

Валидация и метрики

Даниил Потапов

Руководитель направления анализа данных

MTC BigData



План

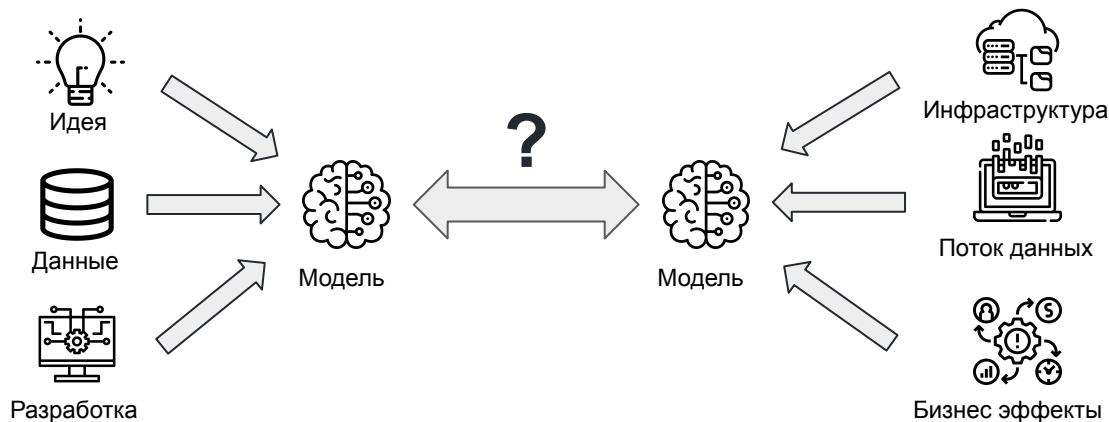
- Валидация
- Метрики
 - Какие бывают
 - Как считаем
- Практические заметки

План

- Валидация
- Метрики
- Практические заметки

Что такое валидация

Валидация - это процесс оценки качества модели. Ее основная задача - понять, насколько модель будет полезна. Это касается как математических метрик (точность, стабильность), так и инженерных (скорость работы, потребляемая память). Поэтому процесс валидации должен максимально воспроизводить условия, в которых модель будет использоваться. Это касается как каких-либо ограничений (смещение в данных, время инференса), так и возможностей (переобучение модели, конвертация модели в оптимизированный вид)



Основные вопросы

Что хотим от модели?

- Предсказанное значение для пары пользователь-объект
- Ранжирование объектов для пользователя

Как будет использоваться модель?

- Рекомендации будут считаться раз в какой-то период
- Онлайн-рекомендации

Какие технические ограничения?

- Время на обучение и время на построение рекомендаций
- Доступность данных в онлайн-режиме

Какие есть особенности у задачи?

- Рекомендуем ли для пользователя объекты, с которым он уже взаимодействовал
 - В контентных системах (книги, фильмы и тд) скорее всего нет
 - В продуктовом ритейле, наоборот, вероятнее всего да
- Cold start - как много появится пользователей или объектов, для которых не известна история по взаимодействиям
- Как много взаимодействий по пользователям есть в данных

Схемы оффлайн валидации

Для разбиения данных на train/test/validation есть два класса подходов:

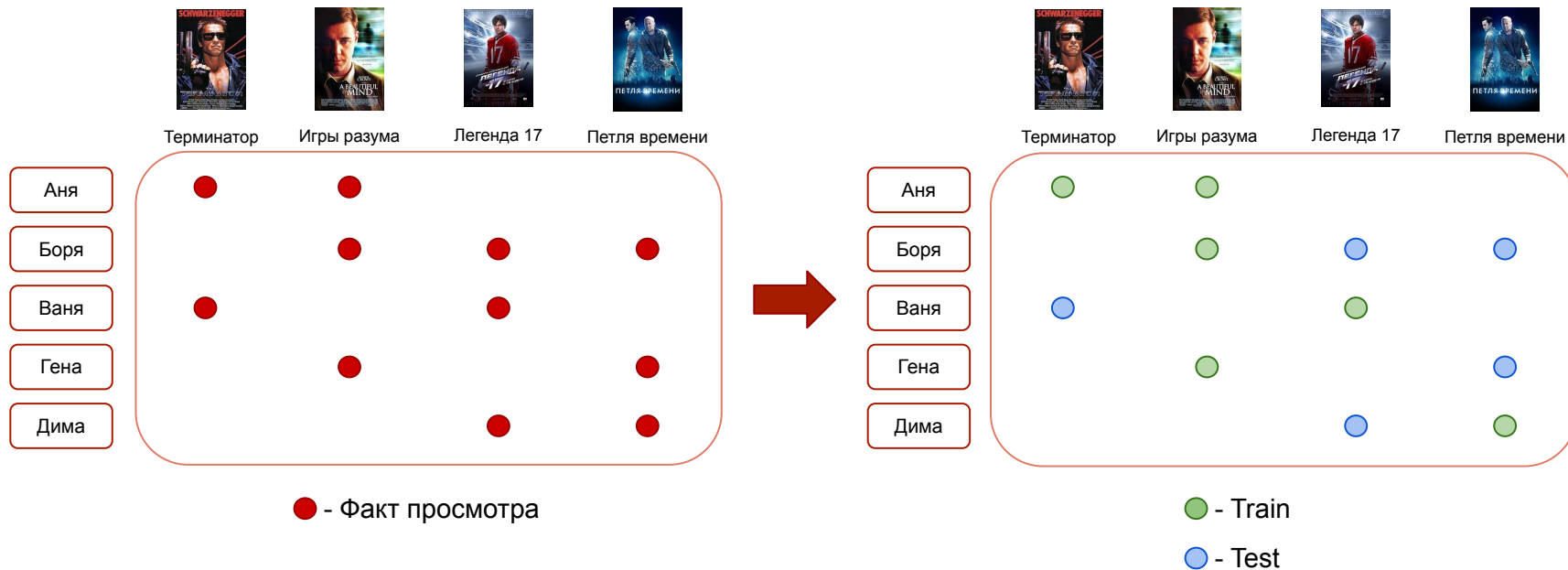
- **Random** - случайное разбиение. Подходит для каких-то *исследовательских задач* или *соревнований*, но в *реальных задачах этот способ редко отражает процесс* использования модели.
- **Temporal** - разбиение с учетом порядка, который чаще всего задается временем. Именно такой сценарий используется в *реальных задачах*.

В свою очередь в них можно выделить следующие схемы:

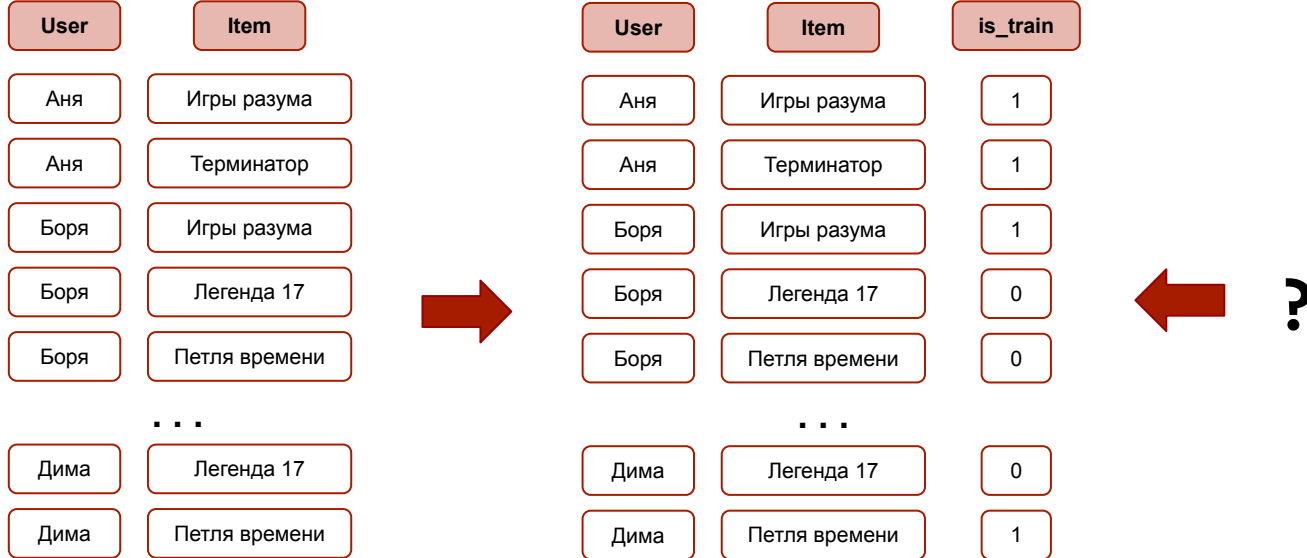
- **Threshold-based** - разбиение на фолды по отсечениям
- **Leave-P-out** - выделение в test/validation какой-то части взаимодействий по user и/или item, остальное в train
- **Leave-P-in** - выделение в train какой-то части взаимодействий по user и/или item, остальное в test/validation

Ну и куда без гибридных методов, которые объединяют все эти подходы в разных пропорциях :)

Оффлайн валидация. Random



Оффлайн валидация. Random



Оффлайн валидация. Random

User	Item
Аня	Игры разума
Аня	Терминатор
Боря	Игры разума
Боря	Легенда 17
Боря	Петля времени
...	
Дима	Легенда 17
Дима	Петля времени



User	Item	is_train
Аня	Игры разума	1
Аня	Терминатор	1
Боря	Игры разума	1
Боря	Легенда 17	0
Боря	Петля времени	0
...		
Дима	Легенда 17	0
Дима	Петля времени	1



`numpy.random.rand(len(df)) < 0.5`

- Как использовать scikit-learn?

Оффлайн валидация. Random

User	Item		User	Item	is_train
Аня	Игры разума		Аня	Игры разума	1
Аня	Терминатор		Аня	Терминатор	1
Боря	Игры разума		Боря	Игры разума	1
Боря	Легенда 17		Боря	Легенда 17	0
Боря	Петля времени		Боря	Петля времени	0
...			...		
Дима	Легенда 17		Дима	Легенда 17	0
Дима	Петля времени		Дима	Петля времени	1

← `numpy.random.rand(len(df)) < 0.5`

- Как использовать *scikit-learn*?

`train_test_split` подойдет для данного случая, но остальные классы уже не нет

- Как гарантировать хотя бы одно взаимодействие в *train* для каждого пользователя?

Random

Как гарантировать хотя бы одно взаимодействие в train для каждого пользователя?

Случайное разбиение - это взять в случайном порядке, значит мы можем взять как минимум один объект для каждого, просто взяв первый по нашему случайному порядку

	user	item
0	Аня	Терминатор
1	Аня	Игры разума
2	Боря	Игры разума
3	Боря	Легенда 17
4	Боря	Петля времени
5	Ваня	Терминатор
6	Ваня	Легенда 17
7	Гена	Игры разума
8	Гена	Петля времени
9	Дима	Легенда 17
10	Дима	Петля времени



	user	item	rank
0	Аня	Терминатор	2
1	Аня	Игры разума	1
2	Боря	Игры разума	1
3	Боря	Легенда 17	2
4	Боря	Петля времени	3
5	Ваня	Терминатор	2
6	Ваня	Легенда 17	1
7	Гена	Игры разума	1
8	Гена	Петля времени	2
9	Дима	Легенда 17	1
10	Дима	Петля времени	2



	user	item	rank	is_train
0	Аня	Терминатор	2	0
1	Аня	Игры разума	1	1
2	Боря	Игры разума	1	1
3	Боря	Легенда 17	2	0
4	Боря	Петля времени	3	0
5	Ваня	Терминатор	2	0
6	Ваня	Легенда 17	1	1
7	Гена	Игры разума	1	1
8	Гена	Петля времени	2	0
9	Дима	Легенда 17	1	1
10	Дима	Петля времени	2	0

```
df['rank'] = df.groupby('user').rank()
```

Добавляем временную колонку rank


```
df['is_train'] = df['rank'] < 2
```

Определяем тест относительно rank

- + “Векторизированные” операции над таблицей/датафреймом
- + Свобода действий по манипуляции с разными срезами (пользователи, объекты и тд)
- Потребление памяти

Time-based

Для данного разбиения нам нужно, чтобы в данных было время (date/timestamp/etc)



	user	item	timestamp
0	Аня	Терминатор	2022-08-02
1	Аня	Игры разума	2022-08-14
2	Боря	Игры разума	2022-08-03
3	Боря	Легенда 17	2022-08-09
4	Боря	Петля времени	2022-08-11
5	Ваня	Терминатор	2022-08-15
6	Ваня	Легенда 17	2022-08-23
7	Гена	Игры разума	2022-08-05
8	Гена	Петля времени	2022-08-04
9	Дима	Легенда 17	2022-08-26
10	Дима	Петля времени	2022-08-07

	user	item	timestamp	is_train
0	Аня	Терминатор	2022-08-02	1
1	Аня	Игры разума	2022-08-14	1
2	Боря	Игры разума	2022-08-03	1
3	Боря	Легенда 17	2022-08-09	1
4	Боря	Петля времени	2022-08-11	1
5	Ваня	Терминатор	2022-08-15	0
6	Ваня	Легенда 17	2022-08-23	0
7	Гена	Игры разума	2022-08-05	1
8	Гена	Петля времени	2022-08-04	1
9	Дима	Легенда 17	2022-08-26	0
10	Дима	Петля времени	2022-08-07	1

```
df['is_train'] = df['timestamp'] < pd.to_datetime('15-08-2022')
```

Функционал датафрейма естественным образом позволяет строить валидацию по дням/неделям или другим срезам, моделируя ситуации, когда мы предрасчитываем рекомендации на какой-то период.

Leave-P-out/in

Для данного разбиения нам нужно, чтобы в данных был задан порядок.

В общем случае Leave-P-out/in применяются ко всем данным

> LeaveOneOut - оставить одно наблюдение для теста, следовательно кол-во фолдов = кол-ву взаимодействий

Применительно к рекомендательным системы Leave-P-out/in могут применяться также на разных уровнях контекста (по пользователям, по объектам и тд)

> UsersLeavePOut - оставить в test как минимум P взаимодействий для каждого пользователя, UsersLeavePIn - оставить в train

UsersLeavePIn(p=2)

	user	item	rank
0	Аня	Терминатор	1
1	Аня	Игры разума	2
2	Боря	Игры разума	1
3	Боря	Легенда 17	2
4	Боря	Петля времени	3
5	Ваня	Терминатор	1
6	Ваня	Легенда 17	2
8	Гена	Петля времени	1
7	Гена	Игры разума	2
10	Дима	Петля времени	1
9	Дима	Легенда 17	2




	user	item	rank	is_train
0	Аня	Терминатор	1	1
1	Аня	Игры разума	2	1
2	Боря	Игры разума	1	1
3	Боря	Легенда 17	2	1
4	Боря	Петля времени	3	0
5	Ваня	Терминатор	1	1
6	Ваня	Легенда 17	2	1
8	Гена	Петля времени	1	1
7	Гена	Игры разума	2	1
10	Дима	Петля времени	1	1
9	Дима	Легенда 17	2	1

`df['is_train'] = df[rank] < 3`

Hot/Warm/Cold-start

При разбиении на train/test может получиться так, что в test будут пользователи, которых не было в train - **cold start**. Противоположная ситуация обозначается соответственно **hot start**. Эти же термины применимы и к объектам, и к любым другим контекстам.

	user	item	timestamp	is_train
0	Аня	Терминатор	2022-08-02	1
1	Аня	Игры разума	2022-08-14	1
2	Боря	Игры разума	2022-08-03	1
3	Боря	Легенда 17	2022-08-09	1
4	Боря	Петля времени	2022-08-11	1
5	Ваня	Терминатор	2022-08-15	0
6	Ваня	Легенда 17	2022-08-23	0
7	Гена	Игры разума	2022-08-05	1
8	Гена	Петля времени	2022-08-04	1
9	Дима	Легенда 17	2022-08-26	0
10	Дима	Петля времени	2022-08-07	1



	user	item	timestamp	is_train
0	Аня	Терминатор	2022-08-02	1
1	Аня	Игры разума	2022-08-14	1
2	Боря	Игры разума	2022-08-03	1
3	Боря	Легенда 17	2022-08-09	1
4	Боря	Петля времени	2022-08-11	1
5	Ваня	Терминатор	2022-08-15	0
6	Ваня	Легенда 17	2022-08-23	0
7	Гена	Игры разума	2022-08-05	1
8	Гена	Петля времени	2022-08-04	1
9	Дима	Легенда 17	2022-08-26	0
10	Дима	Петля времени	2022-08-07	1

Также при разбиении может оказаться так, что для пользователя нет взаимодействий в train, но зато есть “много” в test. В таком случае мы можем использовать эти новые взаимодействия, чтобы персонализировать выдачу для этого пользователя даже если его нет в train. Такая ситуация обозначается **warm start**.

Как подбирать схему

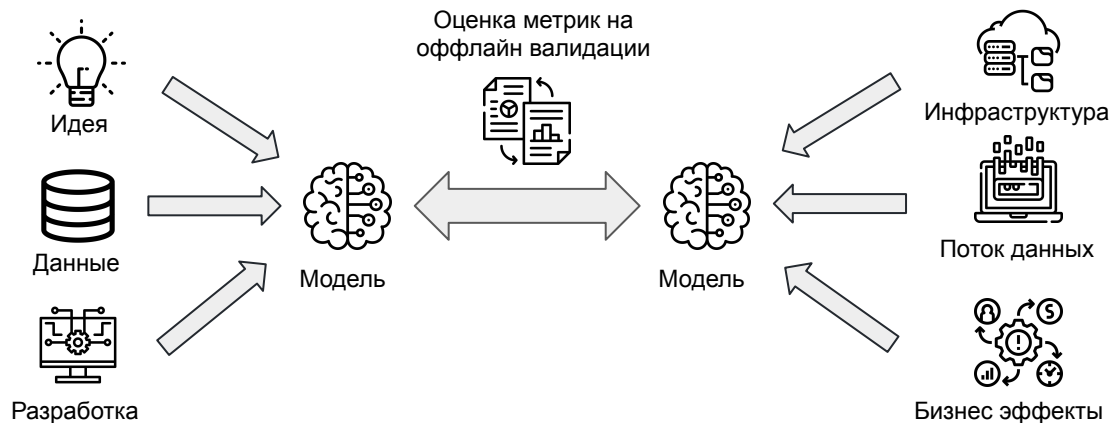
- Помним про основную цель - воспроизвести процесс использования модели
- В большинстве случаев в качестве базовой схемы используется разбиение по времени
 - Random/Leave-P-Out чаще встречаются в соревнованиях или исследованиях
- Определите, на каких сегментах пользователей и объектов надо тестировать модель
 - Hot/Warm/Cold сценарии
 - Другие разбиения (например по устройствам или по подборкам)
- При работе с большим объемом полезно заранее рассчитать фолды или сэмплировать

План

- Валидация
- **Метрики**
- Практические заметки

Что такое метрики

Метрика - функция оценки качества модели. Они позволяют оценить работу как одной модели, так и сравнить между собой несколько моделей. Выбор правильной метрики - один из ключевых этапов построения процесса валидации. При построении рекомендательных систем этот вопрос стоит еще острее, так как от системы мы хотим улучшение бизнесовых-метрик (конверсии, CTR, average session time), а при валидации можем считать только offline метрики (точность, качество ранжирования).



Оффлайн метрики

- **Регрессионные метрики (Error metrics)** применяются, когда наша модель учится предсказывать какое-то число, которое характеризует взаимодействие между пользователем и объектом, чаще всего это рейтинги
- **Классификационные метрики (Accuracy metrics)** оценивают качество топ-N рекомендаций с точки зрения бинарной классификации
- **Метрики ранжирования (Ranking metrics)** оценивают качество топ-N рекомендаций с учетом позиций (ранков)
- **Специальные метрики (Beyond-Accuracy metrics)** помогают оценить рекомендации с точки зрения пользовательского опыта
- На самом деле есть *много и других метрик*, например тех, которые оценивают стабильность рекомендаций или смещение рекомендаций относительно каких-то пользовательских групп

Регрессионные метрики

Регрессионные метрики (Error metrics) применяются, когда наша модель учится предсказывать какое-то число, которое характеризует взаимодействие между пользователем и объектом, чаще всего это рейтинги. На практике такое встречается не слишком часто, так как хорошее предсказание рейтингов не гарантирует хорошие ранжирование.

$$\frac{\sum_{i=1}^N |r_i - \hat{r}_i|}{N} \quad \frac{\sum_{i=1}^N (r_i - \hat{r}_i)^2}{N} \quad \sqrt{MSE}$$

N - кол-во взаимодействий в тесте

r_i - правильный рейтинг

\hat{r}_i - предсказанный рейтинг

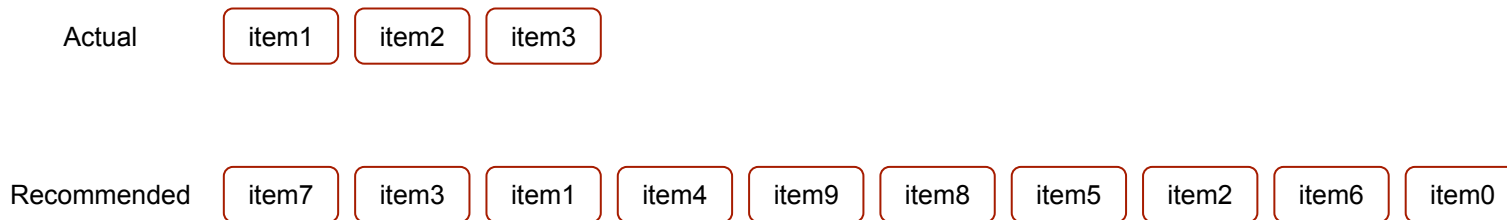
	user	item	timestamp	rating	rating_pred
0	Аня	Терминатор	2022-08-02	5	5.0
1	Аня	Игры разума	2022-08-14	3	3.7
2	Боря	Игры разума	2022-08-03	4	5.0
3	Боря	Легенда 17	2022-08-09	5	4.1
4	Боря	Петля времени	2022-08-11	4	3.2
5	Ваня	Терминатор	2022-08-15	5	4.9
6	Ваня	Легенда 17	2022-08-23	5	5.0
7	Гена	Игры разума	2022-08-05	5	3.8
8	Гена	Петля времени	2022-08-04	3	5.0
9	Дима	Легенда 17	2022-08-26	4	3.4
10	Дима	Петля времени	2022-08-07	5	4.5



- 1) По всем рейтингам
- 2) Усреднить по пользователям

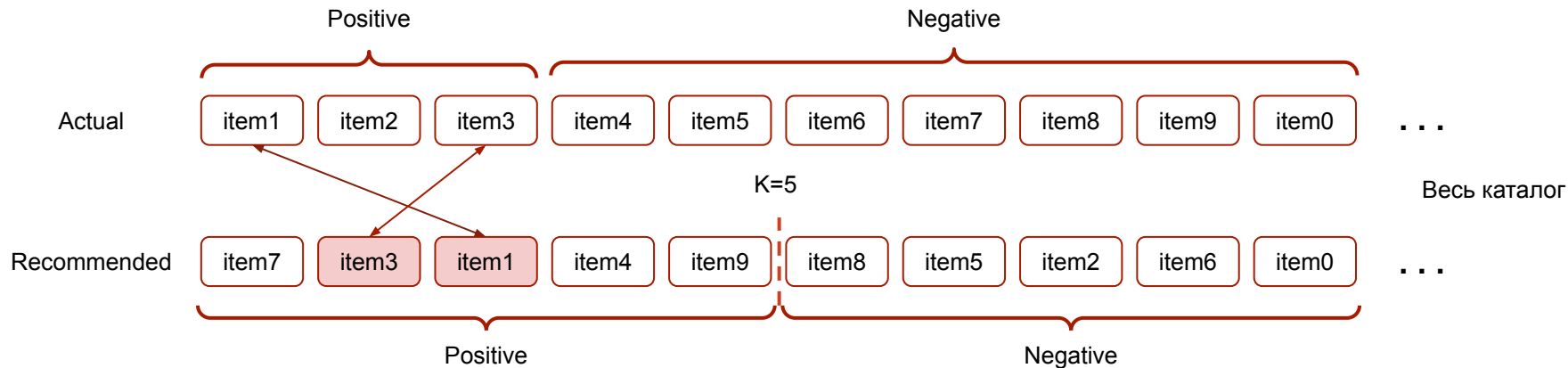
Классификационные метрики

Классификационные метрики (Accuracy metrics) оценивают качество топ-N рекомендаций с точки зрения бинарной классификации.



Классификационные метрики

Классификационные метрики (Accuracy metrics) оценивают качество топ-N рекомендаций с точки зрения бинарной классификации.



Классификационные метрики

Все считается на основе 4 базовых случаев:

- True positive (TP) - модель рекомендовала объект, с которым пользователь провзаимодействовал
- False positive (FP) - модель рекомендовала объект, с которым пользователь не провзаимодействовал
- True negative (TN) - модель не рекомендовала объект, с которым пользователь не провзаимодействовал
- False negative (FN) - модель не рекомендовала объект, с которым пользователь провзаимодействовал

		Recommended		
		Positive	Negative	
Actual	Positive	TP	FN	true_items
	Negative	FP	TN	N_items - true_items
		K	N_items - K	

Особенности:

- Для каждого K будет своя матрица
- Actual negative часто не бывает или бывает очень мало
- TN покрывают львиную долю каталога
- FP не обязательно говорит об ошибке и напрямую зависит от выбранного K

Классификационные метрики

$$Precision@K = \frac{TP@K}{TP@K + FP@K} = \frac{TP@K}{K}$$

- Можно заметить, что под positives мы понимаем рекомендованные объекты, то есть наш топ-K, значит $TP + FP = K$.
- Интерпретируется как доля релевантных рекомендаций

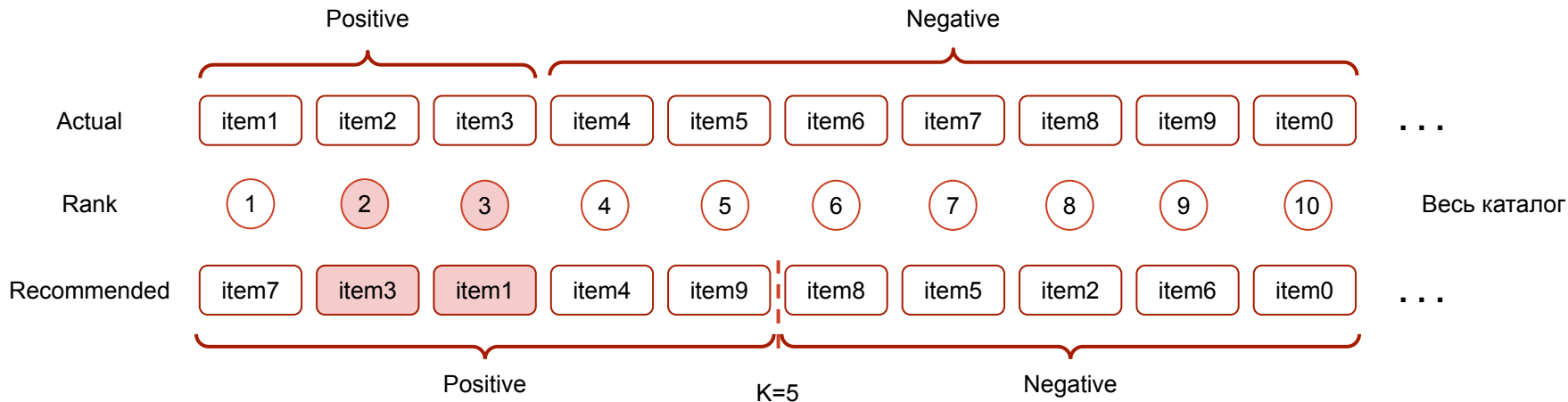
$$Recall@K = \frac{TP@K}{TP@K + FN@K}$$

- $TP + FN$ это количество известных релевантных объектов для пользователя.
- Интерпретируется как доля релевантных объектов, попавших в рекомендации

На основе TP, FP, FN, TN можно построить любую из метрик на основе confusion matrix

Метрики ранжирования

Метрики ранжирования (Ranking metrics) оценивают качество топ-N рекомендаций с учетом позиций (ранков).



Метрики ранжирования

Mean Reciprocal Rank - средний обратный ранг

$$MRR = \frac{1}{N_{users}} \sum_{u=1}^{N_{users}} \frac{1}{rank_u}$$

N_{users} - кол-во пользователей, $rank_u$ - позиция первой релевантной рекомендации для пользователя u

Причем если для пользователя мы не рекомендовали ничего релевантного, то дробь $\frac{1}{rank_u}$ зануляется

Метрики ранжирования

Mean Average Precision - средняя точность по пользователям

$$MAP@K = \frac{1}{N_{users}} \sum_{u=1}^{N_{users}} AP@K(user_u)$$

$$AP@K(user) = \frac{1}{N_{actual}(user)} \sum_{i=1}^K Precision@k * Rel(user, item_i)$$

N_{users} - кол-во пользователей

$N_{actual}(user)$ - кол-во релевантных объектов у пользователя $user$

$Rel(user, item_i)$ - релевантность i -ой рекомендации для пользователя $user$

Метрики ранжирования

Normalized Discounted Cumulative Gain - взвешенная точность по пользователю

$$CG@K(user) = \sum_{i=1}^K Rel(user, item_i) \quad Rel(user, item_i) - \text{вес } i\text{-ой позиции, 0 если не релевантна}$$

$$DCG@K(user) = \sum_{i=1}^K \frac{Rel(user, item_i)}{\log_2(i+1)} \quad N_{actual}(user) - \text{кол-во релевантных объектов у пользователя}$$

$$IDCG@K(user) = \sum_{i=1}^{\min(N_{actual}(user), K)} \frac{Rel(user, item_i)}{\log_2(i+1)}$$

$$nDCG@K(user) = \frac{DCG@K(user)}{IDCG@K(user)} \quad NDCG@K = \frac{1}{N_{users}} \sum_{u=1}^{N_{users}} nDCG@K(user_u)$$

Специальные метрики

Специальные метрики (Beyond-Accuracy metrics) помогают оценить рекомендации с точки зрения пользовательского опыта. Классы метрик:

- Coverage - покрытие. Может считаться как по объектам (сколько из них рекомендованы хотя бы раз), так и по пользователям (скольким можем построить персональные рекомендации)
- Novelty - новизна
- Diversity - разнообразие
- Serendipity - “полезная” неожиданность
- Bias - смещение
- Fairness - честность
- Stability - стабильность
- Вероятностные - PFound, Rank-biased Precision

Обзор - <https://dl.acm.org/doi/10.1145/2926720>

Coverage

Coverage - кол-во уникальных объектов в рекомендациях, то есть объектов, которые рекомендованы хотя бы одному пользователю.

```
recs[item_id].nunique()
```

Normalized coverage - кол-во уникальных объектов в рекомендациях, деленное на объем каталога

```
recs[item_id].nunique() / len(items_catalog)
```

Users coverage - кол-во пользователей, которым мы можем построить рекомендации. Имеет скорее аналитический смысл (Matrix factorization не умеет рекомендовать холодным пользователям)

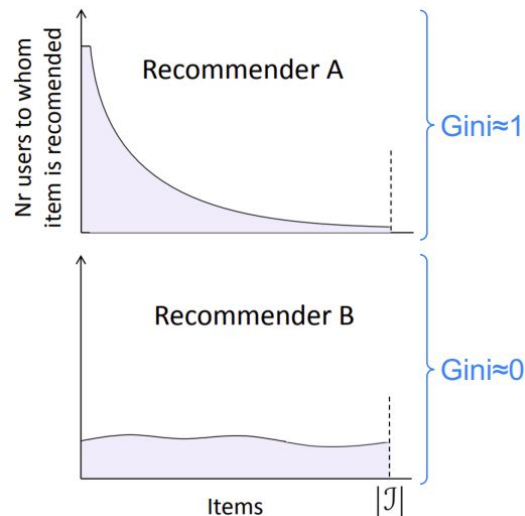
Коэффициент Gini

Коэффициент Gini - равномерность распределения присутствия товаров в рекомендациях

$$Gini = \frac{1}{1 - N_{items}} \sum_{k=1}^{N_{items}} (2k - N_{items} - 1)p(item_k)$$

$\{item_1, \dots, item_N\}$ - порядок сортировки объектов по вероятности вытягивания объекта из списка рекомендаций

$$p(item) = \frac{\text{len}(item \text{ in Recs})}{\text{len}(Recs)} - \text{вероятность}$$



Intra-List Diversity

Intra-List Diversity - определяется как среднее попарное расстояние между объектами

$$ILD@K(user) = \frac{1}{K(K-1)} \sum_{\substack{i=1 \\ i \neq j}}^K \sum_{j=1}^K dist(item_i, item_j)$$

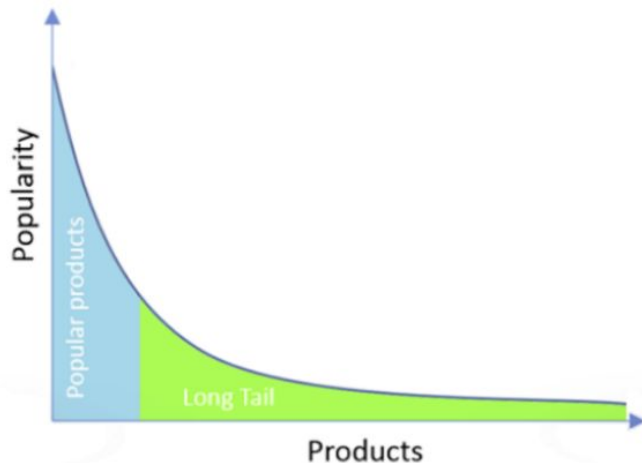
$dist(item_i, item_j)$ - определяется на основе векторных представлений объектов, будь то эмбединги или one-hot представления:

- Евклидово
- Косинусное
- Хэмминга
- Манхэттена

Novelty

Mean Inverse User Frequency - средняя новизна товаров в полке, где “новизна” товара обратно пропорциональна количеству пользователей, которые с ним взаимодействовали

$$MIUF@K(user) = -\frac{1}{K} \sum_{i=1}^K \log_2 \frac{N_{users}(item_i)}{N_{users}}$$



Ускорение подсчета оффлайн метрик

Есть три направления:

- Ускорение кода
- Подсчет на сэмпле
- Аппроксимация на основе более “легких” метрик

Ускорение подсчета оффлайн метрик

Есть три подхода:

- Векторизированные операции
 - Поиск “пересечений” между таргетом и рекомендациям за счет join операции над таблицами
- JIT компиляция
 - Обертка кода в numba или jax
- Распараллеливание
 - Подсчет метрик обладает свойством чрезвычайной параллельности, так как по всем пользователям их можно считать независимо

Векторизация

Основная идея - представить каждое слагаемое в метрик в качестве столбца

Актуальное

User	Item
Аня	Игры разума
Аня	Терминатор

`test.join(recs)`



User	Item	rank
Аня	Игры разума	1
Аня	Легенда 17	2
Аня	Петля времени	3
Аня	Терминатор	4
Аня	Довод	5

Рекомендуемое

Векторизация

Основная идея - представить каждое слагаемое в метрик в качестве столбца

Актуальное

User	Item
Аня	Игры разума
Аня	Терминатор

User	Item	rank
Аня	Игры разума	1
Аня	Легенда 17	2
Аня	Петля времени	3
Аня	Терминатор	4
Аня	Довод	5

Рекомендуемое

`test.join(recs)`



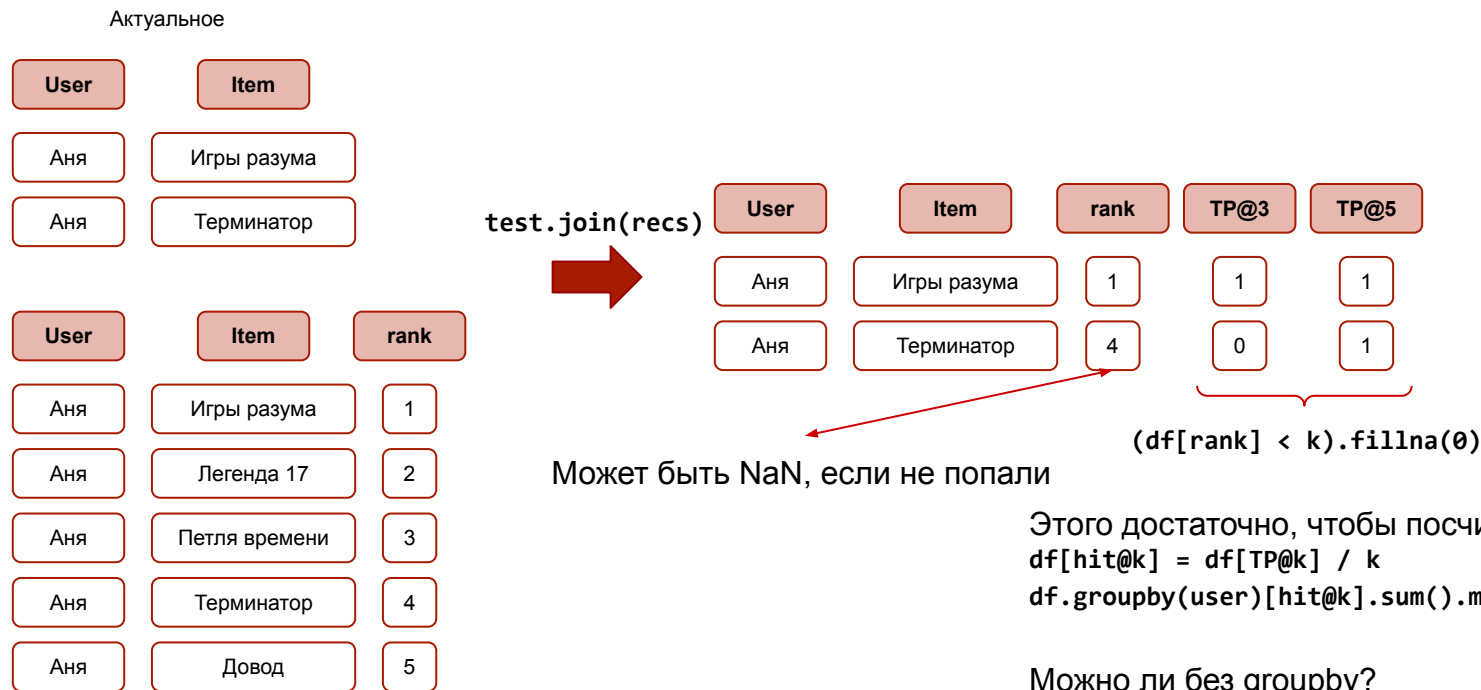
User	Item	rank
Аня	Игры разума	1
Аня	Терминатор	4



Может быть NaN, если не попали

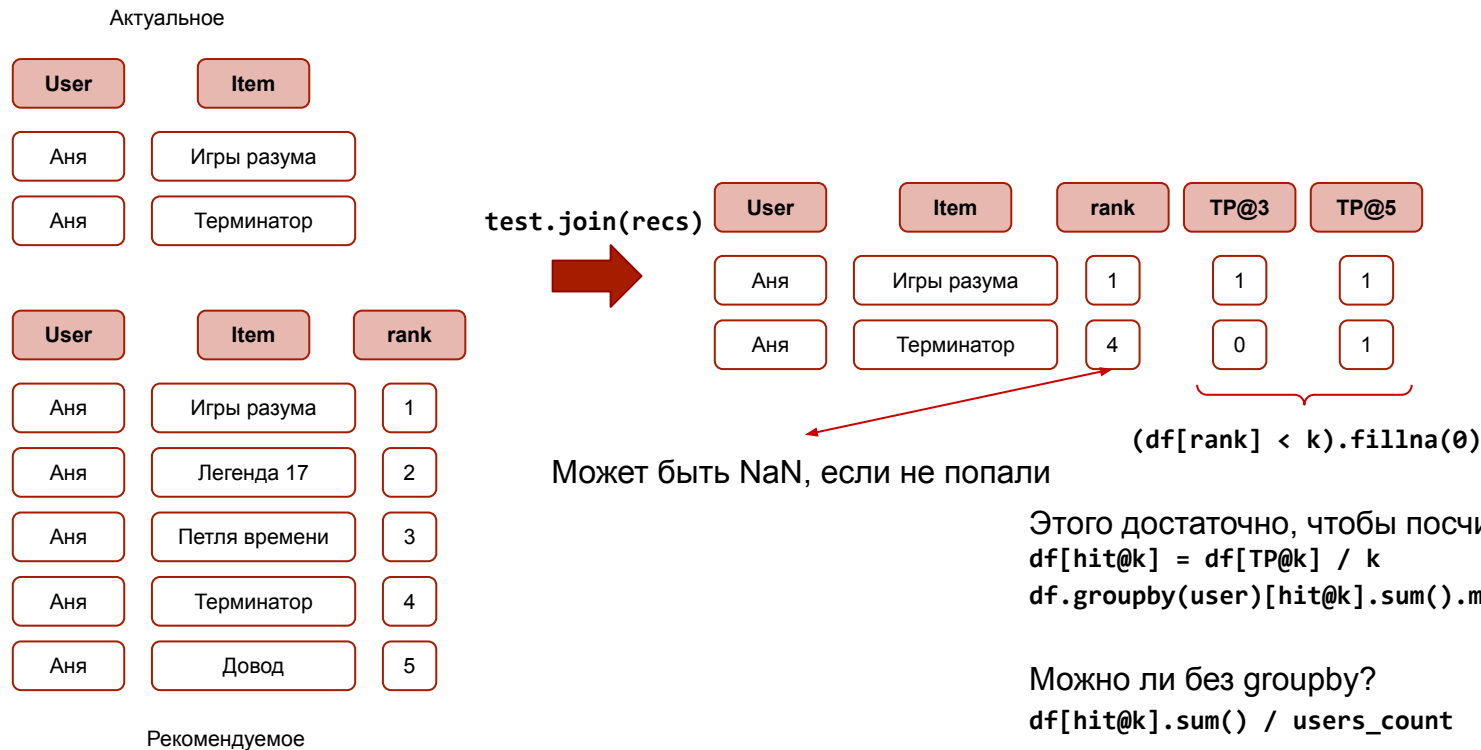
Векторизация

Основная идея - представить каждое слагаемое в метрик в качестве столбца



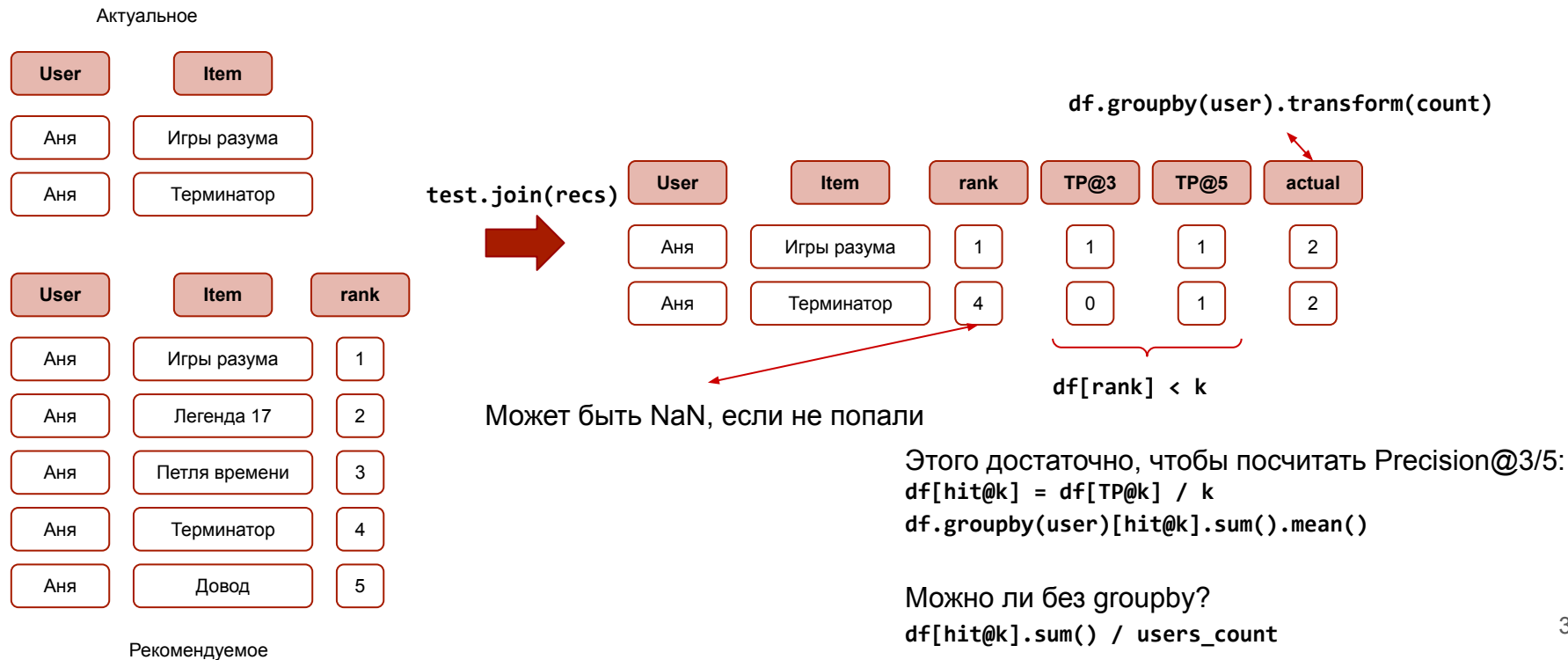
Векторизация

Основная идея - представить каждое слагаемое в метрик в качестве столбца



Векторизация

Основная идея - представить каждое слагаемое в метрик в качестве столбца



Векторизация

Разберем Mean Average Precision для k=5

$$MAP@K = \frac{1}{N_{users}} \sum_{u=1}^{N_{users}} AP@K(user_u)$$

$$AP@K(user) = \frac{1}{N_{actual}(user)} \sum_{i=1}^K Precision@k * Rel(user, item_i)$$

User	Item	rank	TP@3	TP@5	actual
Аня	Игры разума	1	1	1	2
Аня	Терминатор	4	0	1	2

Векторизация

Разберем Mean Average Precision для k=5

$$MAP@K = \frac{1}{N_{users}} \sum_{u=1}^{N_{users}} AP@K(user_u)$$

$$AP@K(user) = \frac{1}{N_{actual}(user)} \sum_{i=1}^K Precision@k * Rel(user, item_i)$$

User	Item	rank	TP@3	TP@5	actual	cumTP
Аня	Игры разума	1	1	1	2	1
Аня	Терминатор	4	0	1	2	2

1

Когда нужно работать с “предыдущими” значениями k, то спасает cumsum/cumprod

```
df[cumTP] = df.groupby(user)[TP@k].cumsum()
```

Векторизация

Разберем Mean Average Precision для k=5

$$MAP@K = \frac{1}{N_{users}} \sum_{u=1}^{N_{users}} AP@K(user_u)$$

$$AP@K(user) = \frac{1}{N_{actual}(user)} \sum_{i=1}^K Precision@k * Rel(user, item_i)$$

User	Item	rank	TP@3	TP@5	actual	cumTP	P@k
Аня	Игры разума	1	1	1	2	1	1
Аня	Терминатор	4	0	1	2	2	1/2

1

Когда нужно работать с “предыдущими” значениями k, то спасает cumsum/cumprod

```
df[cumTP] = df.groupby(user)[TP@k].cumsum()
```

2

Теперь мы можем получить каждое слагаемое

```
df[P@k] = df[cumTP] / df[rank]
```

Векторизация

Разберем Mean Average Precision для k=5

$$MAP@K = \frac{1}{N_{users}} \sum_{u=1}^{N_{users}} AP@K(user_u)$$

$$AP@K(user) = \frac{1}{N_{actual}(user)} \sum_{i=1}^K Precision@k * Rel(user, item_i)$$

User	Item	rank	TP@3	TP@5	actual	cumTP	P@k	P@k/N
Аня	Игры разума	1	1	1	2	1	1	1/2
Аня	Терминатор	4	0	1	2	2	1/2	1/4

1

Когда нужно работать с “предыдущими” значениями k, то спасает cumsum/cumprod

`df[cumTP] = df.groupby(user)[TP@k].cumsum()`

2

Теперь мы можем получить каждое слагаемое

`df[P@k] = df[cumTP] / df[rank]`

3

Вносим $\frac{1}{N_{actual}(user)}$ в сумму в каждое слагаемое

`df[P@k/N] = df[P@k] / df[actual]`

Векторизация

Разберем Mean Average Precision для k=5

$$MAP@K = \frac{1}{N_{users}} \sum_{u=1}^{N_{users}} AP@K(user_u)$$

$$AP@K(user) = \frac{1}{N_{actual}(user)} \sum_{i=1}^K Precision@k * Rel(user, item_i)$$

User	Item	rank	TP@3	TP@5	actual	cumTP	P@k	P@k/N
Аня	Игры разума	1	1	1	2	1	1	1/2
Аня	Терминатор	4	0	1	2	2	1/2	1/4

1

Когда нужно работать с “предыдущими” значениями k, то спасает cumsum/cumprod

`df[cumTP] = df.groupby(user)[TP@k].cumsum()`

2

Теперь мы можем получить каждое слагаемое

`df[P@k] = df[cumTP] / df[rank]`

3

Вносим $\frac{1}{N_{actual}(user)}$ в сумму в каждое слагаемое

`df[P@k/N] = df[P@k] / df[actual]`

4

Чтобы получить AP по user'ам

`df.groupby(user)[P@k/N].sum()`

Подсчет оффлайн метрик на сэмпле

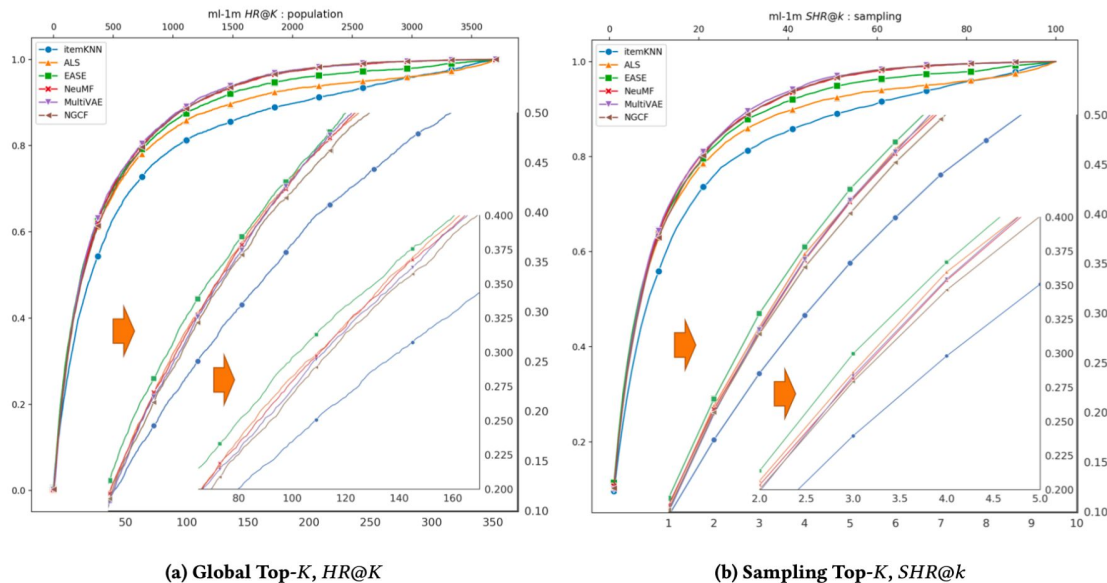
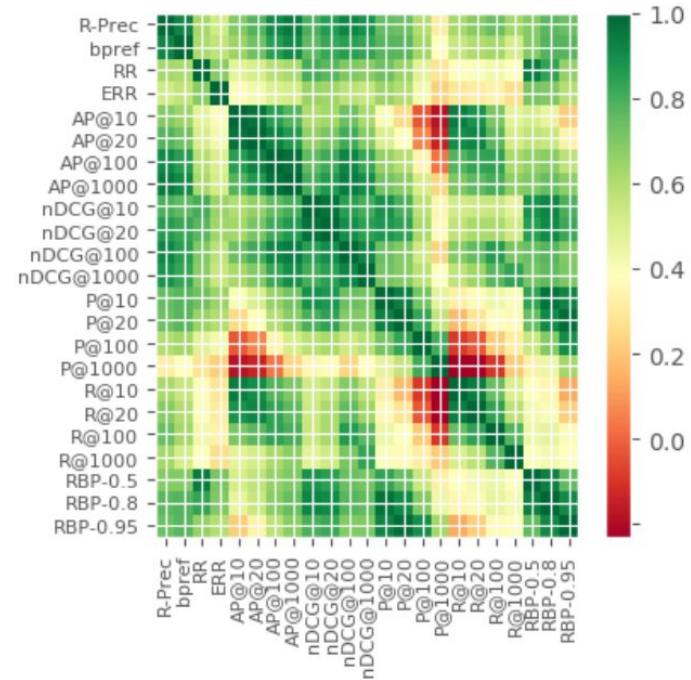


Figure 1: Global vs Sampling Top k Hit-Ratio on MovieLens 1M dataset (ml-1m). To display the details clearly, we zoom in at different range scales. Compare two figures, we can easily conclude that sampling evaluation maintains the same trend as global evaluation for different algorithms even at small error range.

[arxiv: On Sampling Top-K Recommendation Evaluation.pdf](#)

Аппроксимация оффлайн метрик

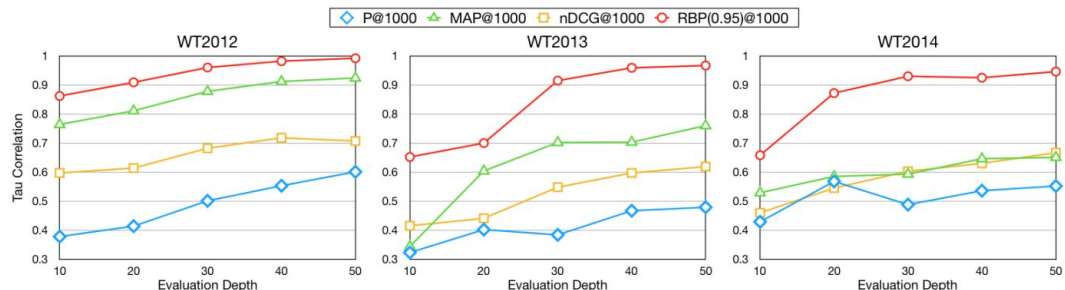
Test Set	Document Set	#Sys	Topics
WT2000 [22]	WT10g	105	451-500
WT2001 [49]	WT10g	97	501-550
RT2004 [48]	TREC 4&5*	110	301-450, 601-700
WT2010 [14]	ClueWeb'09	55	51-99
WT2011 [13]	ClueWeb'09	62	101-150
WT2012 [15]	ClueWeb'09	48	151-200
WT2013 [16]	ClueWeb'12	59	201-250
WT2014 [17]	ClueWeb'12	30	251-300



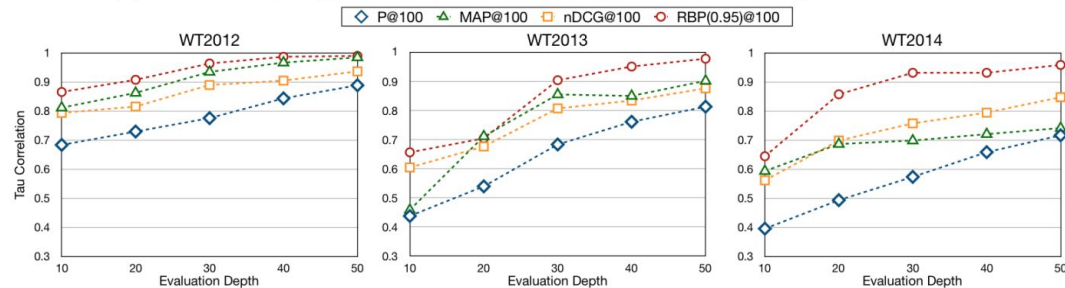
Аппроксимация оффлайн метрик

Table 1: System-wise Prediction of a metric using varying number of metrics $K = [1 - 3]$. Kendall's τ scores higher than 0.9 are bolded.

Predicted Metric	Independent Variables			WT2012		WT2013		WT2014	
				τ	R^2	τ	R^2	τ	R^2
bpref	nDCG	-	-	0.805	-0.693	0.885	0.079	0.915	-1.174
	nDCG	R-Prec	-	0.872	-0.202	0.850	0.094	0.824	-0.989
	nDCG	R-Prec	R@100	0.906	0.284	0.844	0.645	0.866	0.390
ERR	RR	-	-	0.764	-1.874	0.734	0.293	0.704	-1.004
	RR	RBP(0.8)	-	0.790	-1.809	0.777	0.392	0.714	-0.686
	RR	RBP(0.8)	R@100	0.796	-1.728	0.741	0.478	0.704	-0.473
GMAP	bpref	-	-	0.729	-1.216	0.704	-2.982	0.739	-1.034
	nDCG	RBP(0.5)	-	0.817	0.877	0.777	0.600	0.767	0.818
	nDCG	RBP(0.95)	RR	0.817	0.882	0.748	0.514	0.794	0.854
MAP	R-Prec	-	-	0.885	0.754	0.824	0.667	0.952	0.819
	R-Prec	nDCG	-	0.904	0.894	0.905	0.760	0.958	0.897
	R-Prec	nDCG	RR	0.924	0.916	0.901	0.779	0.947	0.922
nDCG	bpref	-	-	0.805	-2.101	0.885	-0.217	0.915	-2.008
	bpref	GMAP	-	0.803	-0.079	0.809	0.574	0.872	0.024
	bpref	GMAP	RBP(0.95)	0.794	-0.113	0.801	0.556	0.850	-0.032
P@10	RBP(0.8)	-	-	0.884	0.942	0.832	0.895	0.866	0.893
	RBP(0.8)	RBP(0.5)	-	0.941	0.994	0.882	0.966	0.914	0.988
	RBP(0.8)	RBP(0.5)	RR	0.946	0.994	0.885	0.968	0.914	0.987
RBP(0.95)	R-Prec	-	-	0.824	0.346	0.651	-0.786	0.607	-2.401
	bpref	P@10	-	0.911	0.952	0.718	0.873	0.728	0.591
	bpref	P@10	RBP(0.8)	0.911	0.967	0.720	0.868	0.744	0.639
R-Prec	R@100	-	-	0.899	0.708	0.871	0.624	0.935	0.019
	R@100	RBP(0.95)	-	0.909	0.952	0.820	0.882	0.820	0.759
	R@100	RBP(0.95)	GMAP	0.924	0.970	0.833	0.914	0.841	0.825
RR	RBP(0.5)	-	-	0.782	0.904	0.806	0.927	0.810	0.878
	RBP(0.5)	RBP(0.8)	-	0.869	0.918	0.809	0.919	0.820	0.942
	RBP(0.5)	RBP(0.8)	ERR	0.876	0.437	0.818	0.924	0.915	0.824
R@100	R-Prec	-	-	0.899	0.423	0.871	0.232	0.935	-1.075
	R-Prec	GMAP	-	0.899	0.433	0.871	0.238	0.940	-1.077
	R-Prec	RR	ERR	0.881	-0.104	0.823	0.355	0.935	-1.187



(a) Predicting High-Cost Measures using Evaluation Depth $D = 1000$



(b) Predicting High-Cost Measures using Evaluation Depth $D = 100$

Fig. 2: Linear regression prediction of high-cost metrics using low-cost metrics

Аппроксимация оффлайн метрик

Основные моменты из статьи:

- MAP, RBP и Precision на малых K можно предсказывать с хорошей точностью на основе 2-3 других метрик
- Некоторые метрики (например $RBP(0.95)$) на $K=1000$ могут предсказаны на метриках, посчитанных для $K=30$
- Учитывая вышесказанное, можно реализовать алгоритм отбора наиболее информативных метрик, вместо подсчета всего на всех K
- Наиболее информативными оказались $MAP@1000$, $P@1000$, $nDCG@1000$ и $RBP(p=0.95)$

Фреймворки

Векторизованные подходы:

- <https://github.com/MobileTeleSystems/RecTools>
- <https://github.com/microsoft/recommenders>
- https://github.com/zuoxingdong/recsys_metrics
- <https://github.com/RUCAIBox/RecBole>

Циклы:

- <https://github.com/AmenRa/ranx> (numba)
- <https://github.com/AstraZeneca/rexmex> (numpy)
- <https://github.com/statisticianinstilettos/recmetrics>
- <https://github.com/sisinflab/elliott>

Большинство фреймворков с моделями в себе содержат крайне не оптимальные методы подсчета метрик

План

- Валидация
- Метрики
- Практические заметки

Валидация на практике

Анализ рекомендаций делится на две части:

- Количественный - показатели модели выраженные в бизнес-эффектах (CTR, конверсии, время сессий и так далее)
 - Проверяется на “в бою” на АБ тестах
 - Ранжируются от short-term (клики, просмотр) до long-term (LTV, конверсия в подписку)
- Качественный - как пользователи воспримут/восприняли новые рекомендации
 - Субъективная оценка
 - Как получить?
 - Посмотреть самому на рекомендации
 - Спросить пользователей
 - User studies/experiments

Offline vs Online

Как по оффлайн метрикам понять, как будет себя вести модель в проде и какие бизнес-эффекты можно ожидать?

Offline vs Online

Как по оффлайн метрикам понять, как будет себя вести модель в проде и какие бизнес-эффекты можно ожидать?

Вкратце - никак :)

Offline vs Online

Как по оффлайн метрикам понять, как будет себя вести модель в проде и какие бизнес-эффекты можно ожидать?

Вкратце - никак :)

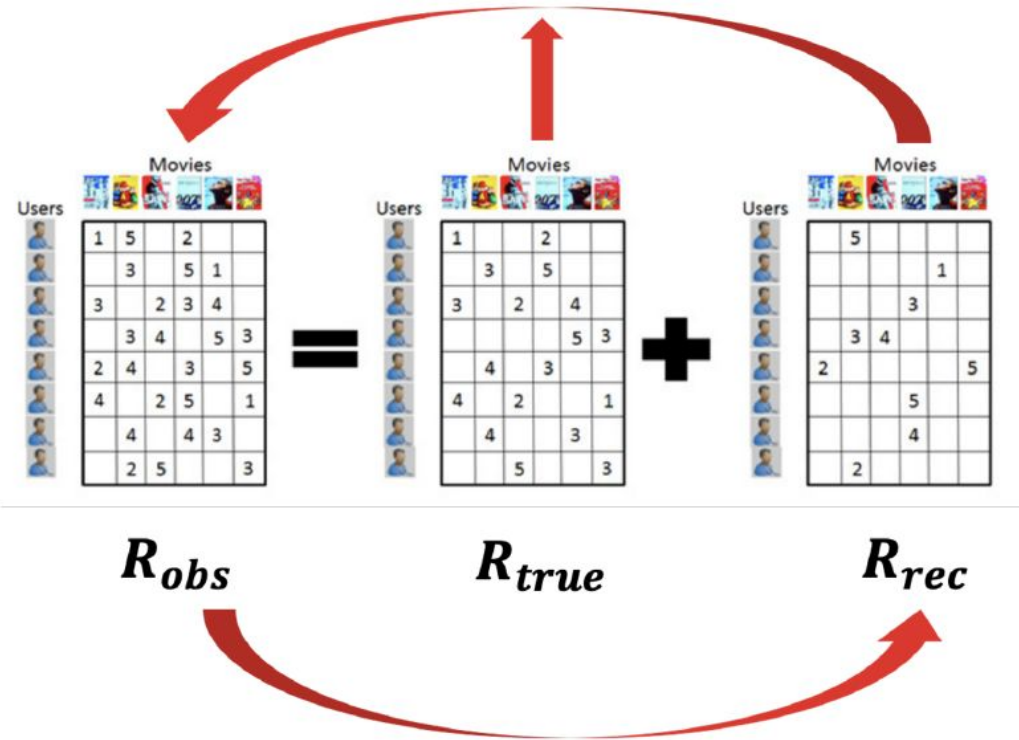
Но тем не менее возможные варианты:

- Проверка боем: если у новой модели не просели оффлайн метрики, а по некоторым даже есть улучшение - то катим в прод и смотрим, что выйдет
- Прогнозирование метрик на АБ по оффлайн метрикам
 - Для этого нужно всестороннее логирование АБ и набранная история по экспериментам

Feedback loop



Feedback Loop



Feedback loop

Как решать:

- Случайно подмешивать рекомендации из популярного/эвристики
- Тестировать бандитов
- Глобальная контрольная группа без экспериментов
- Эвристики

Более сложные решения:

- Попытка выделить R_{true} из данных
- Встраивание Reinforcement learning

Тестирование

- Проверяйте глазами, что выдает ваша модель
 - Полезно иметь аватаров, на которых можно глазами оценить адекватность рекомендаций
- Оценивайте скорость работы модели и потребление памяти
 - В простых случаях помогут *line profiler* и *memory use profiler*
 - Для серьезных бенчмарков лучше использовать Docker с ограничениям по ресурсам
- Проверяйте модели на дубликаты или другие соответствия базовым фильтрам
 - Подавляющее большинство метрик неустойчивы к дубликатам, для этого нужны дополнительные проверки
- Просадка по оффлайн метрикам не так страшна, смотрите на ситуацию шире
 - Потому что offline не шибко коррелирует с online чаще всего, но все зависит от кейса

Тестирование

Полезно иметь инструмент, который позволяет смотреть на рекомендации во время экспериментирования с моделями.

Вариантов два:

- пакет, который сможет выводить результаты в Jupyter
- отдельная “админка” для мониторинга всех моделей


```
from showcase import Showcase
final_show = Showcase.load_data("boosting_july_22", reco_cols=['score'])
```

Select user:

rock_guy	concert_electro	opera_and_drama	opera_classic_theatre	exhibition
classic_bomb	theatre_and_jazz	strange_circus_to...	strange_to_one_the...	musicie_&_kids
classic_&_kids	pop_music_and_je...	bests_agutin	vahtangoff	kids_awesome
2_spectacle	sovremennik_and_...	kids_and_shows	drama	many_drama
drama_and_arbenina	unreliv_fashion	unreliv_rock	unreliv_music	with_excursion
random_0	random_1	random_2	random_3	random_4

User_id 7710088

viewed

item_id	img	premiere_date	meta_title	categories	genres	link
3833597		2019-09-17	Спектакль «Вишневый сад» в театре имени Пушкина	Спектакли	Комедии	Link




Навигация по рекомендательным алгоритмам

Select model:

pop_14	tfidf_100	bm25_k=100_k1=0...	slim_fs256_l1r0.1...	catb_rank_top
catb_clas_top	catb_clas_base	boost_bad		

Model name: catb_rank_top

reco

item_id	img	premiere_date	meta_title	categories	genres	link	score
3199455		2018-05-30	Спектакль «Три сестры» в МХТ имени Чехова	Спектакли	Драмы	Link	0.455344
3936210		2004-08-06	Спектакль «Чайка» в МХТ Чехова	Спектакли	Комедии	Link	0.424485
3973104		2020-09-02	Спектакль «Ва-банк» в театре «Ленком»	Спектакли	Комедии	Link	0.320197

Что надо вынести из лекции

- Валидация - очень важный процесс, который надо тщательно продумывать
- Оффлайн метрик много, надо подбирать под задачу
- Они не гарантируют эффект, но помогают избежать очевидных ошибок
- Тестировать надо не только метрики на тесте, но и саму модель
- Проверку глазами ничто не заменит
- АБ тестирование - основной инструмент оценки полезности рекомендательных систем

Домашнее задание

- Реализовать схему валидации UsersKFoldLeavePOut - 3 балла
 - Данные из Yandex Cup 2022 RecSys
- Реализовать векторизованную версию метрики PFound - 3 балла
 - Векторизованная = без цикла for
- Реализовать функцию подсчета метрики Mean Reciprocal Rank в разных видах и сравнить производительность - 4 балла
 - Три типа: naive, numba и pandas

Подробное описание задач будет в конце семинара