Рекомендательные системы

ШАД

Весна 2025

Лекция 7:

Рантаймы & Case Study

Футболки за топ-10 контеста

напишите мне в личку ваш размер, пожалуйста

План

- сегодня посмотрим на примеры рексистем
- рексистемы будут поменьше и побольше
- поговорим, как они устроены, чем отличаются
- а чем не отличаются

Дисклеймер

- мы не знаем точного устройства всех приводимых рантаймов кроме лавочного (и немного дзеновского)
- вся лекция является художественным вымыслом, поэтому
- все имена функций и моделей в лекции вымышлены, любые совпадения с реальными рантаймами, живыми или мертвыми, случайны

Общее

A complex system that works is invariably found to have evolved from a simple system that worked. A complex system designed from scratch never works and cannot be patched up to make it work. You have to start over with a working simple system.

– Gall's law

Что главное в рексистеме?

Самый полезный слайд во всем курсе

(предыдущие слайды врали)

- базово:
  - логи должны быть не совсем фиговыми
  - логи надо хранить всегда, даже если дорого
- критично важно:
  - единый формат с инкапсуляцией логики парсинга
  - использовать результат в пулах, мониторах, метриках AB и так далее
- все это с некоторой солью: есть размен на дороговизну поддержки
- чем раньше проверите, что у вас нормальные логи, там раньше получите первый профит

Компактная рексистема: Лавка

- Начинаем с Лавки — пример минималистичной рексистемы
- Основная сущность — экшены: нормализованные логи действий пользователей
- Может по-другому называться, но у всех обычно есть что-то такое

Экшены — формат, в который преобразуются клиентские логи. Он удобен для использования в ML-пайплайне: обучение, фичи, A/B тесты, мониторы и т.д .

Профили и источники данных

- Из экшенов считаются **профили пользователей и айтемов**
  - **Пользовательские** : активность, частотности
  - **Айтемные** : CTR, квоты, популярность, статистики по позициям

- **Дополнительные источники** :
  - экшены
  - таблички со статическими фичами (регион, цена и т.д.)
  - внешние данные (анкеты, партнёрские данные)
  - предикты моделей, эмбеды и т.п.

- Профиль = агрегированная информация о сущности. Не обязательно это вектор: могут быть и категориальные признаки, и счётчики.

Как применяются профили документов

Строится индекс

- Данные перекладываются в эффективный для доступа из памяти вид
- Строятся разнообразные ANN-индексы и статические топы
- Все это собирается в кучу и сериализуется
- И едет на тачку

Иногда таких индексов нужно много. Индексы еще будем называть шардами. От английского shard -- осколок

Вспоминаем про шарды

Если вы YouTube, вам придется

1) Делать selection rank
2) Шардироваться
   - Ваши кандидаты скорее всего лучше работают на объединении множеств меньшего размера
   - Ранжирование можно параллелить и заливать таким образом баблом проблему кандидатов

На практике обычно в один шард пихают что-то порядка миллиона айтемов

Случай Лавки

- **Всего ~40–50k айтемов во всей Лавке**
- **Это где-то 4-5 секунд на поранжировать**
- **В одной лавке (один dark store) — 3–10k одновременно доступных айтемов**
- **Candidate Generation (CG) **можно не делать****

Если пофильтроваться в правильный момент

- **0–100 мс** – отклик ощущается как мгновенный.
- **100–300 мс** – небольшая, но заметная задержка.
- **300–1000 мс** – пользователь чувствует, что система "думает".
- **>1 сек** – внимание пользователя начинает рассеиваться.
- **>3 сек – 53% пользователей покидают сайт** (по данным Google).

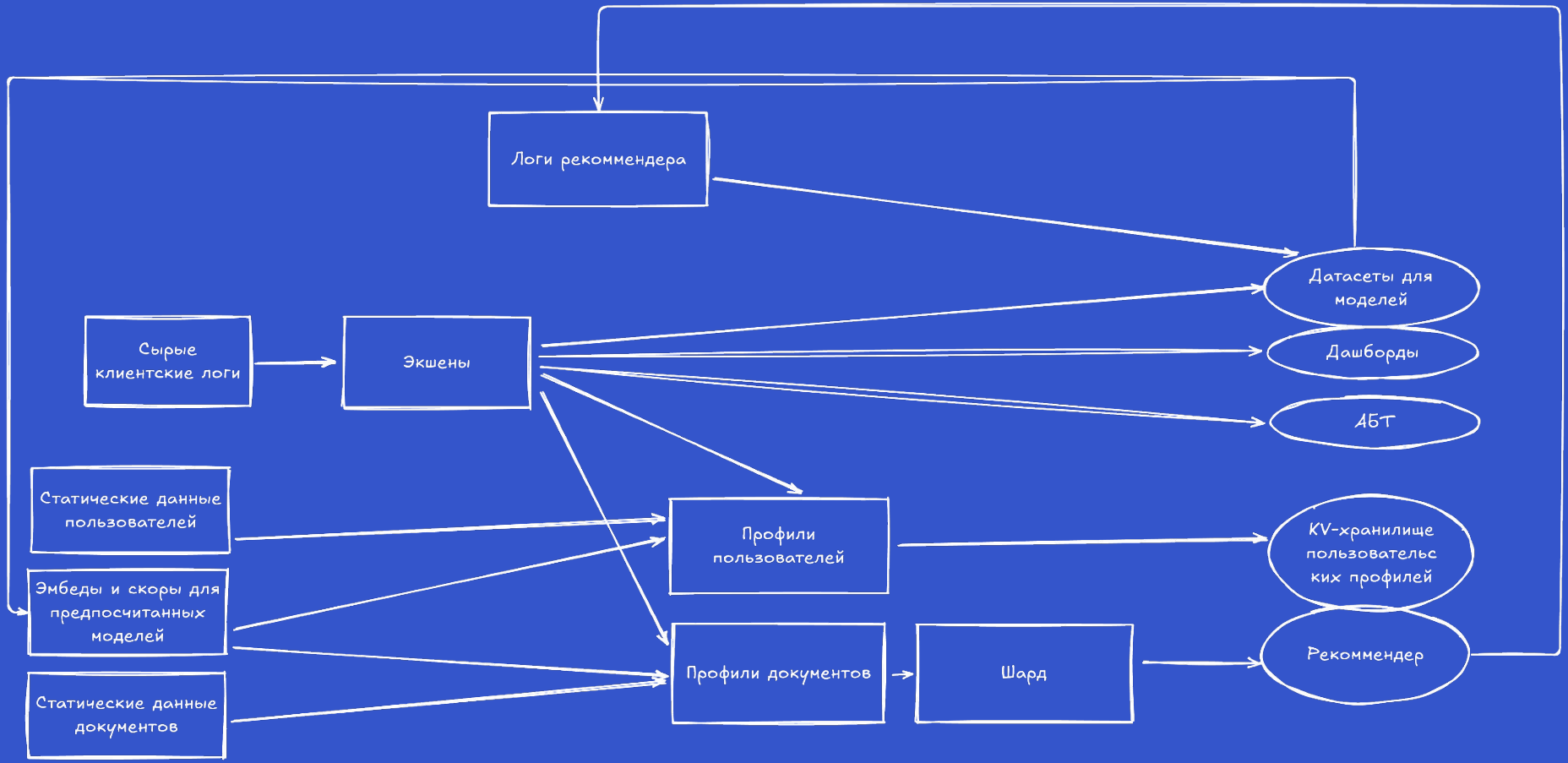
Как применяются профили пользователей

В случае Лавки

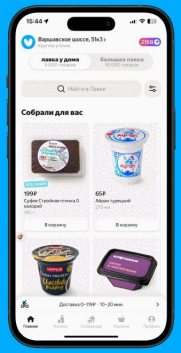
- Всё обсчитывается **оффлайн**, раз в сутки
- Загружается в **KV-хранилище** по user_id
- В онлайне → быстрое обращение (5–10 мс)
- Масштабируется изи: шардирование и отказоустойчивость встроены в большинство KV-хранилищ

Если пользователей немного — можно было бы засовывать профили прямо в шард. Но дальше это плохо масштабируется. Поэтому лучше использовать KV или что-то похожее даже для небольших систем.

Нюансы работы рексистемы Лавки


Логи рекоммендера
Датасеты для моделей
Сырые клиентские логи
Экшены
Дашборды
АБТ
Статические данные пользователей
Профили пользователей
KV-хранилище пользовательских профилей
Эмбеды и скоры для предпосчитанных моделей
Профили документов
Шард
Рекоммендер
Статические данные документов

Много поверхностей

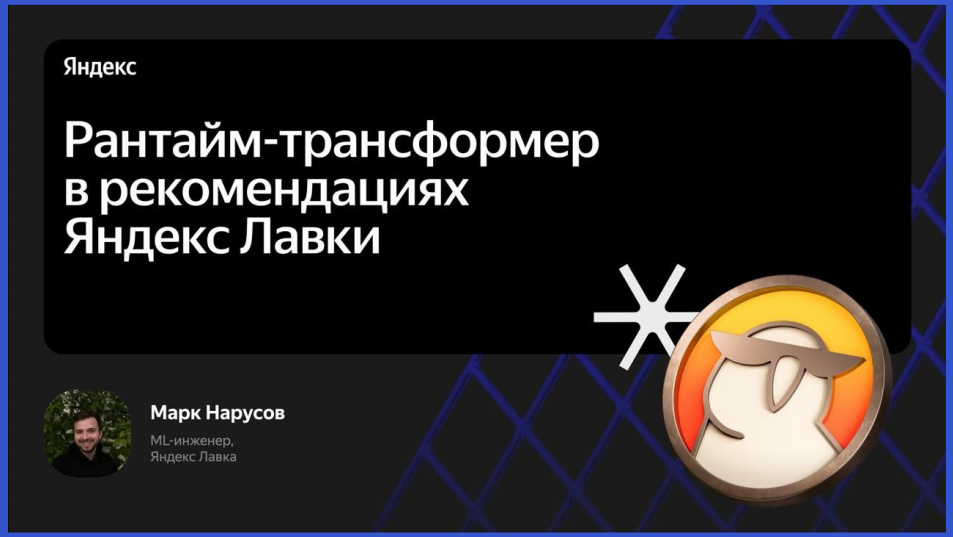


Рекомендации
на главной

- Главная, категории, поиск, корзина, оплата
- Поверхности каннибализируют друг друга
- Особенно достается тем, что в конце пользовательского пути от тех, кто в начале

Информация о сессии облегчает задачу

- Начали строить рантайм-профиль пользователя
- Применяется трансформер на действия пользователя в текущей сессии
- В будущем позволит нам порешать проблему с каннибализацией, еще боремся



Приходите послушать доклад Марка на датафесте, где он расскажет, как мы везли инференс трансформера в рантайм

Холодный старт: почти не проблема

- Все пользователи **залогинены**
- Использование без логина невозможно (надо платить)
- Новички — в меньшинстве
- Контент структурированный, легко курируется

Не банеры на порно-сайте крутим, больше думаем про персонализацию, чем про популярность

Кинопоиск — побольше, посложнее

- ~200–300k фильмов
- Влезает в один шард
- Но уже нужна кандидатогенерация
- Простые кандгены:
  - популярность, досматриваемость
  - новинки
  - редакторские подборки
  - похожесть по жанру, режиссёру и т.д.

Модели

- все, что вы слушали на лекциях, на широкую ногу
- в кандидатах двухбашенки
- простые кандгены с прошлого слайда
- в ранжировании скорее всего катбуст или какой-нибудь dcn-v2

Таргеты

В Лавке всё прозрачно:

- **P(покупка) × цена → GMV**
- **P(покупка) × маржа → прибыль**

В Кинопоиске:

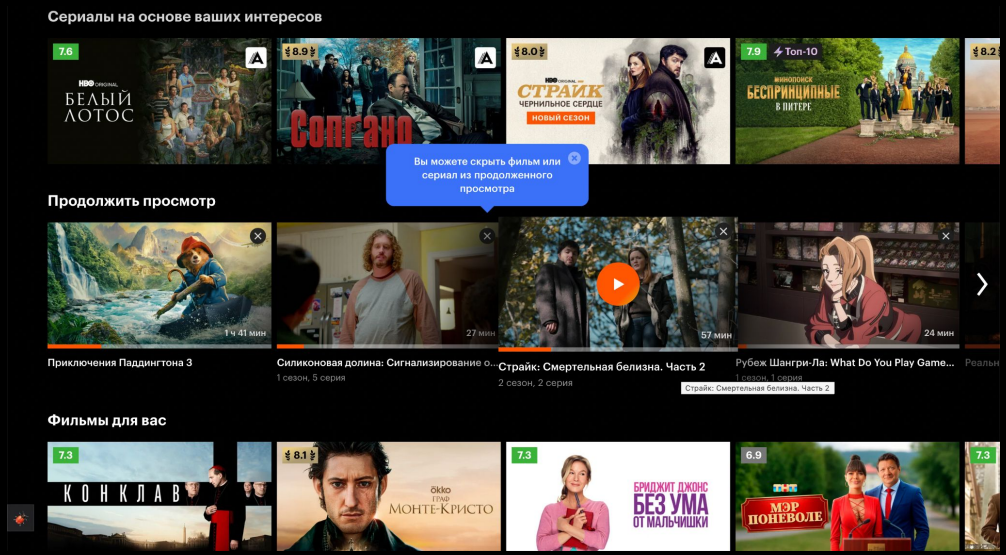
- **Главный KPI — подписка**
- **Прямая оптимизация не работает (долгий цикл)**
- **Делают прокси-метрики аналитически**
- **Таргет может быть существенно сложнее**

Фильтрация

- По типу подписки (Плюс, Амедиатека, Плюс Про Макс и т.д.)
- По лицензии айтема
- Лицензия отличается и по гео, ну вы поняли

Интерфейс: много каруселей

- Общая персональная выдача
- Продолжить просмотр
- Похожие на...
- Жанровые блоки

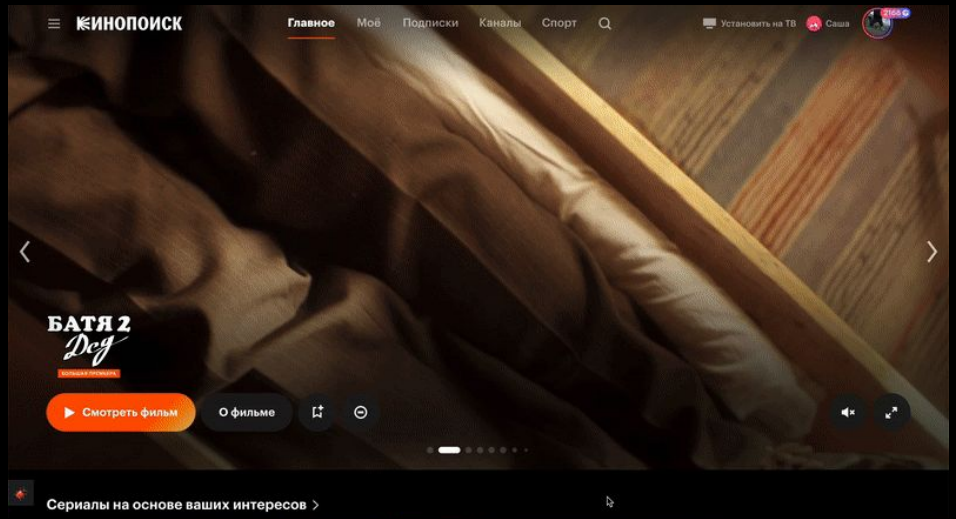


Два подхода поранжировать карусели:

1. Выбрать блоки → ранжировать внутри

2. Генерировать общий пул → нарезать продуктово или еще как-то

Разнородный контент



Но бывают

- **продолжить просмотр**
- **трансляции спорта**
- **один огромный постер с фильмов**

Куча разнородных каруселей, как их поранжировать?

Разнородный контент

Подход с профицитом

- Считаем "выигрыш" и "проигрыш" для блока
- Учитываем размер, клики ниже и т.д.
- Негативный сигнал → штраф

Проблемы

- Каскадность моделей
- Тяжело поддерживать

Бандиты

- Хорошо подходят для гетерогенных блоков
- Работают онлайн, переобучаются сами
- В гугле много используют

Минусы:

- Сложно отладить и поддерживать

Холодный старт

Много пользователей с нулевой историей (бесплатный Плюс и т.д.)

Что делать:

- **модель должна сходиться от пустых фичей**
- **балансировка в пулах**
- **Онбординг**

Пример (Netflix):

- **Показывают топ популярных фильмов**
- **Юзер кликает — забираем фильм и удаляем все события этого юзера**
- **Перестраиваем топ → повторяем**

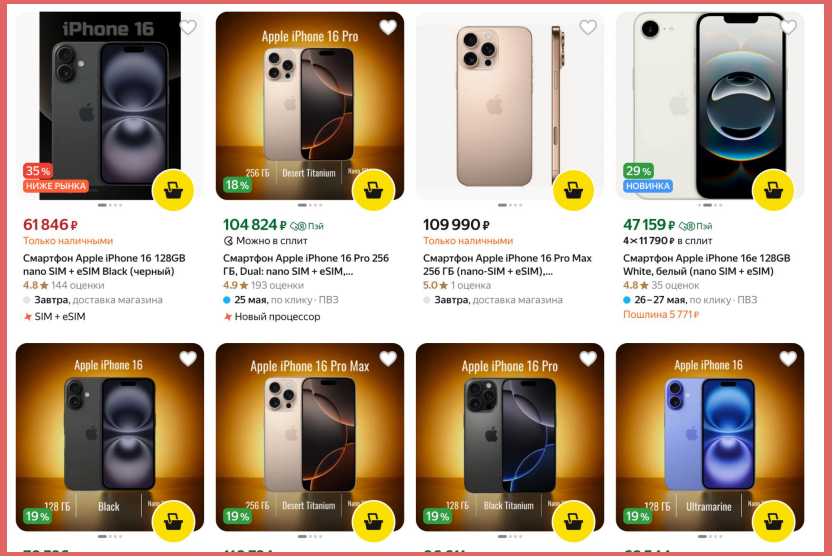
Рантайм-процессинг — тоже не первые 80 процентов

- Люди редко смотрят 2 фильма подряд
- Но могут долго выбирать
- Если ничего не кликает — надо менять выдачу

Яндекс Маркет – много товаров, много проблем

Что усложняет масштаб?

- **Объем данных: десятки миллионов товаров**
- **Много сущностей: модели, СКЮ, продавцы**
- **География: регионы, логистика, наличие**
- **Бизнес-ограничения: реклама, гарантии показов**



Объем и структура айтемов

В Маркете — десятки миллионов товаров. Это выходит за пределы одного шарда, требуется масштабируемая архитектура хранения и обработки.

Кроме того, товары не всегда уникальны:

- **Модель** — обобщённое описание товара (например, смартфон определённой серии)
- **СКЮ** — конкретная вариация модели (объём памяти, цвет)
- **Оффер** — сверху еще продавец + цена

Иерархическая структура позволяет уменьшить дублирование, но добавляет сложность в логике рекомендаций.

Что происходит обычно

Рекомендательный пайплайн делится на два этапа:

1. **Определение релевантных моделей**
2. **Подбор валидных СКЮ и потом офферов с учётом региона, цены, доставки**

Важно как можно раньше исключать модели без подходящих СКЮ.

Фильтрации должны выполняться до применения ML-моделей:

- **Регион пользователя**
- **Наличие у продавца**
- **Условия доставки**
- **Бизнес-ограничения**

Продавцы и гарантии показов

В системе присутствует вторая сущность — продавец , со своими интересами и ожиданиями. Продавцы рассчитывают на получение показов в рекомендациях. Их отсутствие вызывает жалобы и эскалации .

Типичные подходы:

- Встроенный item-level exploration в модель (какие-то RL-варианты)
- PID-контроллеры для стабилизации объёма показов
- Явные квоты на показы в системе

Exploration и работа с новыми товарами

Exploration необходим для:

- Холодных (новых) товаров
- Продавцов без истории
- Товаров из long-tail

Проблема: чрезмерный exploration снижает качество выдачи.

Решения:

- Подмешивание ограниченного числа товаров
- Использование обобщающих моделей
- Сегментация пользователей
- A/B-контроль и аккуратная валидация

Архитектура пайплайна

1. **Кандидатогенерация** (ориентирована на полноту, быстрые методы)
2. **Фильтрация** (по региону, доступности, логистике, правилам)
3. **Ранжирование** (оптимизация под метрику — GMV, CTR, маржинальность)

Дополнительно могут использоваться:

- **Дополнительные уровни переранжирования**
- **Постобработка (например, диверсификация, слоты под рекламу)**

Реклама и сквозное ранжирование

При таком масштабе реклама — обязательный компонент .

Два базовых подхода:

1. **Фиксированные рекламные позиции + локальный аукцион**
2. **Сквозное ранжирование: органика и реклама объединяются в одном списке**

Базовая формула может выглядеть так:

score = p(buy) × margin + bid

Ставка продавца учитывается наряду с вероятностью покупки и ожидаемой выгодой.

Важно: обложиться гардрейлами

Калибровка вероятностей

Ранжирующие модели часто выдают не вероятность, а скор. Чтобы использовать их в бизнес-метриках или в аукционе, требуется калибровка :

Обычно:

- **Изотоническая регрессия**
- **Platt scaling**
- **Как вам по кайфу**

Калиброванные значения позволяют использовать предсказания в денежно-ориентированных формулах и корректно оценивать влияние рекламы.

Влияние вертикалей

Маркет состоит из множества товарных вертикалей:

- **Электроника**
- **Одежда**
- **Автотовары**
- **Строительство и ремонт**

В разных вертикалях:

- **Разная динамика (fast fashion vs. долгоживущая техника)**
- **Разный подход к эксплору**
- **Разные ограничения по актуальности, сезонности и цене ошибки**

Архитектура должна поддерживать конфигурацию по вертикалям.

Дзен – много UGC-контента

Масштабы

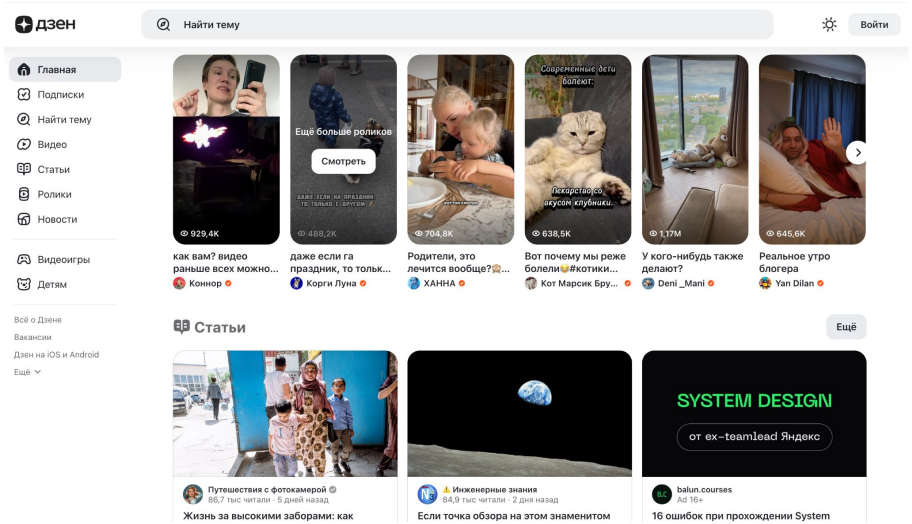
- **Сотни миллионов айтемов, десятки тысяч новых ежедневно**
- **Всё, о чём говорили до этого — здесь тоже есть:**
  - **счастье продавца**
  - **счастье автора**
  - **наливки**
  - **рантайм-процессинг**

Что появляется нового?

- **UGC: пользовательский контент → требует объяснимости**
- **Объяснимость важна: нужно объяснить, почему именно этот контент**
- **Баланс объяснимости и релевантности становится продуктовой задачей**
- **Мощный пайплайн модерации:**
  - **автоматизация + толока**
  - **контент может быть токсичным/опасным**

Мультимодальность

- **Типы контента:**
  - **статьи**
  - **видео**
  - **шортсы**
- **Всё замешано в одном фиде**

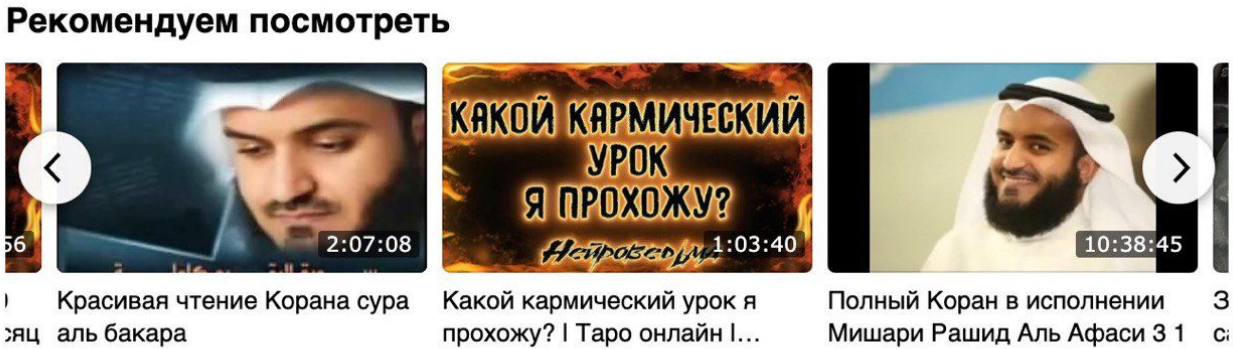


Интерфейс влияет на таргет

На больших интерфейсах (лента, подборки) можно оптимизировать на длинные сессии

На маленьких (один видос на экране или даже карусель) — скорее нет:

- **длинные статьи/видео будут скипаться**
- **получаем артефакты (длинные видео)**



Рантайм-профиль пользователя

- **Очень важно: оффлайн == онлайн**
- **Иначе: A/B тесты врут, фичи не работают**

Часто решается через лямбда-архитектуру:

- **key-value хранилище (день)**
- **Redis или что-то подобное (почасовые апдейты)**
- **на каждый запрос собирается свежий профиль**

Вынос кандидатов и фичей

- **Кандидатогенерация и фичестор становятся отдельными сервисами**
- **Особенно важно при:**
  - **тяжёлой генерации**
  - **экономии памяти и CPU**
- **Позволяет шарить генерацию между интерфейсами**

Деградация моделей и свежесть

- **В ZEN быстро появляются и исчезают тренды**
- **Модели деградируют быстрее, чем в e-commerce**
- **Помогает:**
  - **обучение в рантайме**
  - **online fine-tuning**
  - **конвейеры**

Антифрод и метрики

- Авторам платят за показы → начинается фрод
- Крутят ботов, накручивают CTR
- В хорошей системе таргет антифрод-aware, как и целевая метрика
- Команды работают очень тесно

Индексы для скорости

- **Обычный шард — долгоживущие айтемы**
- **Быстрый шард — живет около суток:**
  - **ограниченное число фич**
  - **загружается мгновенно**
- **Могут быть отдельные модели с квотированием для него**

Semi-online рекомендации

Список рекомендаций хранится и пересчитывается фоном

- **Обновление по триггерам**
- **пример**
  - **смотрим, сколько айтемов просмотрено**
  - **каждая степень двойки → триггер**
  - **считаем и кладев резы к кв**
- **позволяет:**
  - **использовать тяжёлые модели**
  - **держать быстрое время ответа**

Спасибо!

Даня Ткаченко,
Служба ML-сервисов Лавки,
Белград 2025