Поправка Холма-Бонферонни

$$\begin{aligned}P\left(\text{хотя бы один результат значимый}\right) &= 1 - P\left(\text{все результаты незначимы}\right)\\ &= 1 - \left(1 - 0.05\right)^5\\ &= 1 - 0.95^5\\ &\approx 0.2262\end{aligned}$$

С поправкой:

$$\begin{aligned}P\left(\text{хотя бы один результат значимый}\right) &= 1 - P\left(\text{все результаты незначимы}\right)\\ &= 1 - \left(1 - 0.01\right)^5\\ &= 1 - 0.99^5\\ &\approx 0.0491\end{aligned}$$



Neural correlates of interspecies perspective taking in the post-mortem Atlantic Salmon: An argument for multiple comparisons correction

Craig M. Bennett[1], Abigail A. Baird[2], Michael B. Miller[1], and George L. Wolford[3]

[1] Psychology Department, University of California Santa Barbara, Santa Barbara, CA; [2] Department of Psychology, Vassar College, Poughkeepsie, NY;
[3] Department of Psychological & Brain Sciences, Dartmouth College, Hanover, NH

INTRODUCTION

With the extreme dimensionality of functional neuroimaging data comes extreme risk for false positives. Across the 130,000 voxels in a typical fMRI volume the probability of a false positive is almost certain. Correction for multiple comparisons should be completed with these datasets, but is often ignored by investigators. To illustrate the magnitude of the problem we carried out a real experiment that demonstrates the danger of not correcting for chance properly.

METHODS

Subject. One mature Atlantic Salmon (Salmo salar) participated in the fMRI study. The salmon was approximately 18 inches long, weighed 3.8 lbs, and was not alive at the time of scanning.

Task. The task administered to the salmon involved completing an open-ended mentalizing task. The salmon was shown a series of photographs depicting human individuals in social situations with a specified emotional valence. The salmon was asked to determine what emotion the individual in the photo must have been experiencing.

Design. Stimuli were presented in a block design with each photo presented for 10 seconds followed by 12 seconds of rest. A total of 15 photos were displayed. Total scan time was 5.5 minutes.

Preprocessing. Image processing was completed using SPM2. Preprocessing steps for the functional imaging data included a 6-parameter rigid-body affine realignment of the fMRI timeseries, coregistration of the data to a T_1-weighted anatomical image, and 8 mm full-width at half-maximum (FWHM) Gaussian smoothing.

Analysis. Voxelwise statistics on the salmon data were calculated through an ordinary least-squares estimation of the general linear model (GLM). Predictors of the hemodynamic response were modeled by a boxcar function convolved with a canonical hemodynamic response. A temporal high pass filter of 128 seconds was include to account for low frequency drift. No autocorrelation correction was applied.

Voxel Selection. Two methods were used for the correction of multiple comparisons in the fMRI results. The first method controlled the overall false discovery rate

GLM RESULTS

A *t*-contrast was used to test for regions with significant BOLD signal change during the photo condition compared to rest. The parameters for this comparison were $t(131) > 3.15$, p(uncorrected) < 0.001, 3 voxel extent threshold.

Several active voxels were discovered in a cluster located within the salmon's brain cavity (Figure 1, see above). The size of this cluster was 81 mm^3 with a cluster-level significance of p = 0.001. Due to the coarse resolution of the echo-planar image acquisition and the relatively small size of the salmon brain further discrimination between brain regions could not be completed. Out of a search volume of 8064 voxels a total of 16 voxels were significant.

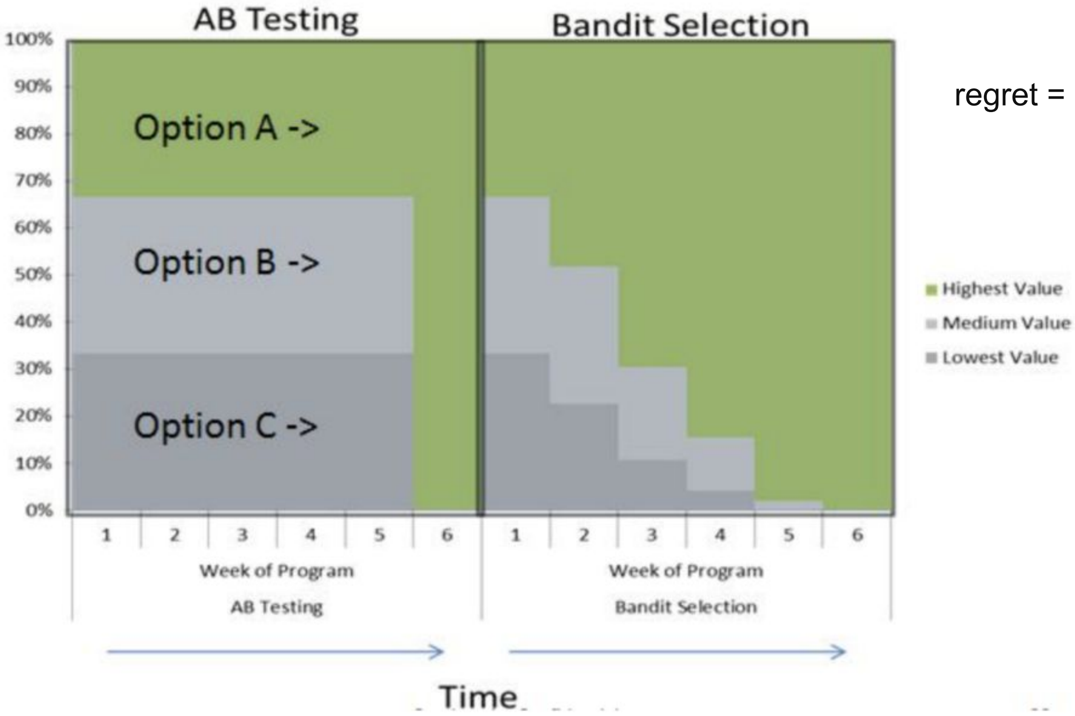
Identical *t*-contrasts controlling the false discovery rate (FDR) and familywise error rate (FWER) were completed. These contrasts indicated no active voxels, even at relaxed statistical thresholds (p = 0.25).

VOXELWISE VARIABILITY

http://prefrontal.org/files/posters/Bennett-Salmon-2009.pdf

2. Многорукие бандиты

A/B vs. многорукие бандиты

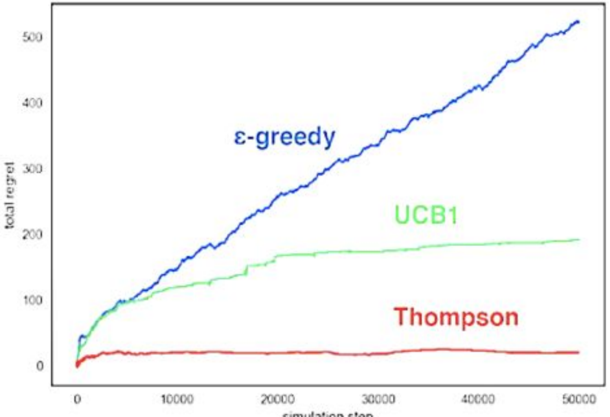


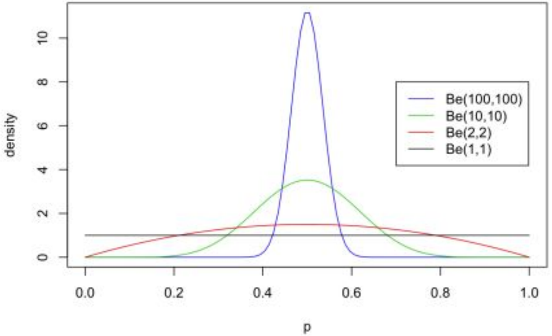
regret = R_opt - R_strategy

Стратегии

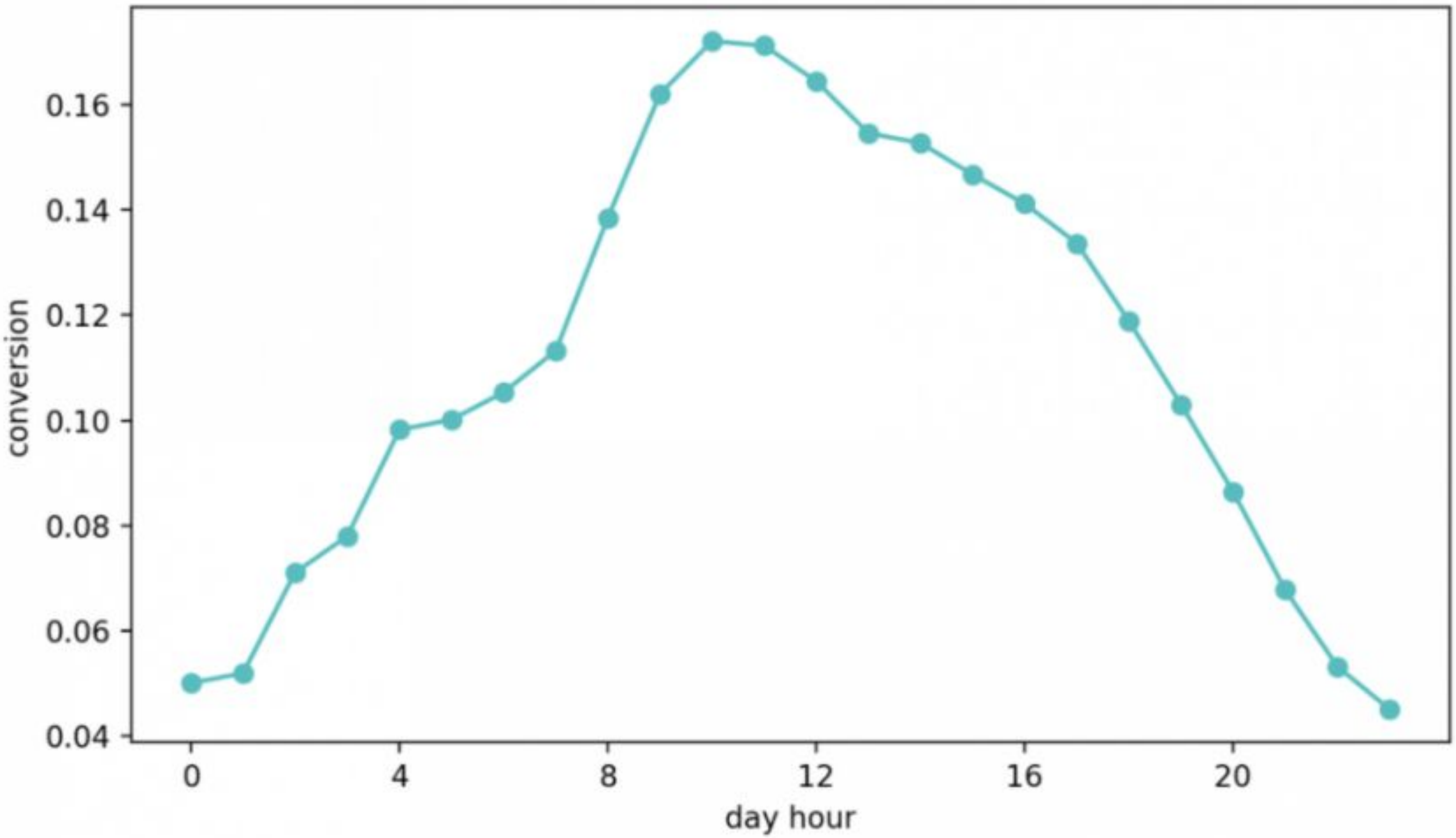
ε - greedy	Upper Confidence Bound	Thompson Sampling
Выбираем arm по наибольшему $\bar{x}$	$\overline{x}_j + \sqrt{\frac{2\ln n}{n_j}}$	$arm = \arg\max_{\text{j}}\ reward_j \mid params$

Total Strategy Regret





https://web.stanford.edu/~bvr/pubs/TS_Tutorial.pdf


0.16
0.14
0.12
0.10
0.08
0.06
0.04
conversion
0
4
8
12
16
20
day hour

Thompson Sampling & Bayes

$$\vec{y}_t = (y_1, y_2, \ldots, y_t).$$

$$f_{a_t}\left(y \mid \vec{\theta}\right)$$

$$\mu_a\left(\vec{\theta}\right) = \mathbb{E}\left[y_t \mid \vec{\theta}, a_t = a\right]$$

$$f_a\left(y \mid \theta_a\right) = \theta_a^y \left(1 - \theta_a\right)^{1-y}$$

$$\mu_a = \theta_a$$

Thompson Sampling & Bayes

$$\vec{y}_t = \left(y_1, y_2, \ldots, y_t\right).$$

$$f_{a_t}\left(y \mid \vec{\theta}\right)$$

$$\mu_a\left(\vec{\theta}\right) = \mathbb{E}\left[y_t \mid \vec{\theta}, a_t = a\right]$$

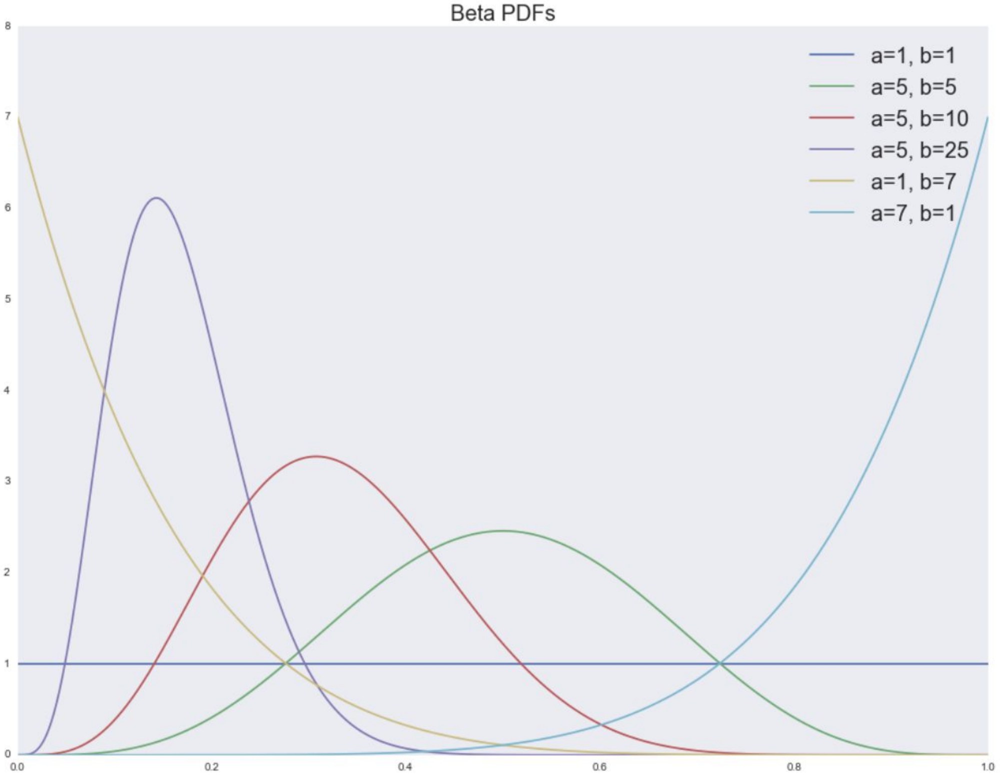
$$\begin{aligned}
p\left(\theta \mid y\right) &\propto p\left(\theta\right) \cdot p\left(y \mid \theta\right) \\
&\propto \tfrac{1}{\mathrm{B}(\alpha,\beta)}\theta^{\alpha-1}\left(1-\theta\right)^{\beta-1} \cdot \theta^{y}\left(1-\theta\right)^{1-y} \\
&\propto \theta^{\alpha-1+y}\left(1-\theta\right)^{\beta-1+1-y}
\end{aligned}$$

$$\mathrm{Beta}\left(\alpha+y, \beta+1-y\right) = \mathrm{Beta}\left(\theta \mid \alpha, \beta\right) \cdot \mathrm{Bernoulli}\left(y \mid \theta\right)$$

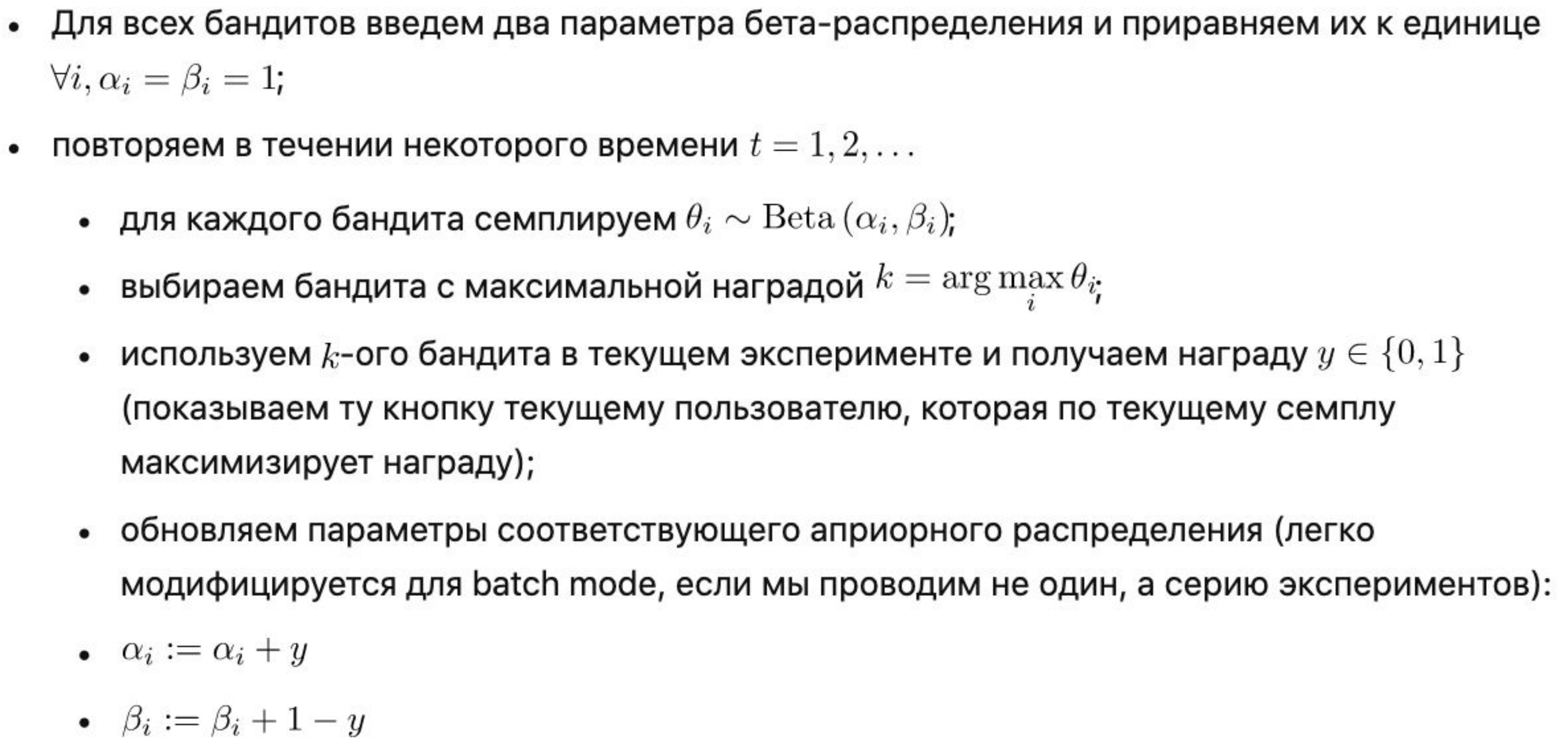
$$f\left(\theta, \alpha, \beta\right) = \tfrac{1}{\mathrm{B}(\alpha,\beta)}\theta^{\alpha-1} \cdot \left(1-\theta\right)^{\beta-1}$$

Thompson Sampling

$$\begin{aligned} \theta_i &\sim \mathrm{Beta}\left(\alpha_i, \beta_i\right) \\ y_i &\sim \mathrm{Bernoulli}\left(\theta_i\right) \end{aligned}$$



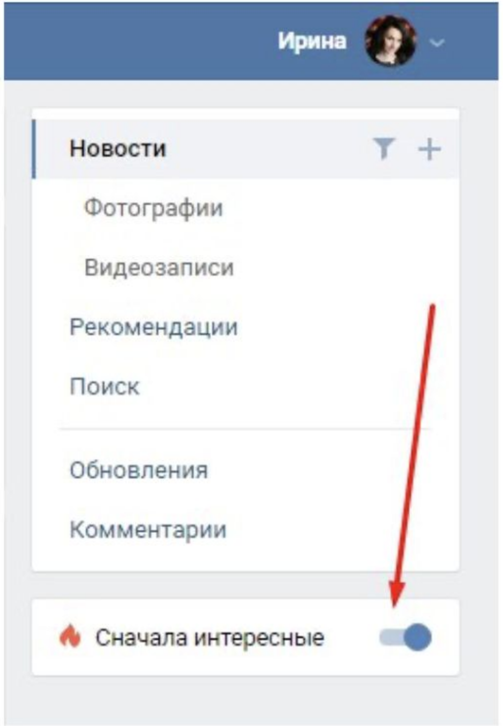
Thompson Sampling & Bayes



- Для всех бандитов введем два параметра бета-распределения и приравняем их к единице $\forall i, \alpha_i = \beta_i = 1$;
- повторяем в течении некоторого времени $t = 1, 2, \ldots$
  - для каждого бандита семплируем $\theta_i \sim \mathrm{Beta}\left(\alpha_i, \beta_i\right)$;
  - выбираем бандита с максимальной наградой $k = \arg\max\limits_i \theta_i$;
  - используем k-ого бандита в текущем эксперименте и получаем награду $y \in \{0, 1\}$ (показываем ту кнопку текущему пользователю, которая по текущему семплу максимизирует награду);
  - обновляем параметры соответствующего априорного распределения (легко модифицируется для batch mode, если мы проводим не один, а серию экспериментов):
  - $\alpha_i := \alpha_i + y$
  - $\beta_i := \beta_i + 1 - y$

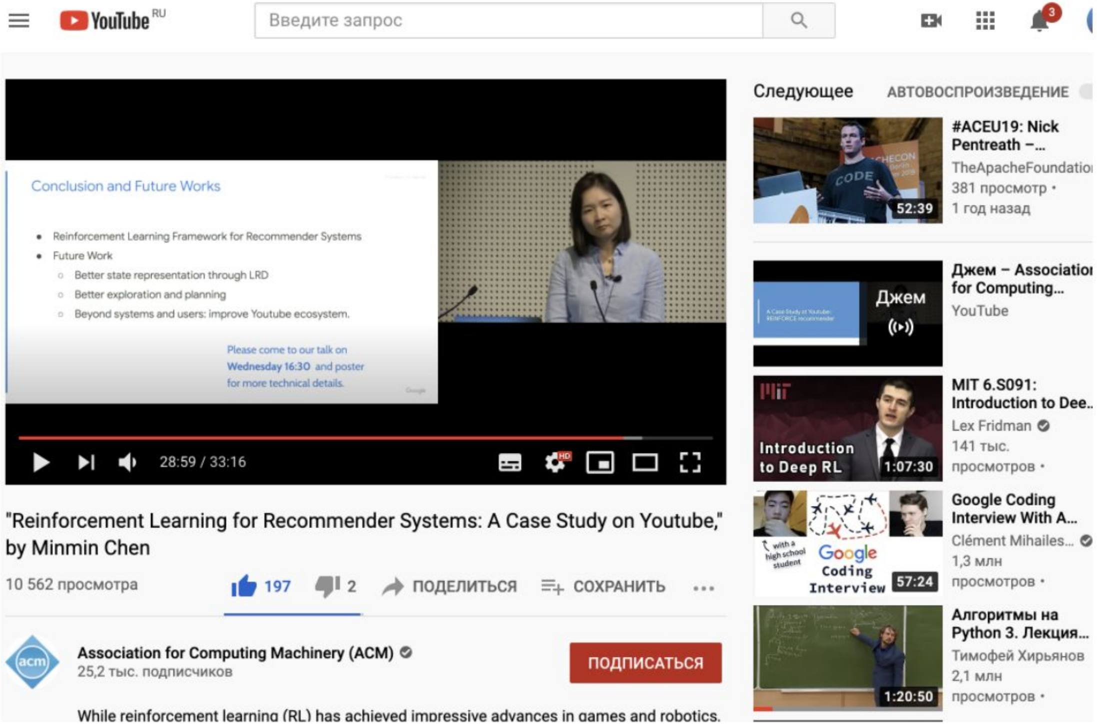
3. RL для RecSys

Интерактивные рекомендации за сессию



Сначала интересные ●

Персональная лента VK



Conclusion and Future Works

- Reinforcement Learning Framework for Recommender Systems
- Future Work
  - Better state representation through LRD
  - Better exploration and planning
  - Beyond systems and users: improve Youtube ecosystem.

Please come to our talk on **Wednesday 16:30** and poster for more technical details.

"Reinforcement Learning for Recommender Systems: A Case Study on Youtube," by Minmin Chen

10 562 просмотра 197 2

Association for Computing Machinery (ACM)
25,2 тыс. подписчиков

While reinforcement learning (RL) has achieved impressive advances in games and robotics.

52:39	#ACEU19: Nick Pentreath –...	TheApacheFoundatio 381 просмотр • 1 год назад
	Джем – Associatio for Computing...	YouTube
1:07:30	MIT 6.S091: Introduction to Dee...	Lex Fridman 141 тыс. просмотров •
57:24	Google Coding Interview With A...	Clément Mihailes... 1,3 млн просмотров •
1:20:50	Алгоритмы на Python 3. Лекция...	Тимофей Хирьянов 2,1 млн просмотров •

Лента с рекомендациями следующих видео YouTube

Новые потребности user-centric рекомендаций

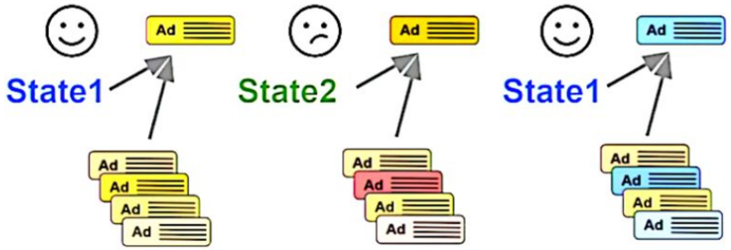
1) Персонализация текущей сессии в моменте, а не по заранее рассчитанным предсказаниям

2) Оценка взаимодействия с последовательность рекомендаций на сервисе за текущую сессию

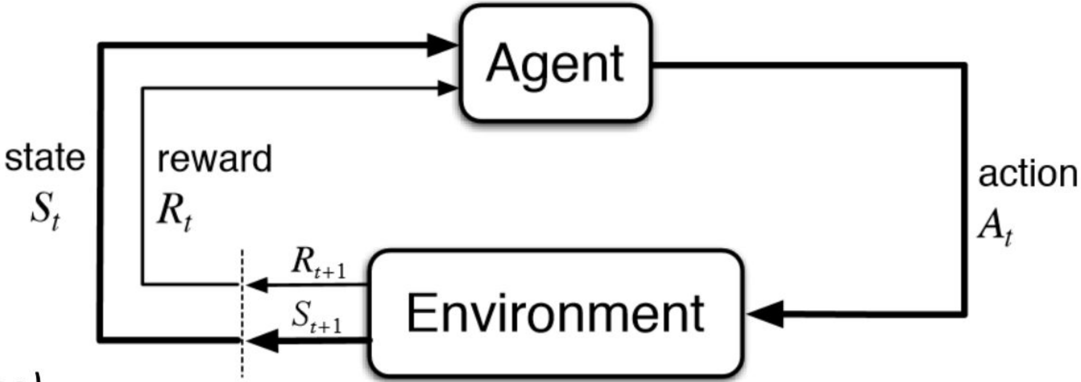
3) Оценка влияния текущего опыта от рекомендаций на дальнейшее использование сервиса

Вызовы user-centric рекомендаций

- Масштаб (пространство пользователей (= кол-во MDPs) и действий, сочетаний объектов)
- Стохастическая природа действий
- Латентное состояние пользователей
- Динамичность эко-системы (и ее тип, например, частично наблюдаемая)
- Дорогой exploration
- Шумный reward



Обозначения



S – множество состояний *(state space)*

A – множество действий *(action space)*

R$_a$ – текущая награда (reward) в результате действия ***a.***

Q(s, a) – функция суммы будущих (дисконтированных) наград.

Вероятность, что действие ***a*** в состоянии ***s*** в момент времени ***t*** приведет к ***s'***:

$$P_a(s,s') = \Pr(s_{t+1} = s' \mid s_t = s, a_t = a)$$

π(a|s) **-** это ***policy*** (функция из состояния в действие)

Одна из постановок задачи

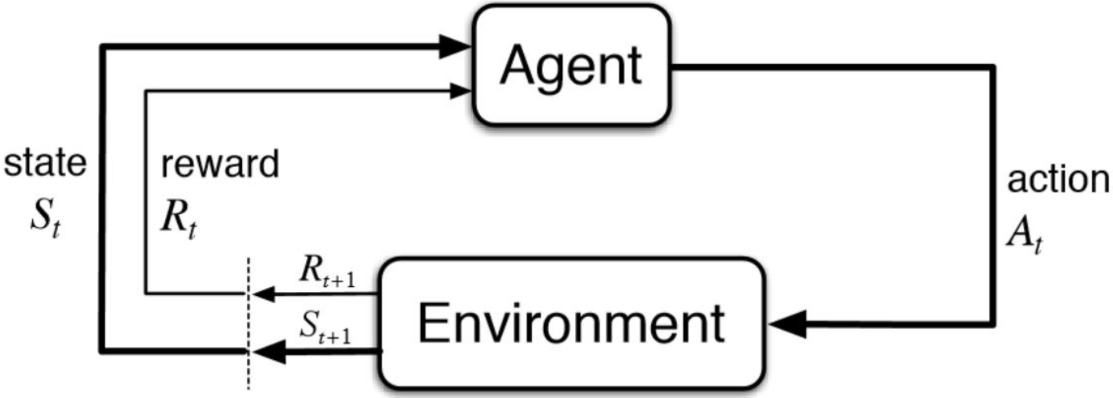
Задача:

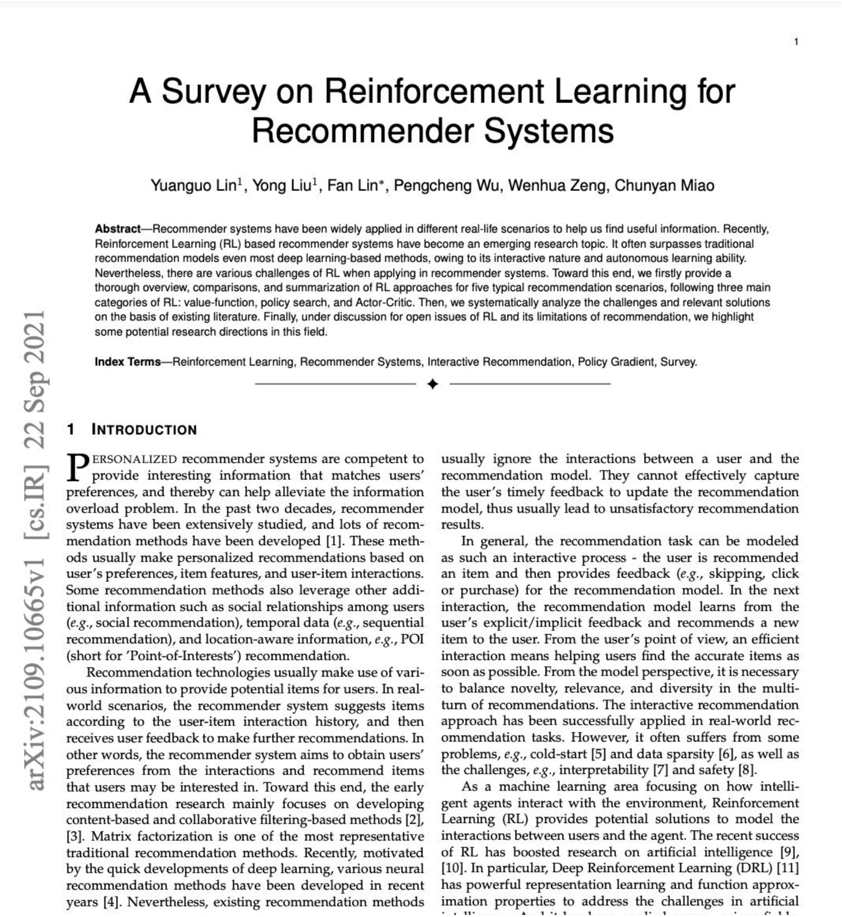
$$\pi = P(a|s) : E[R] \rightarrow max$$

Целевая функция:

$$R_t = r_t + \gamma \cdot r_{t+1} + \gamma^2 \cdot r_{t+2} + \ldots + \gamma^n \cdot r_{t+n}$$

$$R_t = \sum_i \gamma^i \cdot r_{t+i} \qquad \gamma \in (0,1) const$$





A Survey on Reinforcement Learning for Recommender Systems

Yuanguo Lin[1], Yong Liu[1], Fan Lin*, Pengcheng Wu, Wenhua Zeng, Chunyan Miao

Abstract—Recommender systems have been widely applied in different real-life scenarios to help us find useful information. Recently, Reinforcement Learning (RL) based recommender systems have become an emerging research topic. It often surpasses traditional recommendation models even most deep learning-based methods, owing to its interactive nature and autonomous learning ability. Nevertheless, there are various challenges of RL when applying in recommender systems. Toward this end, we firstly provide a thorough overview, comparisons, and summarization of RL approaches for five typical recommendation scenarios, following three main categories of RL: value-function, policy search, and Actor-Critic. Then, we systematically analyze the challenges and relevant solutions on the basis of existing literature. Finally, under discussion for open issues of RL and its limitations of recommendation, we highlight some potential research directions in this field.

Index Terms—Reinforcement Learning, Recommender Systems, Interactive Recommendation, Policy Gradient, Survey.

1 Introduction

Personalized recommender systems are competent to provide interesting information that matches users' preferences, and thereby can help alleviate the information overload problem. In the past two decades, recommender systems have been extensively studied, and lots of recommendation methods have been developed [1]. These methods usually make personalized recommendations based on user's preferences, item features, and user-item interactions. Some recommendation methods also leverage other additional information such as social relationships among users (*e.g.*, social recommendation), temporal data (*e.g.*, sequential recommendation), and location-aware information, *e.g.*, POI (short for 'Point-of-Interests') recommendation.

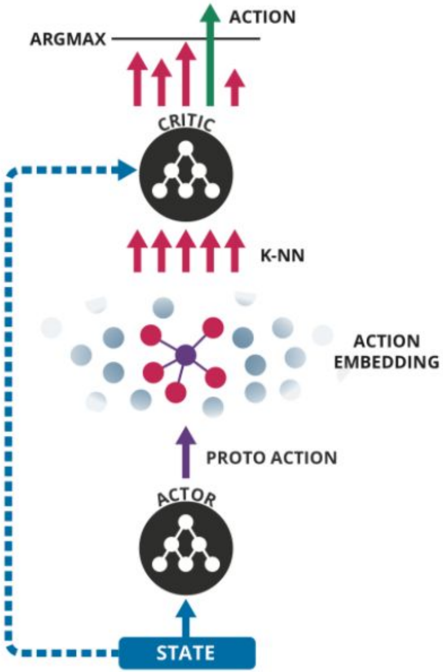
Recommendation technologies usually make use of various information to provide potential items for users. In real-world scenarios, the recommender system suggests items according to the user-item interaction history, and then receives user feedback to make further recommendations. In other words, the recommender system aims to obtain users' preferences from the interactions and recommend items that users may be interested in. Toward this end, the early recommendation research mainly focuses on developing content-based and collaborative filtering-based methods [2], [3]. Matrix factorization is one of the most representative traditional recommendation methods. Recently, motivated by the quick developments of deep learning, various neural recommendation methods have been developed in recent years [4]. Nevertheless, existing recommendation methods usually ignore the interactions between a user and the recommendation model. They cannot effectively capture the user's timely feedback to update the recommendation model, thus usually lead to unsatisfactory recommendation results.

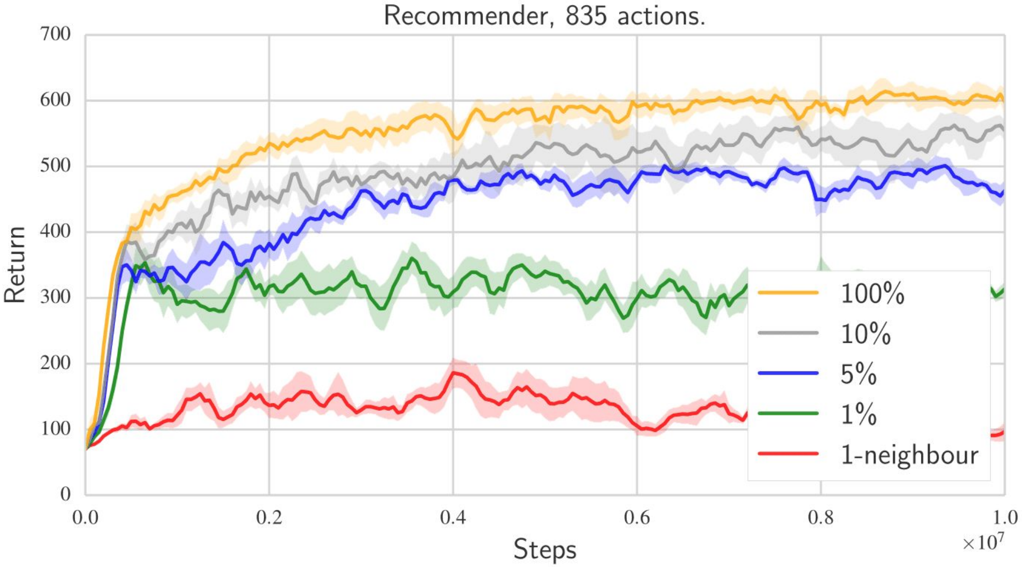
In general, the recommendation task can be modeled as such an interactive process - the user is recommended an item and then provides feedback (*e.g.*, skipping, click or purchase) for the recommendation model. In the next interaction, the recommendation model learns from the user's explicit/implicit feedback and recommends a new item to the user. From the user's point of view, an efficient interaction means helping users find the accurate items as soon as possible. From the model perspective, it is necessary to balance novelty, relevance, and diversity in the multi-turn of recommendations. The interactive recommendation approach has been successfully applied in real-world recommendation tasks. However, it often suffers from some problems, *e.g.*, cold-start [5] and data sparsity [6], as well as the challenges, *e.g.*, interpretability [7] and safety [8].

As a machine learning area focusing on how intelligent agents interact with the environment, Reinforcement Learning (RL) provides potential solutions to model the interactions between users and the agent. The recent success of RL has boosted research on artificial intelligence [9], [10]. In particular, Deep Reinforcement Learning (DRL) [11] has powerful representation learning and function approximation properties to address the challenges in artificial

https://arxiv.org/pdf/2109.10665.pdf

2015 – Deep RL in Large Discrete Action Spaces

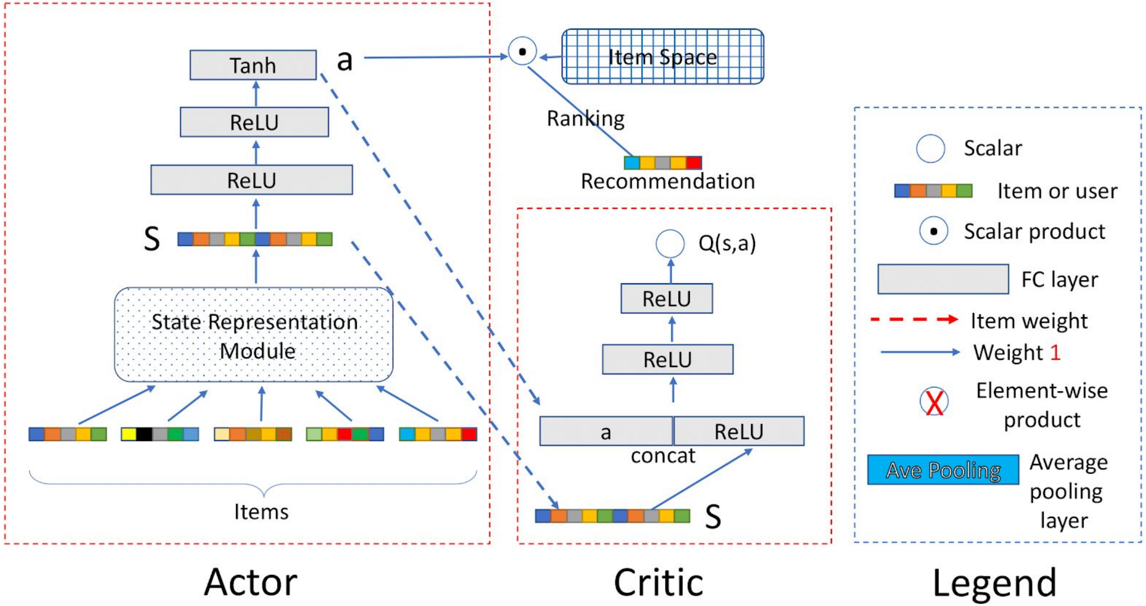




https://arxiv.org/abs/1512.07679

https://github.com/catalyst-team/dl-course/tree/master/week-12

2018 – Deep RL based Recommendation with Explicit User-Item Interactions Modeling



https://arxiv.org/abs/1810.12027

2019 – Top-K Off-Policy Correction for a REINFORCE

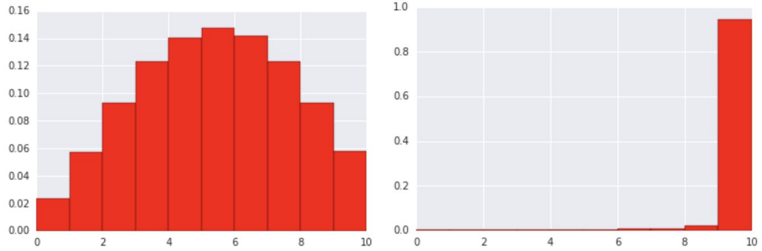


Figure 2: learned policy π_θ when behavior policy β is skewed to favor the actions with least reward, *i.e.*, $\beta(a_i) = \frac{11-i}{55}, \forall i = 1, \cdots, 10$. (left): without off-policy correction; (right): with off-policy correction.

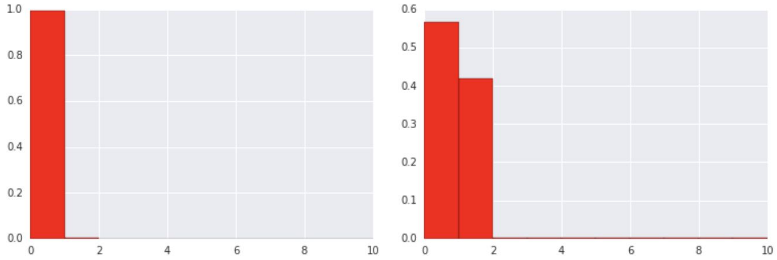
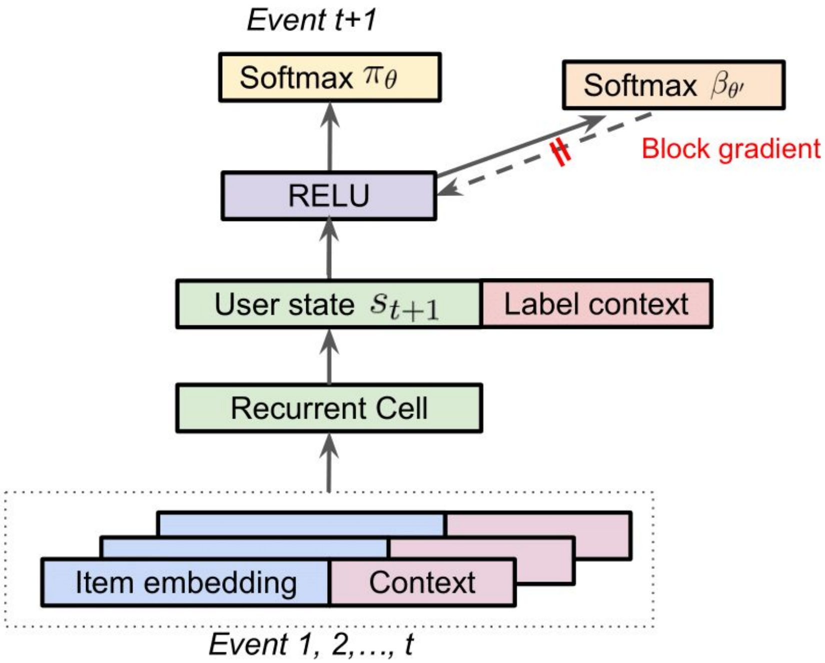
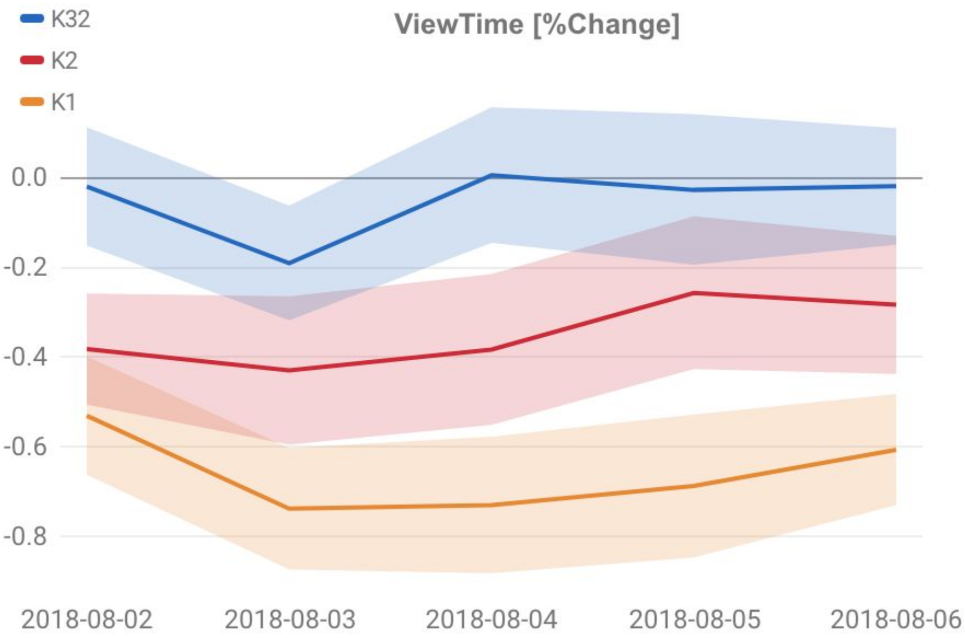


Figure 3: Learned policy π_θ (left): with standard off-policy correction; (right): with top-k correction for top-2 recommendation.



https://arxiv.org/abs/1812.02353

2019 – Top-K Off-Policy Correction for a REINFORCE RS

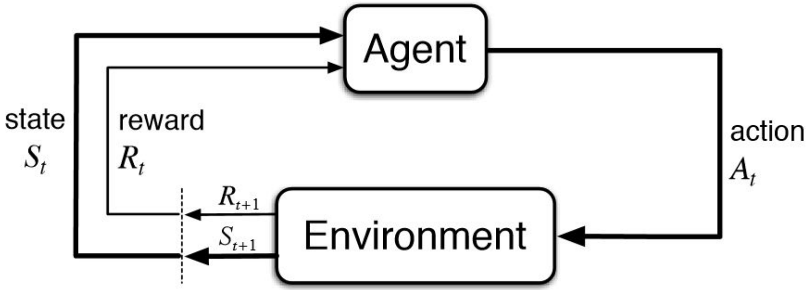


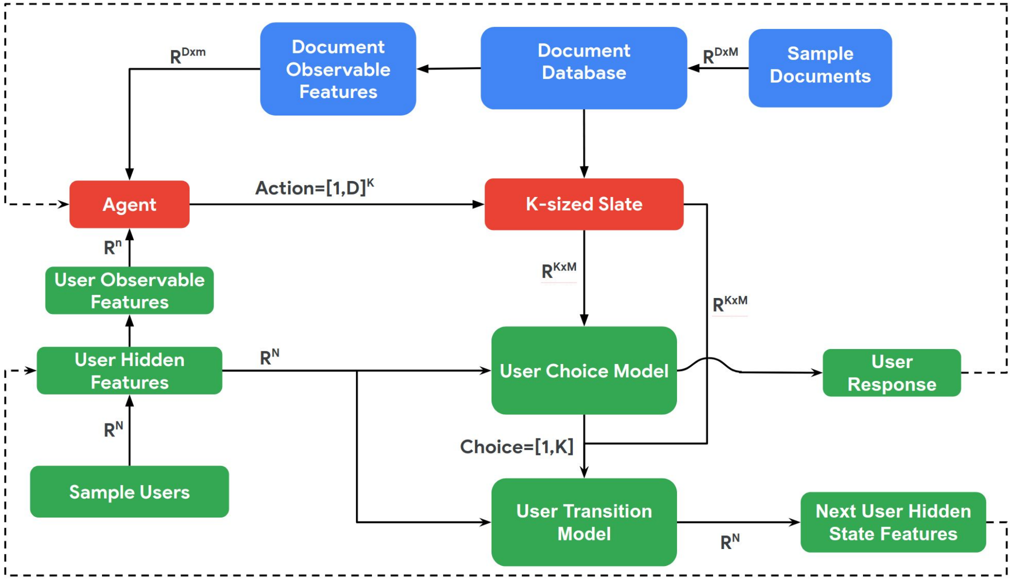
Top-K off-policy correction with varying K

- **off-policy correction**
  + 0.53% NumVideo
- **top-K off-policy correction**
  + 0.85% ViewTime,
  - 0.16% NumVideo
- **K hypertuning (K=8)**
  + 0.15% ViewTime

https://arxiv.org/abs/1812.02353

2019 – **RecSim**. A Configurable Simulation Platform for Recommender Systems





N - number of features that describe the user's hidden state

n - number of features that describe user's observed state

M - number of features describing document hidden state

m - number of features describing document observed state

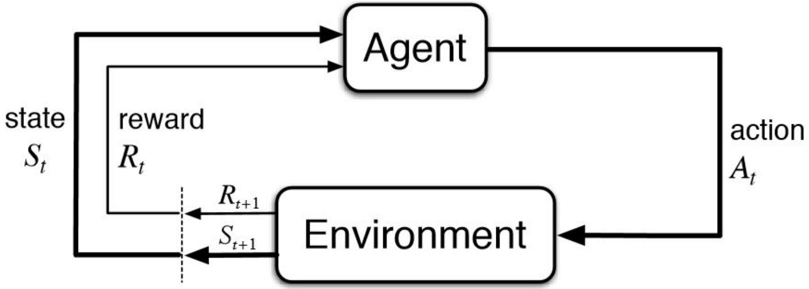
D - total number of documents in the corpus

K - size of slate

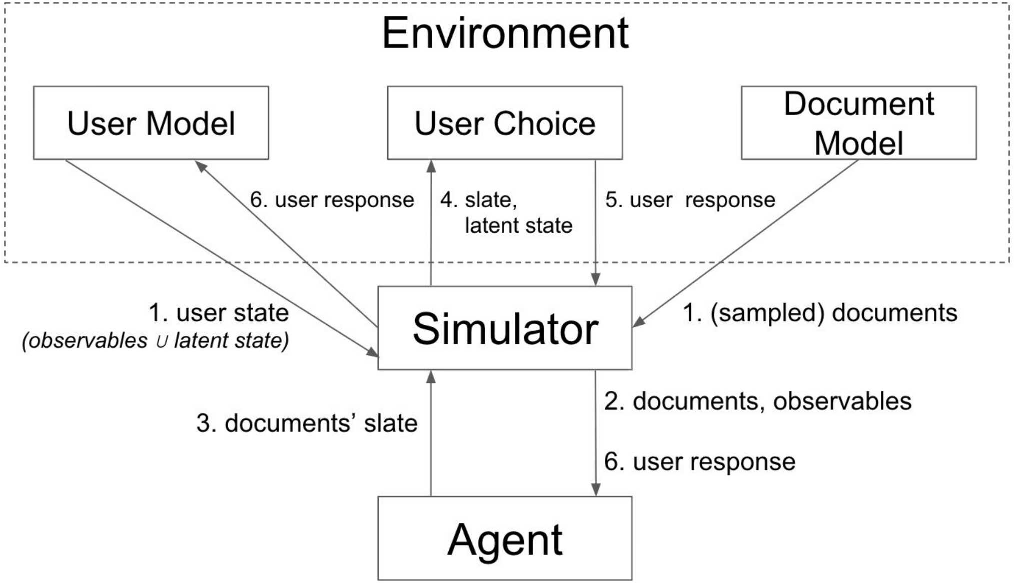
https://github.com/google-research/recsim

https://arxiv.org/abs/1909.04847

2019 – **RecSim**



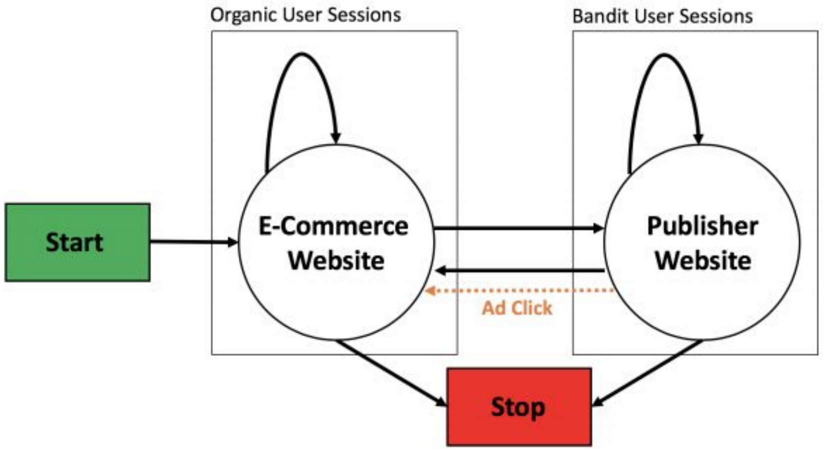
Agent
state
S_t
reward
R_t
action
A_t
R_{t+1}
S_{t+1}
Environment

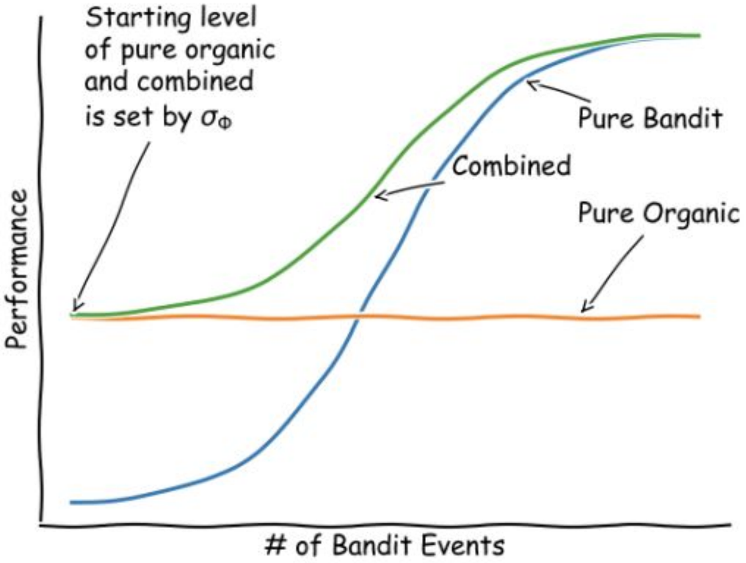


Environment
User Model
User Choice
Document Model
6. user response
4. slate, latent state
5. user response
1. user state (observables ∪ latent state)
Simulator
1. (sampled) documents
3. documents' slate
2. documents, observables
6. user response
Agent

Reco-gym by Criteo

-





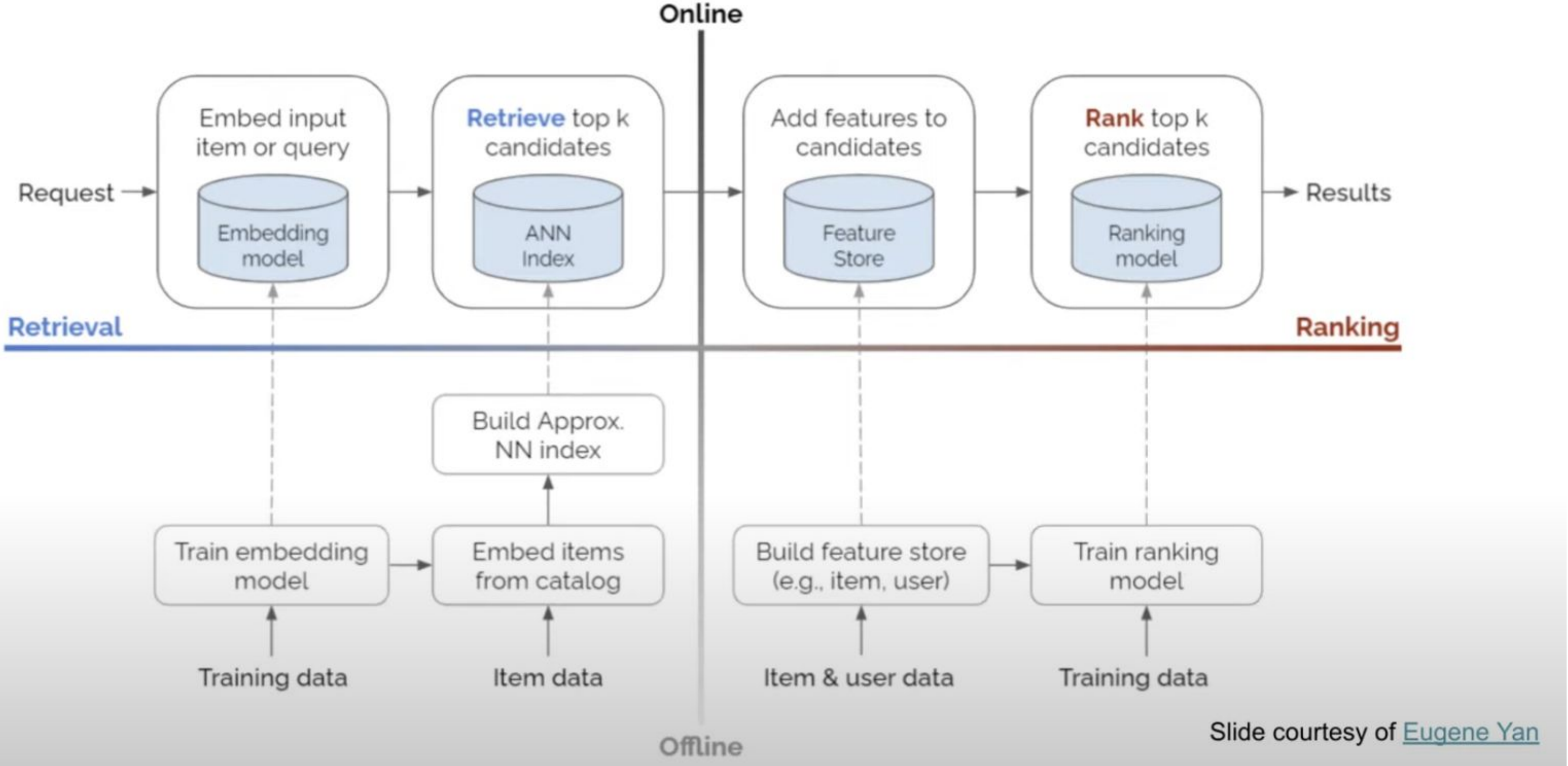
(a) Performance as #bandit events increases

https://github.com/criteo-research/reco-gym

https://arxiv.org/abs/1808.00720

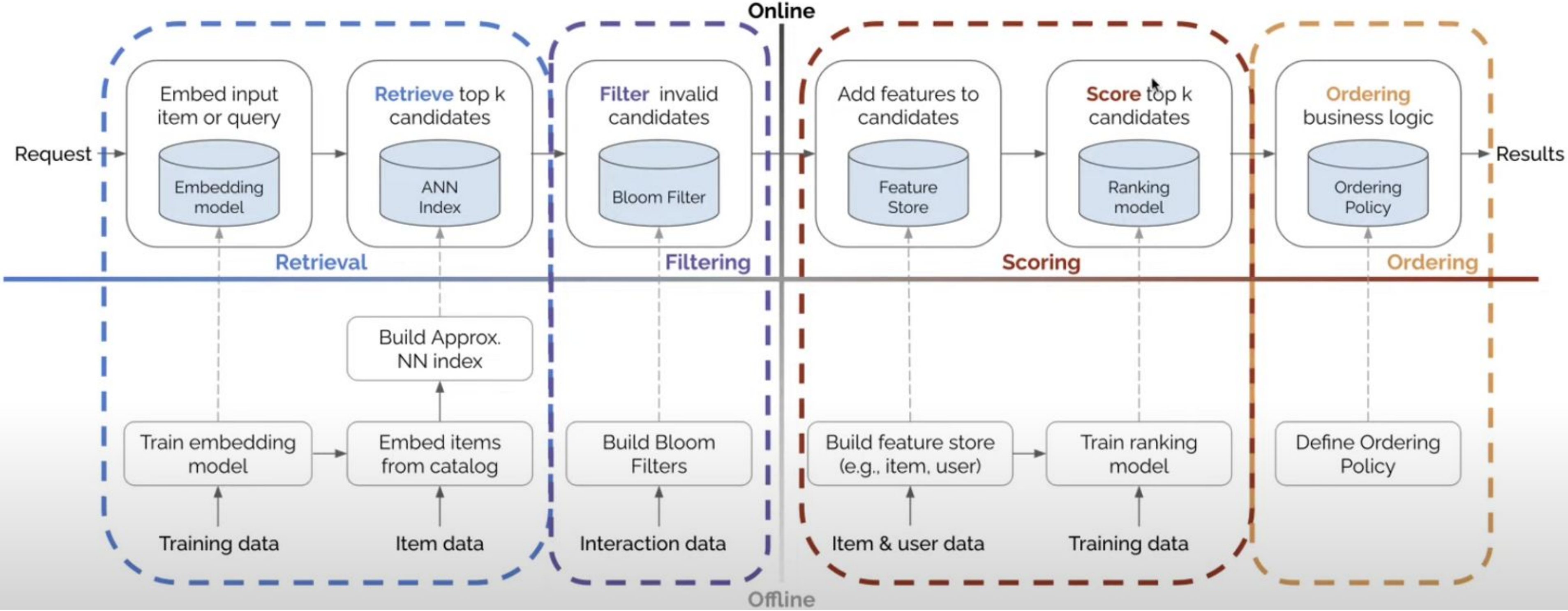
4. Архитектуры RecSys в индустрии

Two-stage Recommender Systems

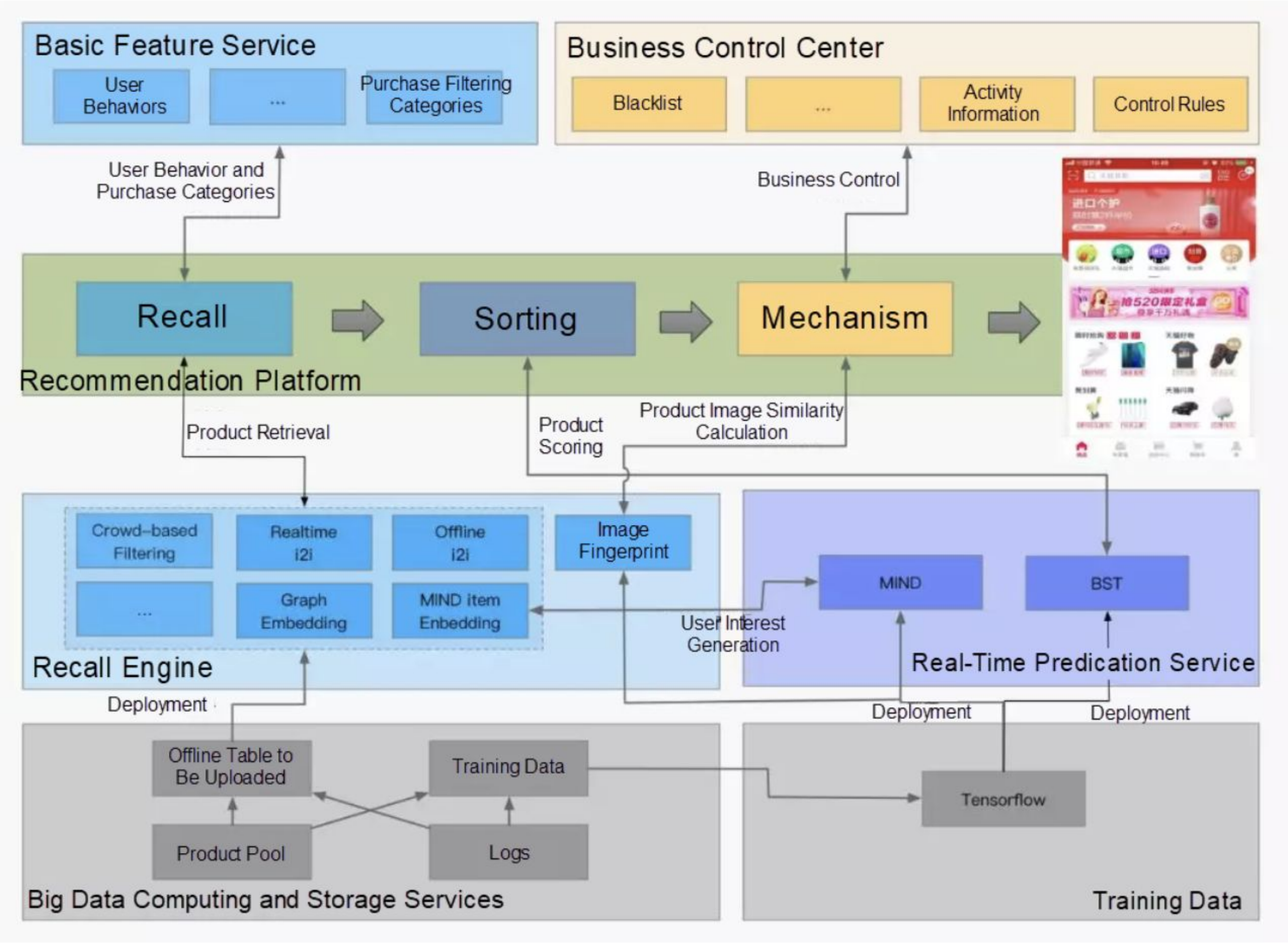


Slide courtesy of Eugene Yan

Four-stage Recommender Systems



Tmall (Alibaba)


Basic Feature Service
User Behaviors
...
Purchase Filtering Categories
Business Control Center
Blacklist
...
Activity Information
Control Rules
User Behavior and Purchase Categories
Business Control
Recall
Sorting
Mechanism
Recommendation Platform
Product Retrieval
Product Scoring
Product Image Similarity Calculation
Crowd-based Filtering
Realtime i2i
Offline i2i
Image Fingerprint
...
Graph Embedding
MIND item Enbedding
User Interest Generation
MIND
BST
Recall Engine
Real-Time Predication Service
Deployment
Deployment
Deployment
Offline Table to Be Uploaded
Training Data
Tensorflow
Product Pool
Logs
Big Data Computing and Storage Services
Training Data