Рекомендательные системы

2 курс магистратуры ФТИАД ФКН
2022/2023

Марина Ананьева, M.S.
email: ananyeva.me@gmail.com, mananeva@hse.ru
tg: @ananyevame

Course outline

Полезные ссылки

1. Wiki ФКН
   http://wiki.cs.hse.ru/RecSys_2022_2023

2. Github репозиторий
   https://github.com/anamarina/RecSys_course

3. Таблица с оценками
   ссылка на docs.google

4. Коммуникация - через учебные почты и чат в Телеграме

Оценивание

Final grade = 0.3 * Home Assignments + 0.15 * Article Summary + 0.15 * Weekly Quizzes + 0.4 * Exam (Case-study)

где:

- **Home Assignments** - 3 домашних работы в Jupyter Notebook (max 10 баллов за каждую).
- **Article Summary** - конспект/презентация статьи из предложенного списка с критическим анализом (без выступления на семинаре) (max 10 баллов).
- **Weekly Quizzes** - 7 квизов по мотивам материалов семинаров, которые сдаются перед началом следующего занятия в Google Forms (ариф.среднее за все квизы, max 10 баллов за каждый).
- **Exam** - письменный экзамен в формате решения case-study построения рекомендательной системы для бизнеса (max 10 баллов).

Course outline

Тема 1. Введение. Формализация задачи. Функции ранжирования, алгоритмов, метрики.

Тема 2. User и item-based подходы. Матричные факторизации.

Тема 3. Коллаборативная фильтрация.

Тема 4. Content-based подходы.

Тема 5. Гибридные подходы.

Тема 6. Автоэнкодеры и вариационные автоэнкодеры.

Тема 7. Sequential и session-based модели.

Тема 9. Context-aware модели.

Тема 10. Graph-based модели.

Тема 11. Knowledge-based модели.

Тема 12. RL-based модели.

Тема 13. Интерпретируемость моделей и объяснения рекомендаций.

Тема 14. Дизайн A/B/N тестов и тестирование рекомендаций.

Тема 15. Vanilla RecSys production service. Часть 1

Тема 16. Vanilla RecSYs production service. Часть 2

Introduction

Рекомендательные системы и персонализация

Персонализация - это набор различных техник и моделей для учета интересов и потребностей пользователей на основе имеющихся о них данных.

Рекомендательные системы - это тип систем информационной фильтрации, цель которых выдавать каждому пользователю наиболее релевантные ему объекты. Может рассматриваться как ML и не-ML задача, с персонализацией и без персонализацией.

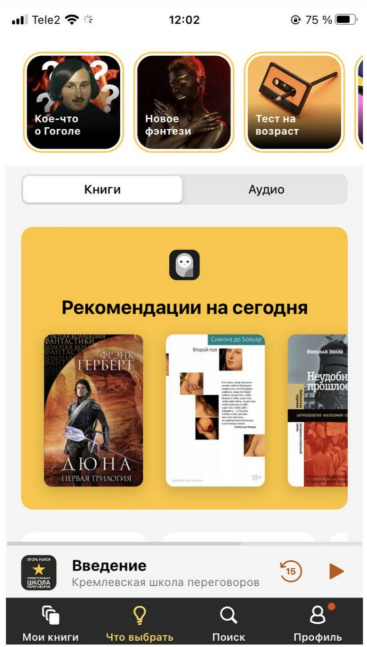
Зачем рекомендации пользователю?

- Более релевантные контент, акции, продукты, коммуникации в сервисе
- Снижение негатива (от лишних коммуникаций, нерелевантных предложений)
- Положительный customer experience (удобство использования сервиса, интерес)
- Экономия времени на поиск внутри сервиса
- Готовность и интерес к exploration, когда есть актуальный контекст и понятные ожидания

Зачем рекомендации пользователю?

- Более релевантный контент, акции, продукты, коммуникации.

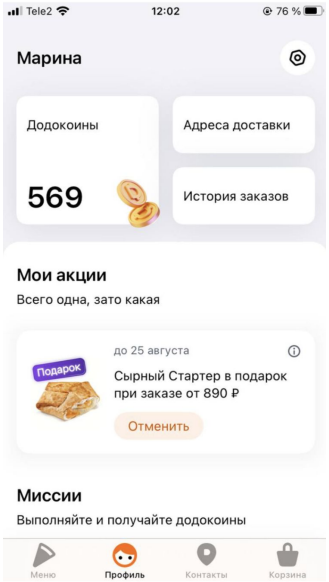
Примеры: рекомендации товаров в корзине Ozon, рекомендации книг в MyBook, персональные акции и промокоды в Dodo Pizza.



Зачем рекомендации пользователю?

- Снижение негатива (удержания лишних коммуникаций, нерелевантных предложений)

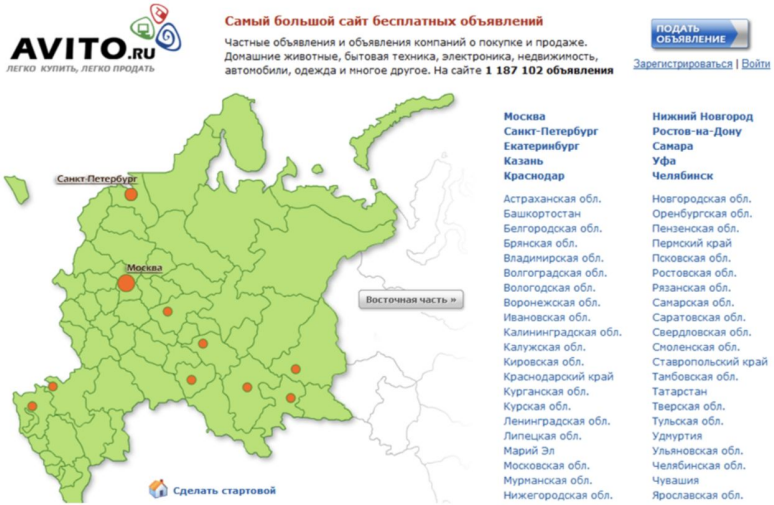
Примеры: отбор таргетной аудитории для маркетинговой акции, прогнозирование персонального времени отправки пуша, ранжирование лучших каналов для коммуникаций, ранжирование маркетинговых акций.

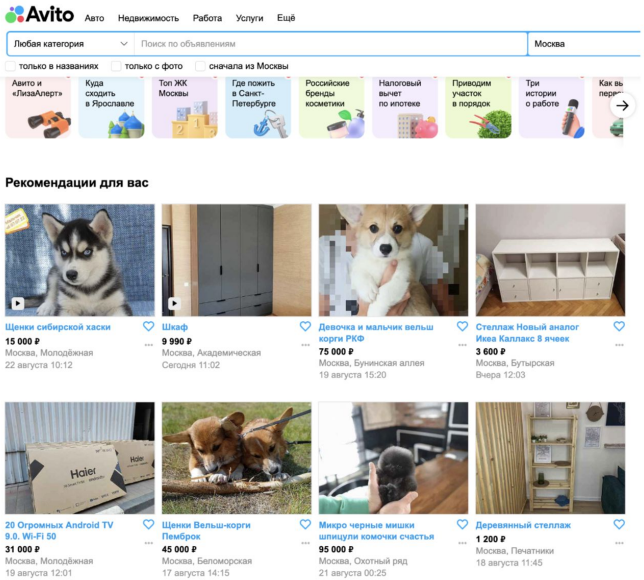


Зачем рекомендации пользователю?

- Положительный customer experience (повышается удобство использование сервиса)

Примеры: персональная лента объявлений на Avito на основе предыдущих запросов вместо гео-карты города на главной, персональная лента кэшбэков в Tinkoff на основе предыдущих транзакций.

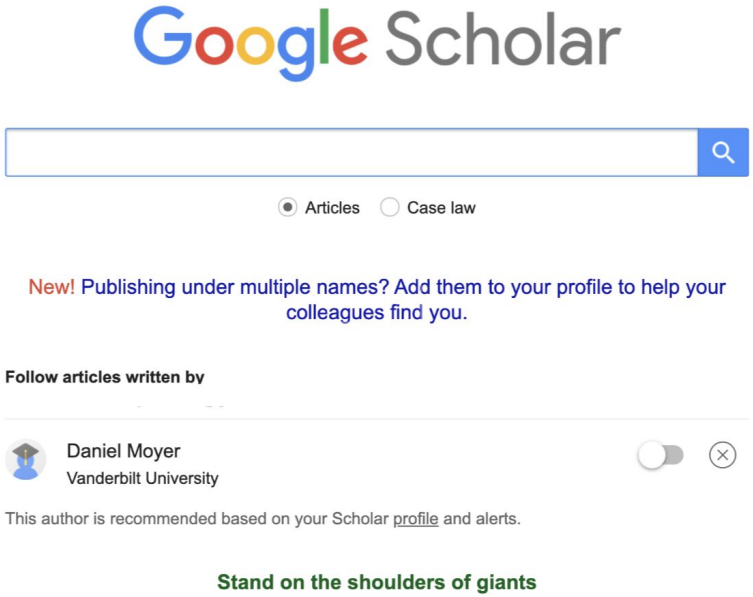




Зачем рекомендации пользователю?

- Экономия времени на поиск внутри приложения/платформы

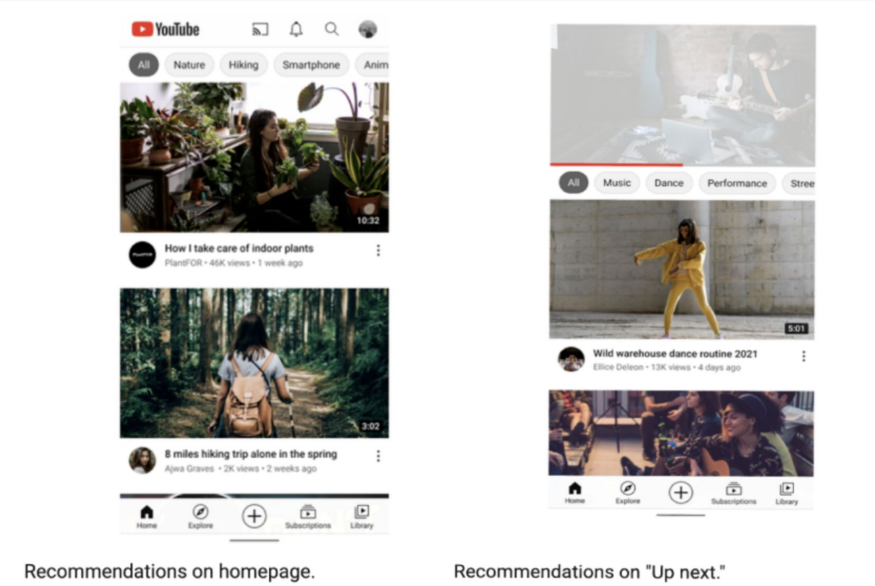
Примеры: рекомендации научных статей и подписки на авторов в Google Scholar



Зачем рекомендации пользователю?

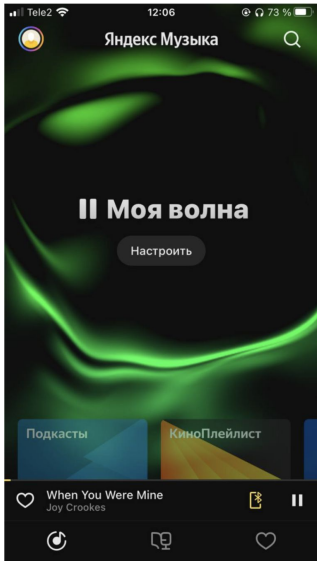
- Удовлетворение потребностей клиентов, готовых к exploration (чему-то новому и интересному через рекомендации сервиса)

Примеры: рекомендации видео на YouTube, рекомендации контента в TikTok/Instagram/etc., рекомендации музыки в Spotify и Яндекс.Музыке.



Recommendations on homepage.

Recommendations on "Up next."



Зачем рекомендации бизнесу?

Зачем рекомендации бизнесу?

- Увеличение целевых бизнес-метрик (конверсии в заказы, клики, сохранения в избранное, подписки)
- Экономия на лишних коммуникациях, костах на привлечение, x-sell, акции
- Наращивание лояльности, NPS
- Дополнительная монетизация (uplift к бизнес-метрикам, подписка за использование сервиса, плата за доп.фичи)

Постановка задачи ранжирования

X — множество объектов.

$X^l = \{x_1, \dots, x_l\}$ — обучающая выборка

$i \prec j$ на парах $(i, j) \in \{1, \dots, l\}^2$

Задача:

Построить ранжирующую функцию $a : X \to \mathbb{R}$ такую, что: $i \prec j \Rightarrow a(x_i) < a(x_j)$.

Линейная модель ранжирования:

$a(x; w) = \langle x, w \rangle$, где $x \mapsto (f_1(x), \dots, f_n(x)) \in \mathbb{R}^n$ — вектор признаков объекта x.

Пример 1. Ранжирование поисковой выдачи

D — коллекция текстовых документов.

Q — множество запросов.

$D_q \subseteq D$ — множество документов, найденных по запросу q.

$X = Q \times D$ — объектами являются пары (запрос, документ): $x \equiv (q, d), q \in Q, d \in D_q$.

Y — упорядоченное множество рейтингов.

$y : X \to Y$ — оценки релевантности, поставленные асессорами (экспертами): чем выше оценка $y(q, d)$, тем релевантнее документ d по запросу q.

Правильный порядок определен только между документами, найденными по одному и тому же запросу q: $(q, d) \prec (q, d') \Leftrightarrow y(q, d) < y(q, d')$.

Релевантные ответы запросу q — это список документов d, упорядоченных с помощью функции ранжирования $a(q, d)$.

Пример 2. Рекомендации пользователям

U — пользователи.

I — предметы (фильмы, книги и т.д.).

$X = U \times I$ — объектами являются пары (пользователь, предмет).

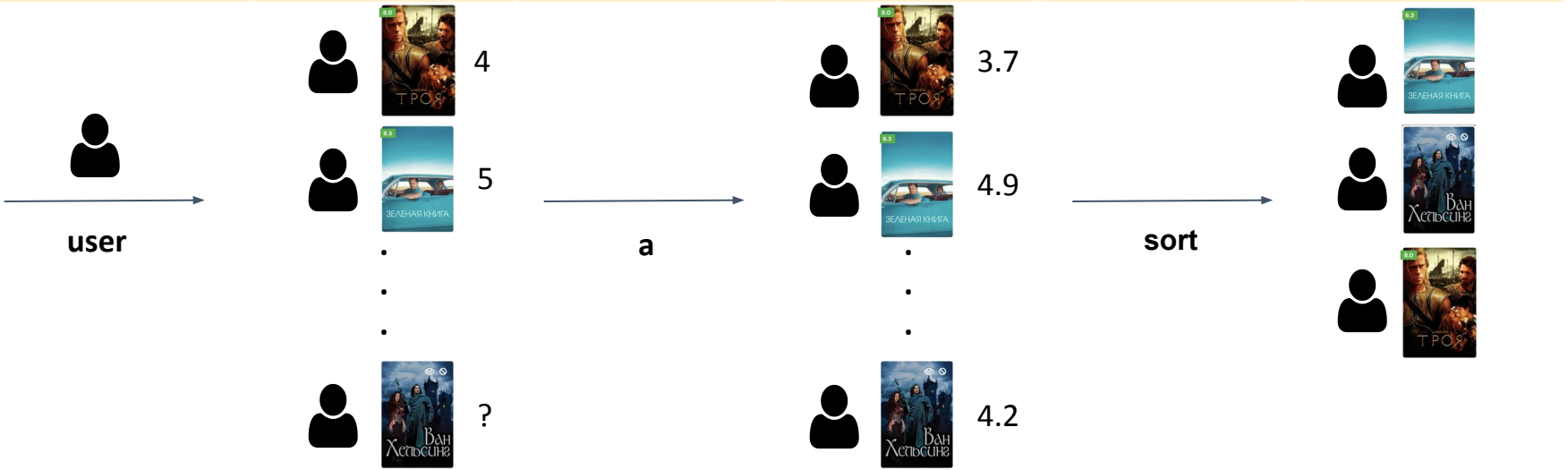
$$(u, i) \prec (u, i') \Leftrightarrow y(u, i) < y(u, i')$$

Рекомендация пользователю u — это список предметов i, упорядоченный с помощью функции ранжирования $a(u, i)$.

Пример 2. Рекомендации пользователям

$$R = \{(u, i, r_{iu})\}$$

i-тый пользователь	(user, item, rating)	применение алгоритма	скор алгоритма	сортировка по скору (или рангу)	top-k рекомендаций *i*-му пользователю (перестановка)



Explicit & implicit данные

Explicit feedback / Явные данные - тип данных, когда известны оценки пользователей в результате взаимодействия с объектом. Например, датасет с оценками пользователей по 5-балльной шкале, насколько им понравились просмотренные фильмы.

Implicit feedback / Неявные данные - тип данных, когда нам неизвестно насколько понравился или не понравился объект после того, как пользователь провзаимодействовал с ним.

Матрица интеракций - это матрица, у которой по одной оси - id пользователей, по второй оси - id объектов, а на пересечении - оценка/наличие взаимодействия.

	item #1	**....**	**item #N**
user #1	0	1	...
....	1	0	...
user #M	...	...	1

implicit ?

	item #1	**....**	**item #N**
user #1	0	3	...
....	5	0	...
user #M	...	...	1

explicit ?

Explicit & implicit данные

Explicit feedback / Явные данные - тип данных, когда известны оценки пользователей в результате взаимодействия с объектом. Например, датасет с оценками пользователей по 5-балльной шкале, насколько им понравились просмотренные фильмы.

Implicit feedback / Неявные данные - тип данных, когда нам неизвестно насколько понравился или не понравился объект после того, как пользователь провзаимодействовал с ним.

```
'''
Пример 1. По 5 клиентам магазина известна история покупок из 10 уникальных товаров.
1 - товар был куплен, 0 - не был.
'''


import numpy as np
from numpy import random

random.randint(low=0, high=2, size=(5,10))
```

```
array([[0, 0, 0, 0, 0, 0, 1, 1, 0, 0],
       [0, 0, 0, 1, 0, 0, 0, 0, 0, 1],
       [1, 0, 0, 0, 0, 1, 1, 1, 0, 1],
       [1, 0, 0, 0, 0, 0, 1, 1, 1, 0],
       [0, 1, 1, 0, 0, 0, 1, 0, 0, 0]])
```

Explicit & implicit данные

Explicit feedback / Явные данные - тип данных, когда известны оценки пользователей в результате взаимодействия с объектом. Например, датасет с оценками пользователей по 5-балльной шкале, насколько им понравились просмотренные фильмы.

Implicit feedback / Неявные данные - тип данных, когда нам неизвестно насколько понравился или не понравился объект после того, как пользователь провзаимодействовал с ним.

```
'''
Пример 2. По 7 пользователям и 5 постам в соц.сети известна история реакций.
0 - посту поставлен дислайк, 1 - лайк, np.nan - пользователь не ставил реакцию у поста.
'''


num_nans = 10
matrix = np.random.randint(0, 2, (7, 5)).astype(float)
matrix.ravel()[np.random.choice(matrix.size, num_nans)] = np.nan
matrix
```

```
array([[ 0.,  1., nan,  0.,  0.],
       [ 0.,  0.,  1., nan,  1.],
       [ 1.,  0.,  0.,  1.,  0.],
       [ 1.,  1.,  1.,  0.,  1.],
       [ 0., nan,  1.,  1.,  0.],
       [ 1., nan,  0., nan,  0.],
       [nan, nan, nan, nan,  0.]])
```

Explicit & implicit данные

Explicit feedback / Явные данные - тип данных, когда известны оценки пользователей в результате взаимодействия с объектом. Например, датасет с оценками пользователей по 5-балльной шкале, насколько им понравились просмотренные фильмы.

Implicit feedback / Неявные данные - тип данных, когда нам неизвестно насколько понравился или не понравился объект после того, как пользователь провзаимодействовал с ним.

```
'''
Пример 3. По 3 пользователям и 8 ресторанам известна история отзывов на рестораны и их рейтинг по 5-балльной шкале.
1 - самая низкая оценка, ресторан очень не понравился, 5 - отличная оценка, np.nan - нет рейтинга и отзыва.
'''


matrix = np.random.randint(1, 6, (3, 8)).astype(float)
matrix.ravel()[np.random.choice(matrix.size, num_nans)] = np.nan
matrix
```

```
array([[ 1.,  3.,  3.,  4., nan,  5., nan, nan],
       [nan, nan, nan,  3.,  4.,  4.,  4.,  2.],
       [nan, nan,  2.,  1.,  4., nan,  5.,  4.]])
```

Explicit & implicit данные

Explicit feedback / Явные данные - тип данных, когда известны оценки пользователей в результате взаимодействия с объектом. Например, датасет с оценками пользователей по 5-балльной шкале, насколько им понравились просмотренные фильмы.

Implicit feedback / Неявные данные - тип данных, когда нам неизвестно насколько понравился или не понравился объект после того, как пользователь провзаимодействовал с ним.

```
'''
Пример 4. По 5 клиентам есть градация по разным типам действий.
0.1 - просмотр
0.3 - просмотр + клик
0.5 - просмотр + клик + лайк
0.7 - просмотр + клик + комментарий
1 - просмотр + клик + лайк + комментарий
'''
```

```python
# В каком виде лучше хранить такую матрицу, когда у нас очень много нулей?

from scipy.sparse import coo_matrix, csr_matrix

import sys

matrix = random.randint(low=0, high=2, size=(10,10))

print(matrix)
print(f'Sparse матрица: {sys.getsizeof(csr_matrix(matrix))} байтов',
      f'Обычный np.ndarray: {sys.getsizeof(matrix)} байтов',
      sep='\n')
```

```
[[1 1 1 1 0 1 0 1 0 0]
 [0 0 0 0 1 0 1 1 1 0]
 [0 0 1 0 0 1 0 1 1 1]
 [1 1 0 1 1 1 1 0 0 0]
 [0 0 0 0 1 1 0 0 0 1]
 [0 0 1 1 0 1 1 1 0 1]
 [1 0 0 1 0 1 1 1 0 0]
 [1 0 0 0 1 0 1 0 0 1]
 [0 1 1 1 0 1 1 0 1 1]
 [1 0 0 0 0 0 0 0 0 0]]
Sparse матрица: 64 байтов
Обычный np.ndarray: 920 байтов
```

Функции ранжирования

Функции ранжирования

1. **Pointwise (точечные)**

Например, решение через задачу регрессии или классификации.

$$\sum_{i=1}^{l} (a(x_i) - y(x_i))^2 \to min$$

Функции ранжирования

1.**Pointwise (точечные)**

2. **Pairwise (попарные)**

$$\sum_{x_i<x_j}[a(x_j)-a(x_i)<0]\rightarrow min$$

Функции ранжирования

1.**Pointwise (точечные)**

2. **Pairwise (попарные)**

$$\sum_{x_i<x_j}[a(x_j)-a(x_i)<0]\rightarrow min$$

$$\sum_{x_i<x_j}L(a(x_j)-a(x_i))\rightarrow min$$

Функции ранжирования

1.**Pointwise (точечные)**

2. **Pairwise (попарные)**

3. **Listwise (списочные)**

RankNet:

$$\omega := \omega + \eta \frac{1}{1+exp(\langle \omega, x_j - x_i \rangle)}(x_j - x_i)$$

LambdaRank:

$$\omega := \omega + \eta \frac{1}{1+exp(\langle \omega, x_j - x_i \rangle)}(x_j - x_i)|\Delta nDCG_{ij}|(x_j - x_i)$$

Функции ранжирования

- **BPR** (Bayesian Personalised Ranking, pairwise)
- **WARP** (Weighted Approximate-Rank Pairwise)
- **RankNET** (pairwise)
- **LambdaRANK** (pairwise/listwise)
- **LambdaMART** (pairwise/listwise)

BPR: Bayesian Personalized Ranking

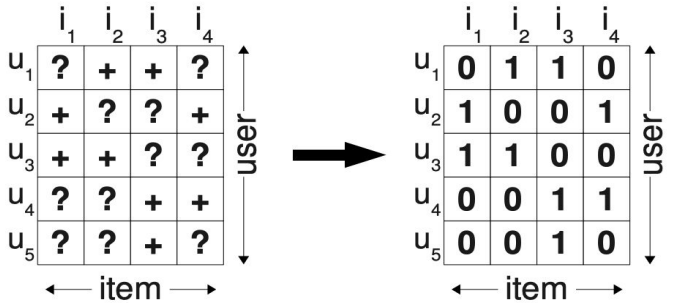
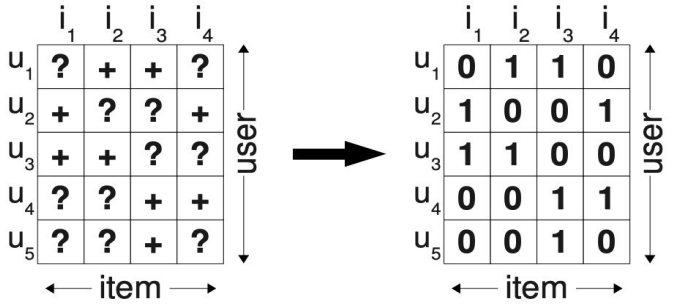


Рис. 1: Бинаризация implicit данных, где отсутствие интеракции кодируется как отрицательный пример.

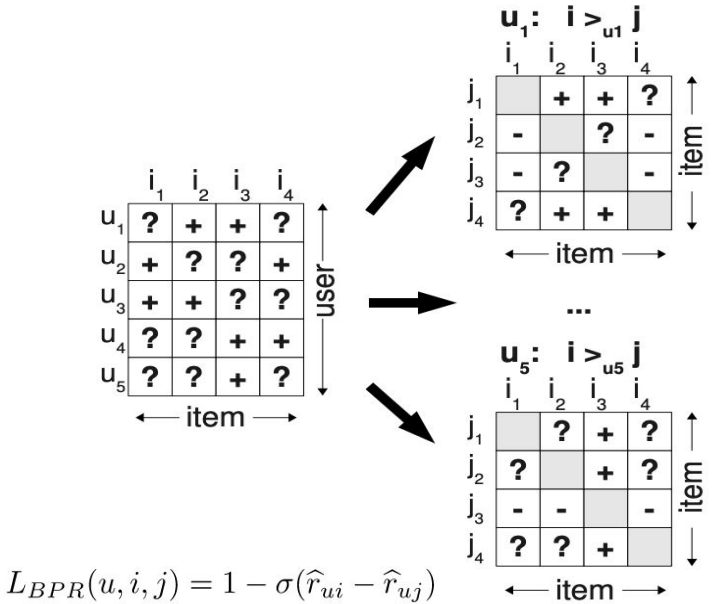
Rendle, S., Freudenthaler, C., Gantner, Z. and Schmidt-Thieme, L., 2009, June. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence* (pp. 452-461).

BPR: Bayesian Personalized Ranking



$$positive >_{user} unknown$$

Рис. 1: Бинаризация implicit данных, где отсутствие интеракции кодируется как отрицательный пример.



$$L_{BPR}(u,i,j) = 1 - \sigma(\widehat{r}_{ui} - \widehat{r}_{uj})$$

Рис. 2: Попарные предпочтения пользователю по объектам для BPR. + означает, что пользователь предпочитает i объект j-му, знак минуса - наоборот.

Rendle, S., Freudenthaler, C., Gantner, Z. and Schmidt-Thieme, L., 2009, June. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence* (pp. 452-461).

WARP:Weighted Approximate-Rank Pairwise*

- Тоже смотрим на тройки: **(user, positive_item, negative_item)**
- Делаем шаг оптимизации, только если попалась дефектная пара - алгоритм дал выше скор отрицательному примеру, чем положительные
- Добавляем adaptive learning rate: штрафуем больше, если алгоритм ошибается почти сразу.

Algorithm 1 K-OS algorithm for picking a positive item.

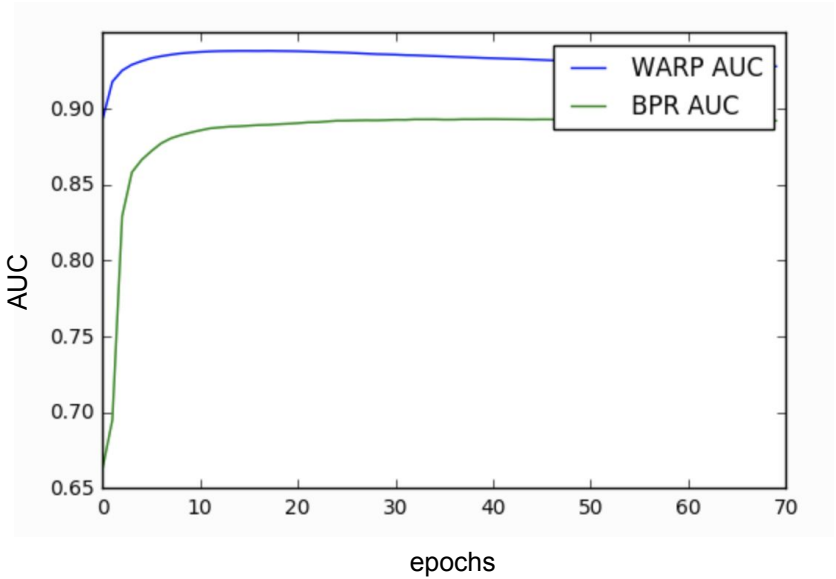
We are given a probability distribution P of drawing the i^{th} position in a list of size K. This defines the choice of loss function.
Pick a user u at random from the training set.
Pick $i = 1, \ldots, K$ positive items $d_i \in \mathcal{D}_u$.
Compute $f_{d_i}(u)$ for each i.
Sort the scores by descending order, let $o(j)$ be the index into d that is in position j in the list.
Pick a position $k \in 1, \ldots, K$ using the distribution P.
Perform a learning step using the positive item $d_{o(k)}$.

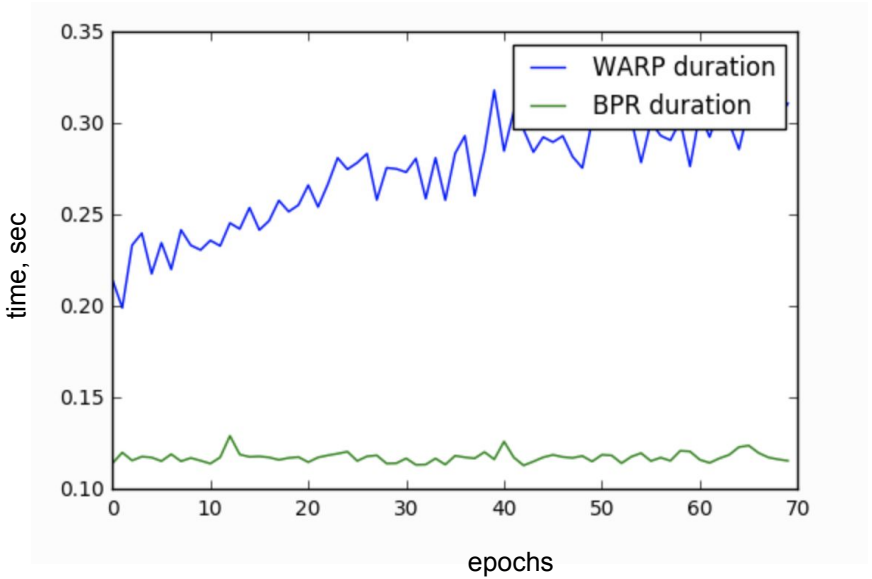
Algorithm 2 K-OS WARP loss

Initialize model parameters (mean 0, std. deviation $\frac{1}{\sqrt{m}}$).
repeat
  Pick a positive item d using Algorithm 1.
  Set $N = 0$.
  repeat
    Pick a random item $\bar{d} \in \mathcal{D} \setminus \mathcal{D}_u$.
    $N = N + 1$.
  until $f_{\bar{d}}(u) > f_d(u) - 1$ or $N \geq |\mathcal{D} \setminus \mathcal{D}_u|$
  if $f_{\bar{d}}(y) > f_d(u) - 1$ **then**
    Make a gradient step to minimize:
    $\Phi(\frac{|\mathcal{D} \setminus \mathcal{D}_u|}{N}) \max(0, 1 + f_{\bar{d}}(u) - f_d(u))$.
    Project weights to enforce constraints, e.g. if $||V_i|| > C$ then set $V_i \leftarrow (CV_i)/||V_i||$.
  end if
until validation error does not improve.

Weston, J., Yee, H. and Weiss, R.J., 2013, October. Learning to rank recommendations with the k-order statistic loss. In *Proceedings of the 7th ACM Conference on Recommender Systems* (pp. 245-248).

WARP vs. BPR*





Обычно, WARP лучше по качеству…

но более медленный =(

*https://making.lyst.com/lightfm/docs/examples/warp_loss.html

Overview текущих подходов

Одна из таксономий

- **General** подходы (Matrix Factorization, Factorization Machines, Collaborative Filtering, user/item similarity based methods …)
- **Content-based** models
- **Context-based** models
- **Sequential** based models
- **RL** for RecSys
- **Hybrid** models

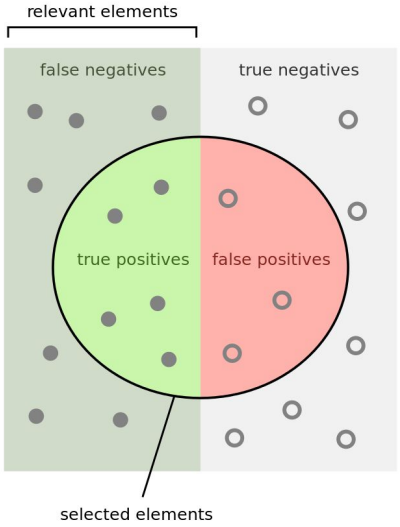
Метрики оценки качества ранжирования

Зоопарк метрик качества ранжирования

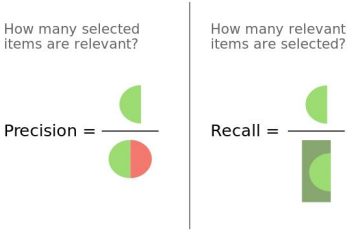
- **Hit Rate (Hit ratio)**
- **Precision@k**
- **Recall@k**
- **AP@k, MAP@k, MNAP@k**
- **DCG@k, NDCG@k**
- **MRR**

и другие :)

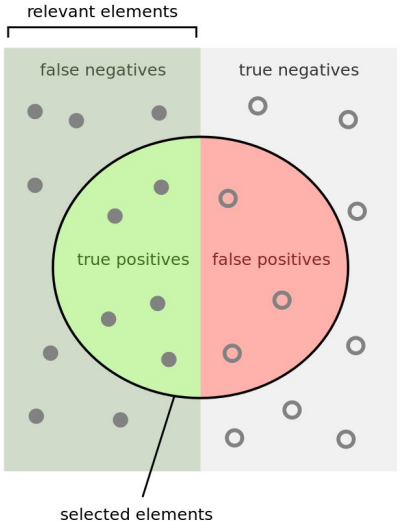
Hitrate, Precision@k, Recall@k

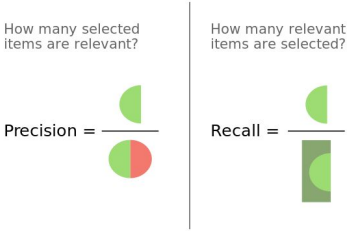


$$Hit\ Rate = \frac{\#\ hits}{\#\ hits + \#\ misses}$$



Hitrate, Precision@k, Recall@k



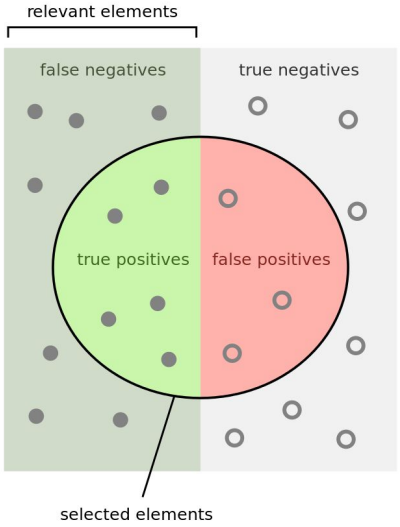


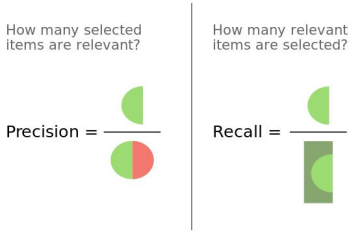
$$Hit\ Rate = \frac{\#\ hits}{\#\ hits + \#\ misses}$$

Precision – ?
Recall – ?

Эти метрики зависят от количества рекомендаций для каждого пользователя?

Hitrate, Precision@k, Recall@k



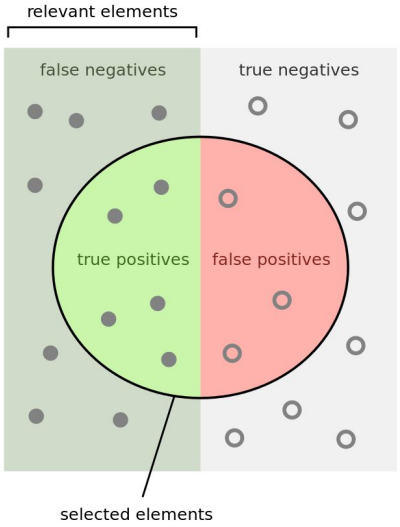


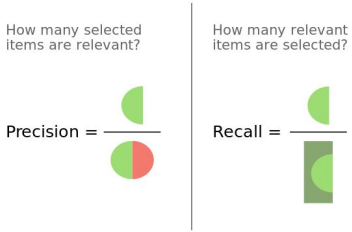
$$Hit\ Rate = \frac{\#\ hits}{\#\ hits + \#\ misses}$$

Precision – сколько из рекомендованных объектов оказались релевантными.
Recall – сколько из релевантных для пользователя объектов мы порекомендовали.

Precision@k, Recall@k – фиксируем для каждого пользователя топ k рекомендованных объектов, считаем метрику по бинарному таргету (0 – нерелевантная рекомендация, 1 – релевантная).

Hitrate, Precision@k, Recall@k





$$Hit\ Rate = \frac{\#\ hits}{\#\ hits + \#\ misses}$$

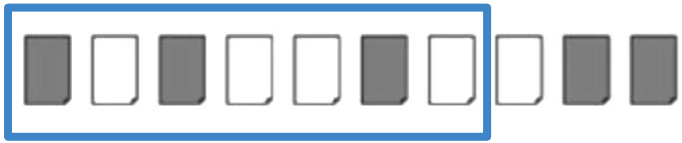
Precision – сколько из рекомендованных объектов оказались релевантными.
Recall – сколько из релевантных для пользователя объектов мы порекомендовали.

Precision@k, Recall@k – фиксируем для каждого пользователя топ k рекомендованных объектов, считаем метрику по бинарному таргету (0 – нерелевантная рекомендация, 1 – релевантная).

Посчитаем precision@k, recall@k

- документы выдачи по запросу, белым цветом отмечены нерелевантные

Посчитаем precision@k, recall@k



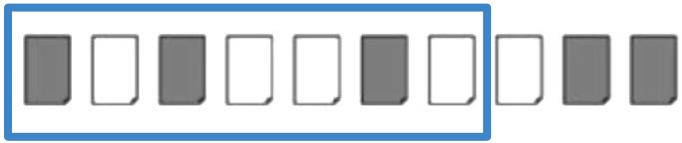
- документы выдачи по запросу, белым цветом отмечены нерелевантные

Пусть k=7

$$precision@7 = \frac{\#\ predicted\ relevant}{k\ predicted}$$

$$recall@7 = \frac{\#\ predicted\ relevant}{\#\ relevant}$$

Посчитаем precision@k, recall@k



- документы выдачи по запросу, белым цветом отмечены нерелевантные

Пусть k=7

$$precision@7 = \frac{\#\ predicted\ relevant}{k\ predicted} = \frac{3}{7}$$

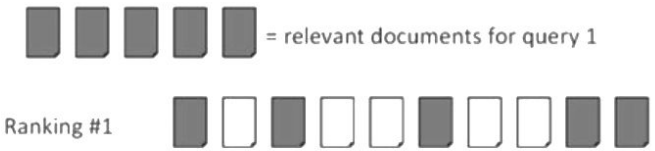
$$recall@7 = \frac{\#\ predicted\ relevant}{\#\ relevant} = \frac{3}{5}$$

Average precision @ k (AP@k)

- Учитывает порядок рекомендованных объектов в топе k.

$$ap@K = \frac{1}{K}\sum_{k=1}^{K} r^{true}(\pi^{-1}(k)) \cdot p@k$$

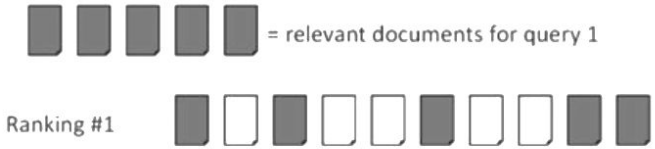
$$ap@k = \frac{1}{k}\sum_{i=1}^{k} P@i$$



Average precision @ k (AP@k)

- Учитывает порядок рекомендованных объектов в топе k.

$$ap@k = \frac{1}{k}\sum_{i=1}^{k} P@i$$



$$ap@3 = \frac{p@1 + p@2 + p@3}{3} \approx 0.77$$

Mean average precision (MAP@k)

Считаем AP@K для каждого объекта и усредняем:

$$map@K = \frac{1}{N} \sum_{j=1}^{N} ap@K_j$$

Как вы думаете, что такое MNAP@K?

Mean average precision (MAP@k)

$$map@K = \frac{1}{N}\sum_{j=1}^{N} ap@K_j$$

$$\text{MNAP@20} = \frac{1}{|U|}\sum_{u \in U}\frac{1}{\min(n_u, 20)}\sum_{i=1}^{20} r_u(i)p_u@i$$

$$p_u@k = \frac{1}{k}\sum_{i=1}^{k} r_u(i)$$

- **r_u(i)** — потребил ли пользователь u контент, предсказанный ему на месте i (1 либо 0)
- **n_u** — количество элементов, которые пользователь потребил за тестовый период
- **U** — множество тестовых пользователей

Discounted Cumulative Gain (DCG@k)

Чем выше релевантные элементы в списке рекомендаций, тем лучше.

$$CG@K = \sum_{k=1}^{K} r^{true}(\pi^{-1}(k))$$

$$DCG@K = \sum_{k=1}^{K} \frac{r^{true}(\pi^{-1}(k))}{\log_2(k+1)}.$$

Normalized Discounted Cumulative Gain (DCG@k)

$$nDCG@K = \frac{DCG@K}{IDCG@K}$$

где IDCG@K - это идеальное (максимальное) значение DCG@K:

$$IDCG@K = \sum_{k=1}^{K} \frac{1}{\log_2(k+1)}$$

NDCG@k

Пример вычисления DCG и nDCG:

Дано множество документов, где каждый документ оценивается от 3 до 0, где 3 — очень релевантен, а 0 — не релевантен. Пусть таким множеством будет $S = \{D_1, D_2, D_3, D_4, D_5, D_6\}$, где оценка релевантности по опросу пользователей задается(в том же порядке) множеством $R = \{3, 2, 3, 0, 1, 2\}$.

Тогда $DCG@6 = \sum_{i=1}^{6} \frac{rel_i}{log(i+1)} = 3 + 1.262 + 1.5 + 0 + 0.387 + 0.712 = 6.861.$

i	rel_i	$log(i+1)$	$\frac{rel_i}{log(i+1)}$
1	3	1	3
2	2	1.585	1.262
3	3	2	1.5
4	0	2.322	0
5	1	2.585	0.387
6	2	2.807	0.712

NDCG@k

Идеальный порядок оценок релевантности $Ideal = \{3, 3, 2, 2, 1, 0\}$. DCG для данного множества будет следующим:

$$maxDCG@6 = \sum_{i=1}^{6} \frac{rel_i}{log(i+1)} = 3 + 1.893 + 1 + 0.861 + 0.387 + 0 = 7.141.$$

i	rel_i	$log(i+1)$	$\frac{rel_i}{log(i+1)}$
1	3	1	3
2	3	1.585	1.893
3	2	2	1
4	2	2.322	0.861
5	1	2.585	0.387
6	0	2.807	0

Итого $nDCG@6 = \frac{DCG@6}{maxDCG@6} = \frac{6.861}{7.141} = 0.961.$

Измерение качества рекомендаций

Diversity (разнообразие) - число рекомендаций из разных категорий, степень различия рекомендаций между сессиями пользователя, различие рекомендаций между пользователями.

Novelty (новизна) - сколько среди рекомендаций новых для пользователей объектов.

Coverage (покрытие) - доля объектов, которые хотя бы раз побывали среди топа рекомендаций

Serendipity (догадливость) - способность угадывать неожиданные нетривиальные предпочтения пользователя.

- Можно выбрать один из них!
- Зачем выбирать, если можно оптимизировать линейную комбинацию из них!

Рекомендованные источники

- ACM RecSys 2021 tutorials and workshops
- Лекция Евгения Соколова по задаче ранжирования. Да и в целом курс
- Лекция Евгения Соколова по memory-based подходам, коллаборативной фильтрации, MF.
- Воронцов про коллаборативную фильтрацию
- Лекция Сергея Николенко про подходы для рекомендательных систем
- ИТМО Wiki. Задача ранжирования
- Библиотека RecBole с 50+ алгоритмами RecSys
- Библиотека Microsoft recommenders
- Библиотека implicit в частности с оптимальной реализацией ALS. Оригинальная статья по ALS
- Библиотека LightFM с реализацией этого алгоритма по статье. Это контентная модель - то есть, можно обучать как ALS, а можно еще добавлять признаки по пользователям и объектам.
- Обучение градиентного бустинга для задачи рекомендаций. Catboost ranking task and metrics
- Рекомендации в Okko: как заработать сотни миллионов, перемножив пару матриц