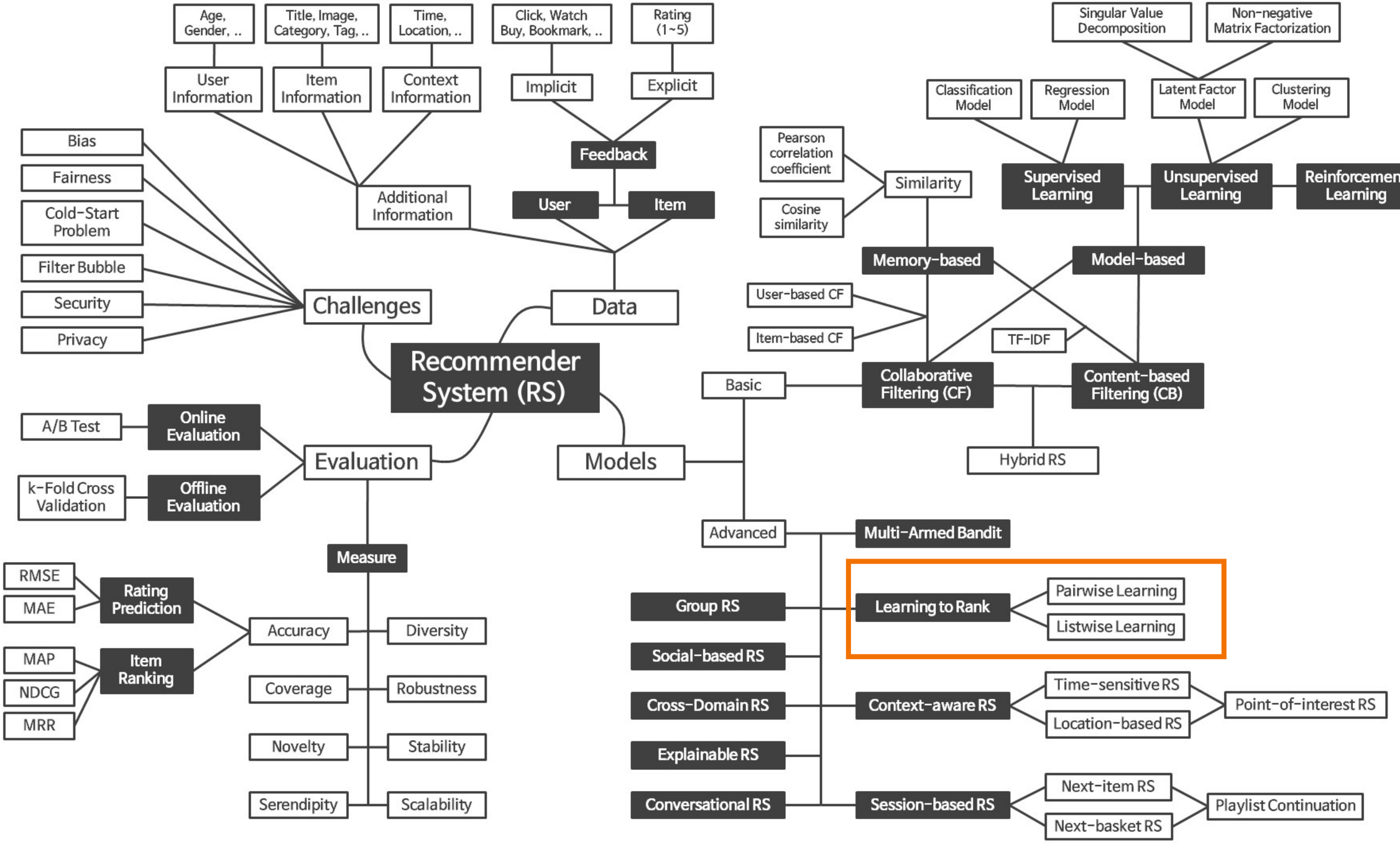
LTR, двухуровневые ранжирующие системы

Красно Александр, 24.03.2025, AI masters


Recommender System (RS)
Challenges
Bias
Fairness
Cold-Start Problem
Filter Bubble
Security
Privacy
Data
Additional Information
User Information
Item Information
Context Information
Feedback
Implicit
Explicit
User
Item
Age, Gender, ..
Title, Image, Category, Tag, ..
Time, Location, ..
Click, Watch Buy, Bookmark, ..
Rating (1~5)
Evaluation
Online Evaluation
A/B Test
Offline Evaluation
k-Fold Cross Validation
Measure
Rating Prediction
RMSE
MAE
Item Ranking
MAP
NDCG
MRR
Accuracy
Diversity
Coverage
Robustness
Novelty
Stability
Serendipity
Scalability
Models
Basic
Collaborative Filtering (CF)
Similarity
Pearson correlation coefficient
Cosine similarity
Memory-based
User-based CF
Item-based CF
Model-based
Supervised Learning
Classification Model
Regression Model
Unsupervised Learning
Latent Factor Model
Clustering Model
Singular Value Decomposition
Non-negative Matrix Factorization
Reinforcement Learning
Content-based Filtering (CB)
TF-IDF
Hybrid RS
Advanced
Multi-Armed Bandit
Learning to Rank
Pairwise Learning
Listwise Learning
Group RS
Social-based RS
Cross-Domain RS
Explainable RS
Conversational RS
Context-aware RS
Time-sensitive RS
Location-based RS
Point-of-interest RS
Session-based RS
Next-item RS
Next-basket RS
Playlist Continuation

LTR, постановка задачи

LTR - Learning to rank

Пусть $X = \{x_1, \ldots, x_n\}$ - обучающая выборка

отношение $i \prec j$ - x_j "лучше" x_i между документами из X

найти **ранжирующую функцию** $f : X \to \mathbb{R}$, которая восстанавливает порядок
$i \prec j \Rightarrow f(x_i) < f(x_j)$

Простой пример: линейная функция $f(x, w) = \langle x, w \rangle$

LTR, примеры

I - коллекция документов

U - пользователи

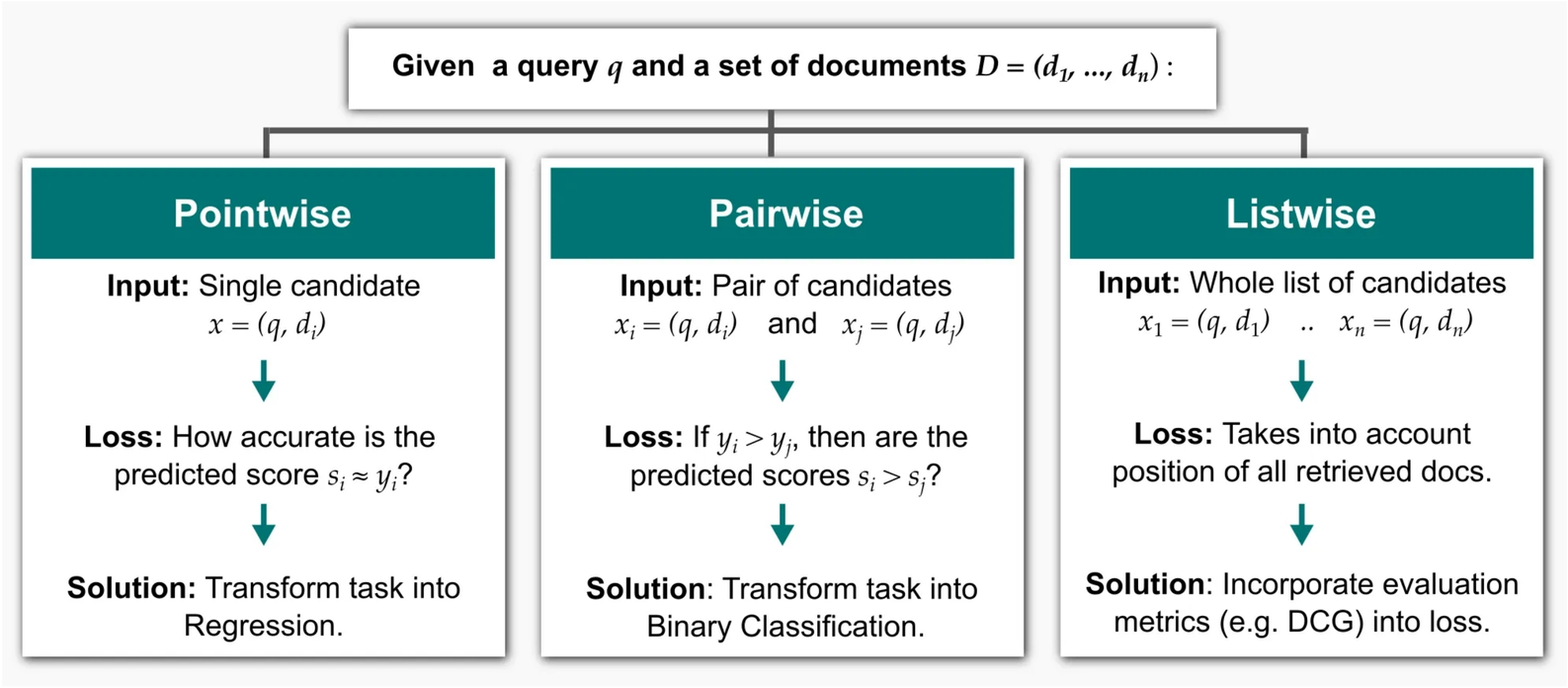
$X = U \times I$ - пары пользователь-документ

$(u, i_k) \prec (u, i_m) \Rightarrow f(u, i_k) < f(u, i_m)$

т.е. рекомендации пользователю u - список документов i, упорядоченный функцией $f(u, i)$

LTR, функции ранжирования

- point-wise (точечная оценка, например вероятность клика)
- pair-wise (попарное сравнение)
- list-wise (оценка списка, группы)



LTR, функции ранжирования

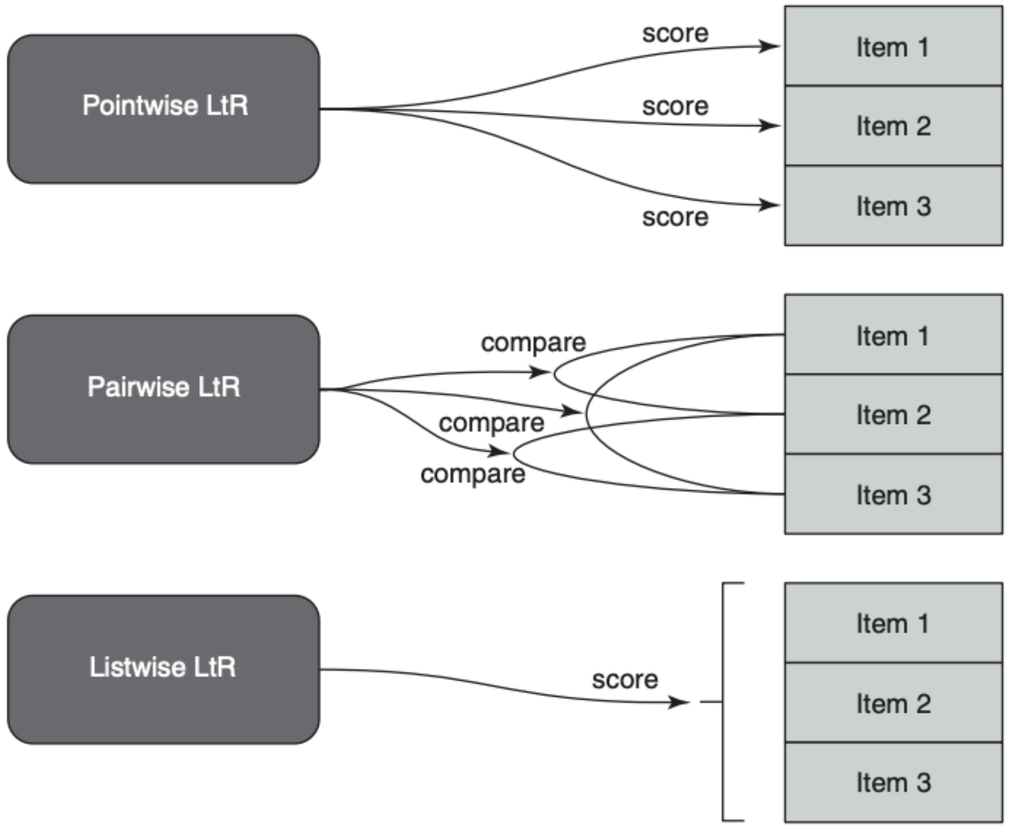
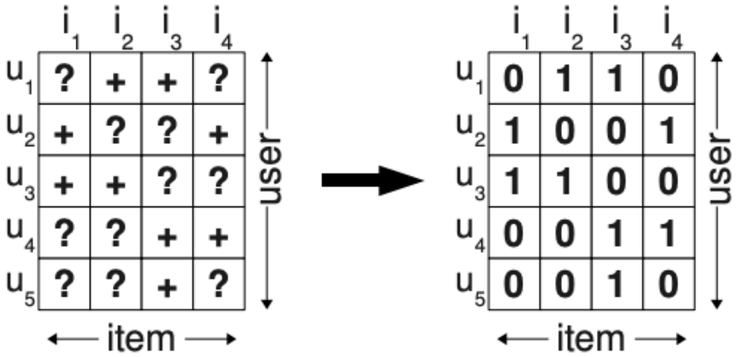
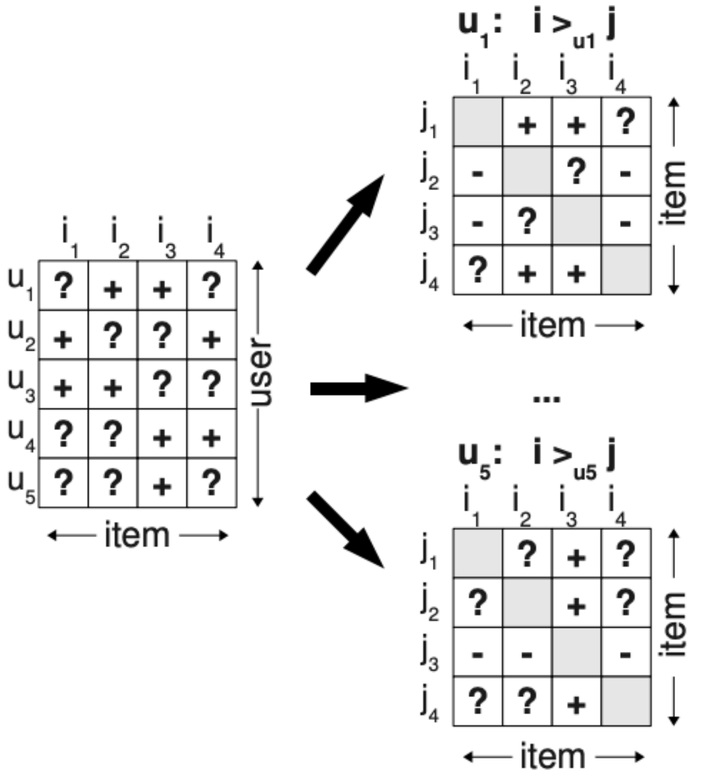


Figure 13.5 The three different subgroups of LTR algorithms: pointwise, pairwise, and listwise

Practical Recommender Systems

LTR, BPR





$$D_S := \left\{(u, i, j) \mid i \in I_u^+ \wedge j \in I \backslash I_u^+\right\}$$

$$L_{bpr} = -\frac{1}{D_S} \sum_{j=1}^{D_S} \log \sigma\left(r_i - r_j\right) + reg \quad \sigma(x) = \frac{1}{1 + e^{-x}}$$

BPR: Bayesian Personalized Ranking from Implicit Feedback

LTR, WARP

Интуиция алгоритма WARP - как и в BPR работаем с триплетами из D_S

1. Для (u, j) юзер-позитивный документ выбираем случайный негатив i среди всех остальных документов. Считаем предсказание - если порядок нарушился - обновляем градиент, иначе ищем другой негатив.

2. Чем быстрее находим негатив i, тем больше шаг градиента можем сделать.

Learning to Rank Recommendations with the k-Order Statistic Loss

LTR, WARP

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

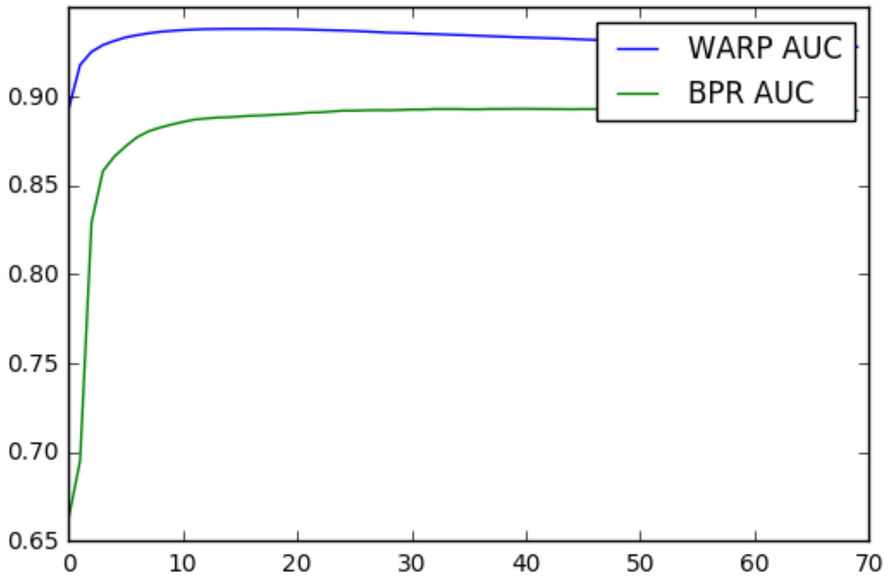
We are given a probability distribution P of drawing the i^{th} position in a list of size K. This defines the choice of loss function.
Pick a user u at random from the training set.
Pick $i = 1, \ldots, K$ positive items $d_i \in \mathcal{D}_u$.
Compute $f_{d_i}(u)$ for each i.
Sort the scores by descending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

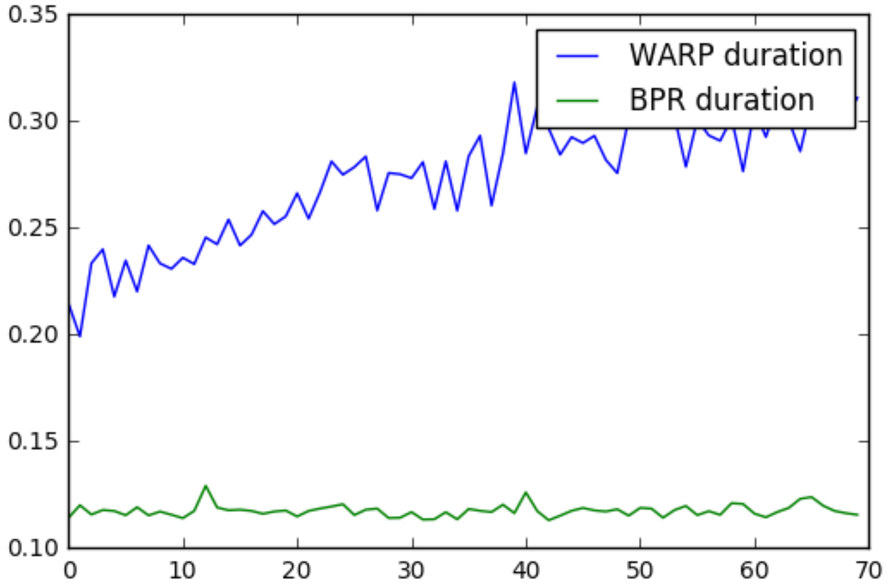
Algorithm 2 K-OS WARP loss

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.

Learning to Rank Recommendations with the k-Order Statistic Loss

LTR, BPR vs WARP





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

LTR, RankNet

$$Pr(i \succ j) = P_{ij} \equiv \frac{1}{1 + e^{-(s_i - s_j)}}, s_i = f(q, d_i), s_j = f(q, d_j)$$

$$L = -\sum_{i \neq j} \frac{1}{2}(1 + S_{ij}) log P_{ij} + \frac{1}{2}(1 - S_{ij}) log(1 - P_{ij}), S_{ij} \in \{-1, 0, 1\}$$

https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/MSR-TR-2010-82.pdf

LTR, RankNet

Факторизация градиента

$$\frac{\partial L}{\partial w_k} = \sum_{\{i,j\}} \left[\frac{\partial L}{\partial s_i} \frac{\partial s_i}{\partial w_k} + \frac{\partial L}{\partial s_j} \frac{\partial s_j}{\partial w_k} \right]$$

$$= \sum_i \frac{\partial s_i}{\partial w_k} \left(\sum_{\forall j < i} \frac{\partial L(s_i, s_j)}{\partial s_i} \right) + \sum_j \frac{\partial s_j}{\partial w_k} \left(\sum_{\forall i > j} \frac{\partial L(s_i, s_j)}{\partial s_j} \right)$$

$$\frac{\partial L(s_i, s_j)}{\partial s_i} = -\frac{\partial L(s_i, s_j)}{\partial s_j} = \sigma\left[\frac{1}{2}(1 - S_{ij}) - \frac{1}{1 + e^{s_i - s_j}}\right]$$

$$\frac{\partial L}{\partial w_k} = \sum_i \frac{\partial s_i}{\partial w_k} \left[\sum_{\forall j < i} \frac{\partial L(s_i, s_j)}{\partial s_i} + \sum_{\forall j < i} \frac{\partial L(s_j, s_i)}{\partial s_i} \right]$$

$$= \sum_i \frac{\partial s_i}{\partial w_k} \left[\sum_{\forall j < i} \frac{\partial L(s_i, s_j)}{\partial s_i} - \sum_{\forall j > i} \frac{\partial L(s_j, s_i)}{\partial s_j} \right] = \sum_i \frac{\partial s_i}{\partial w_k} \lambda_i$$

https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/MSR-TR-2010-82.pdf

LTR, RankNet

Факторизация градиента

- Для каждого документа в данном запросе существует компонент вектора градиента, который мы обозначили как λ, которая вычисляется путем рассмотрения всех превосходящих и низшие документы по сравнению с ним
- Относительно худший документ будет толкать текущий документ вверх, а относительно лучший документ будет сдвинет его вниз.
- Во время обучения вместо обновления по каждой паре документов, мы можем обновлять по каждому запросу
- λ намного дешевле в вычислениях, весь процесс обучения может значительно ускориться

https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/MSR-TR-2010-82.pdf

LTR, LambdaRank

- Проблема с RankNet в том, что оптимизируется число попарных ошибок, а это не всегда то, что нужно.
- Как оптимизировать NDCG?

https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/MSR-TR-2010-82.pdf

LTR, LambdaRank

$$\lambda_i = \left[\sum_{\forall j \prec i} \frac{\partial L(s_i, s_j)}{\partial s_i} - \sum_{\forall j \succ i} \frac{\partial L(s_j, s_i)}{\partial s_j} \right]$$

$$= \left[\sum_{\forall j \prec i} \lambda_{ij} - \sum_{\forall j \succ i} \lambda_{ij} \right]$$

$$\lambda_{ij} \equiv \frac{\partial L(s_i, s_j)}{\partial s_i} \cdot |\Delta NDCG_{ij}|$$

https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/MSR-TR-2010-82.pdf

LTR, реализации

- BPR https://github.com/Nemexur/revisit-bpr
- WARP https://making.lyst.com/lightfm/docs/examples/warp_loss.html
- LambdaRank https://github.com/haowei01/pytorch-examples/blob/master/ranking/LambdaRank.py, https://lightgbm.readthedocs.io/en/latest/pythonapi/lightgbm.LGBMRanker.html
- YetiRank/YetiRankPairwise (посмотрим на семинаре) https://catboost.ai/docs/en/concepts/loss-functions-ranking#YetiRank

LTR, вопросы

Ранжирующие системы

Вводные

- хотим работать в реалтайме
- хотим отвечать (отдавать список документов) быстро
- хотим в момент запроса рассмотреть максимальное количество документов-кандидатов
- хотим учесть всю актуальную информацию, которая у нас есть

Ранжирующие системы

Ограничения

- время ответа системы (10 - 1000 ms)
- большая база документов (1kk+)
- бизнес ограничения
- ограничения по железу

Многоуровневая система

retrieval

filtering

ranking

re-ranking

Retrieval

- кандидатогенерация/кандген/l1/retrieval
- отбор кандидатов из полного каталога документов
- сужаем воронку кандидатов до сотен/тысяч
- быстрые операции
  - достать по ключу из хэш-таблицы
  - ann/knn на gpu
  - быстрые модели расчета эмбединга
- оптимизируем Recall

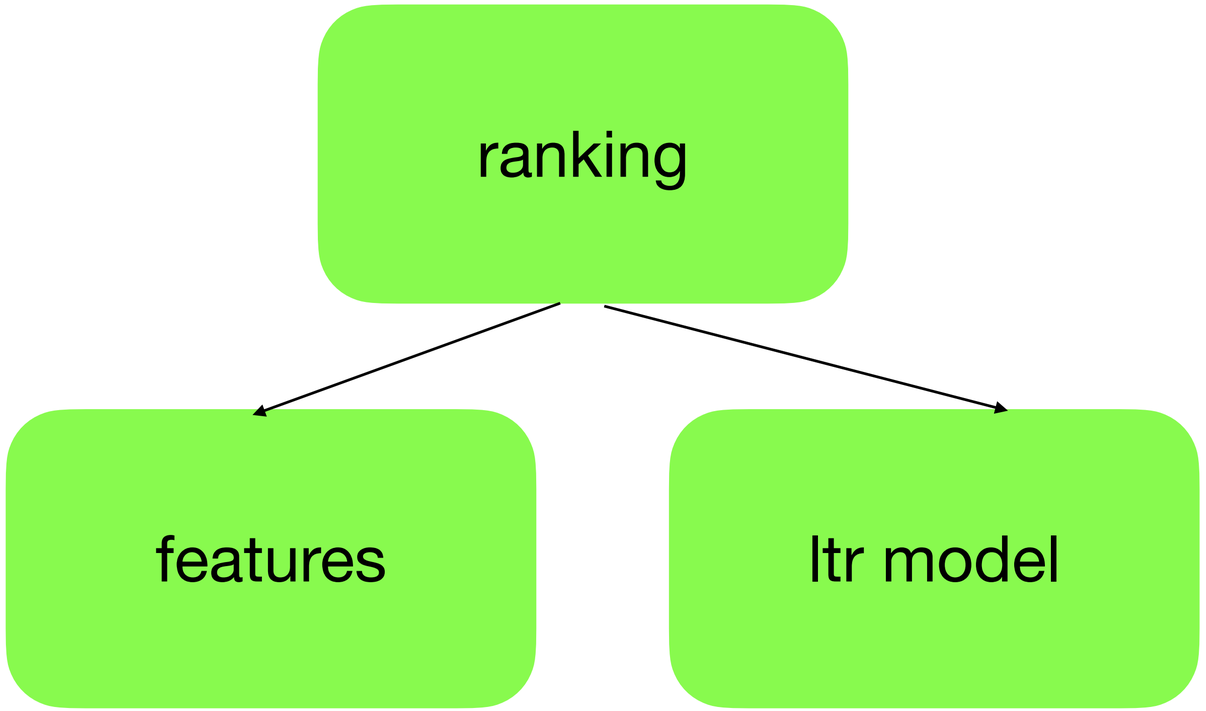
Filtering

- фильтрация по бизнес правилам (не показывать лекарства, 18+)
- фильтрация на доступность
- фильтрация при пагинации - фильтр товаров, которые были выше в ленте

Ranking

- ranking/l2
- задача отобрать самых релевантных 10-1000 документов для re-ranking
- более сложные модели LTR (boosting, NN)
- расчет признаков $Q \times U \times I$
- NDCG, MAP

Ranking



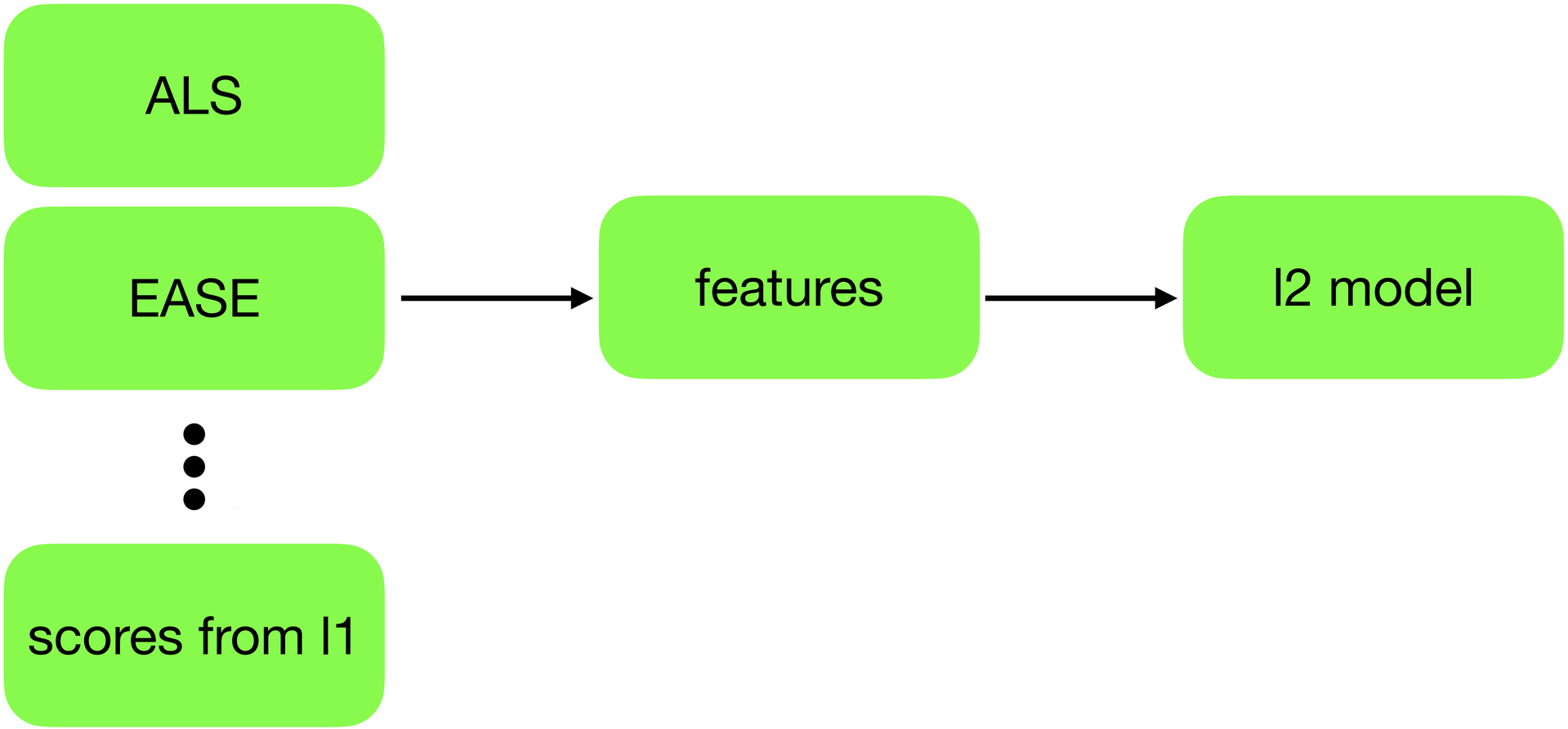
CatBoost LightGBM

Ranking, Features

- I - конверсии, рейтинг, категориальная информация
- $U \times I$ - парные статистики, ценовые предпочтения, embedding dist
- $Q \times I$ - парные статистики, embedding dist
- $Q(I) \times I$ - item2item задача, embedding dist, совстречаемость в сессиях

Ranking, Features

Стекинг



Re-ranking

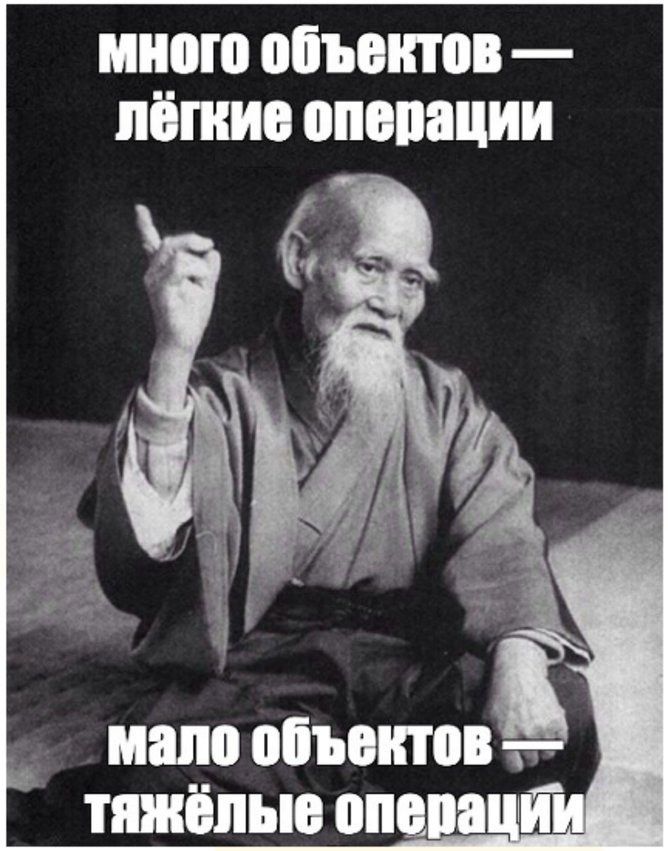
- re-ranking/l3
- задача сформировать финальный топ выдачи
- учесть разнообразие*, рекламные ставки (аукционы FPA, VCG)
- оценка кандидата с учетом места в финальной выдаче
- Diversity, Novelty, Serendipity

Многоуровневая система

Базовый принцип

Воспользуемся мудростью сенсея (он справа →)

- Нижние стадии → много объектов → самые лёгкие операции
- Отсекать как можно больше объектов и как можно раньше
- Нижние стадии «глупые» → нельзя потерять важные объекты
- Чем дальше — тем более тяжёлые операции можем выполнять



Я team

https://habr.com/ru/companies/yandex/articles/816205/

Retrieval-Ranking

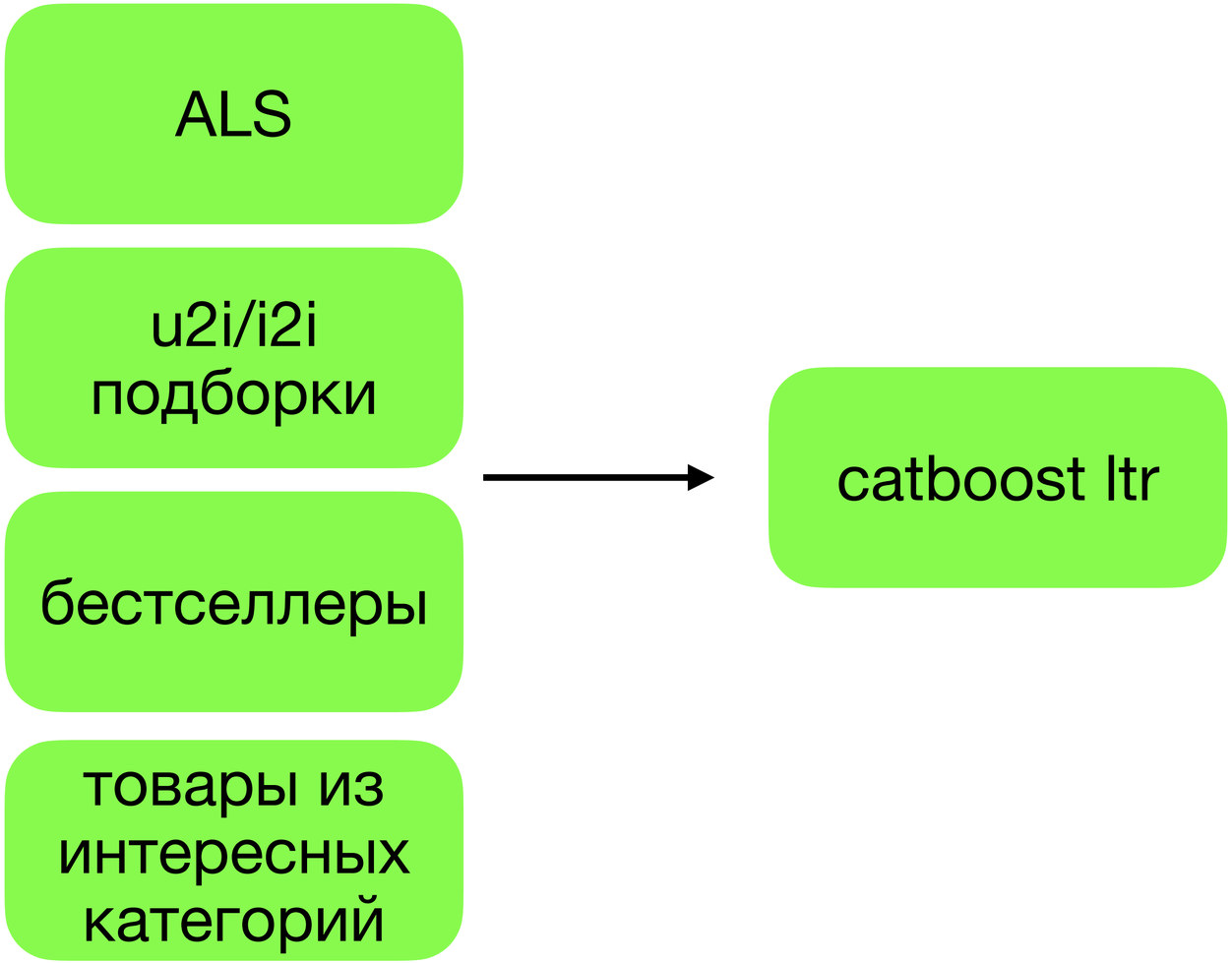
Другие определения

Task		Specification (Inputs / Outputs)
Ranking	x_is	$\Phi_0, a_0, \Phi_1, a_1, \ldots, \Phi_{n_c-1}, a_{n_c-1}$
	y_is	$a_0, \varnothing, a_1, \varnothing, \ldots, a_{n_c-1}, \varnothing$
Retrieval	x_is	$(\Phi_0, a_0), (\Phi_1, a_1), \ldots, (\Phi_{n_c-1}, a_{n_c-1})$
	y_is	$\Phi'_1, \Phi'_2, \ldots, \Phi'_{n_c-1}, \varnothing$ ($\Phi'_i = \Phi_i$ if a_i is positive, otherwise $\varnothing$)

Actions Speak Louder than Words: Trillion-Parameter Sequential Transducers for Generative Recommendations

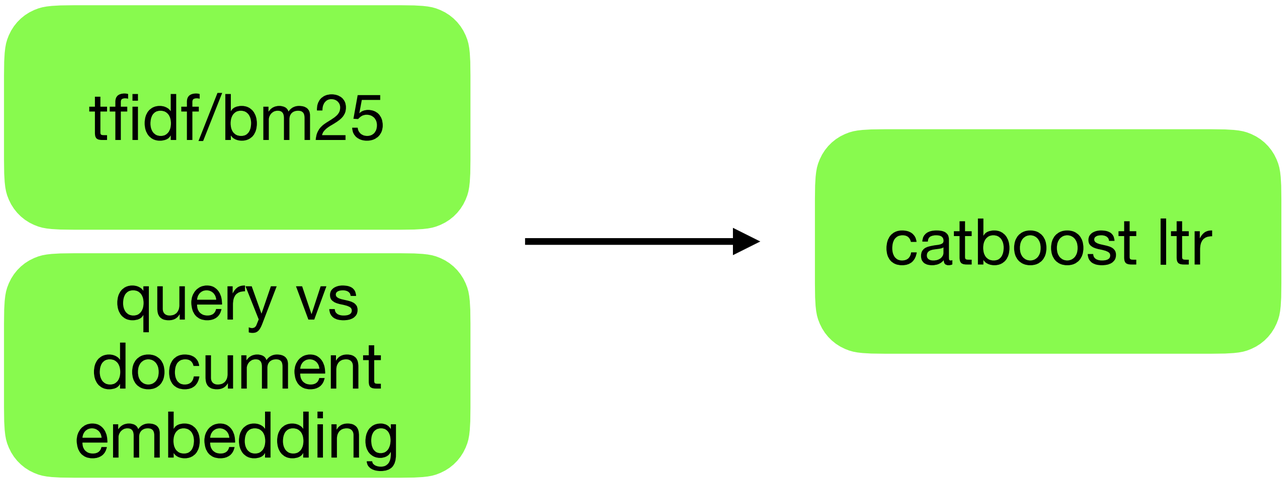
Многоуровневая система

Базовый пример рексиса



Многоуровневая система

Базовый пример поиска



Retrieval, прочее

Несколько источников

Проблема: если источников больше 1, сколько кандидатов набирать из каждого?

Простое решение - задавать константами (ALS: 1000, I2I: 1000, TopPopular: 500) и проверять различные конфигурации через AB тесты.

Решение с ml - отдельной моделью предсказывать какой источник выбрать/пропорцию кандидатов из каждого источника

Retrieval, прочее

Легкое ранжирование

Иногда между этапами retrieval и ranking добавляют этап легкого ранжирования.

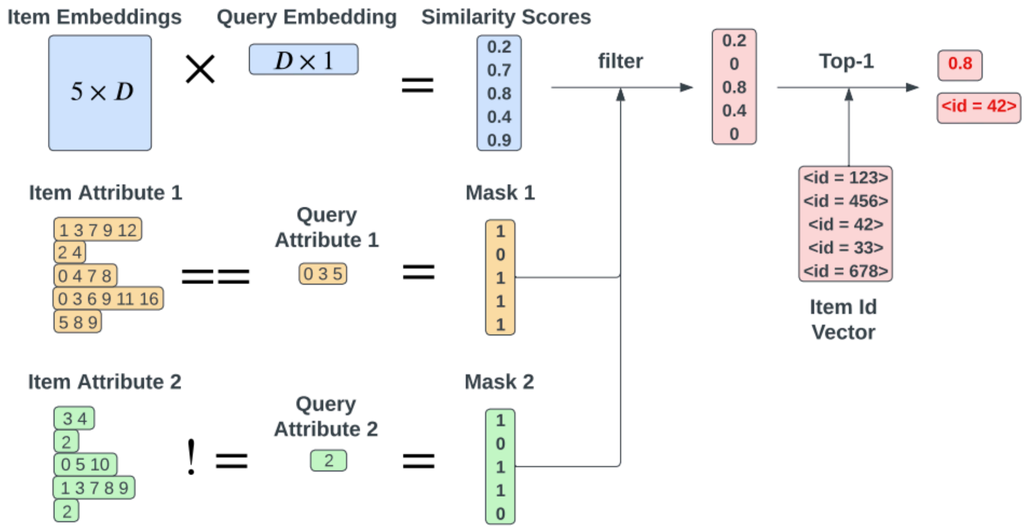
На этом этапе считаем меньше и легче признаки + более простая модель (линейная или бустинг с меньшим кол-вом деревьев)

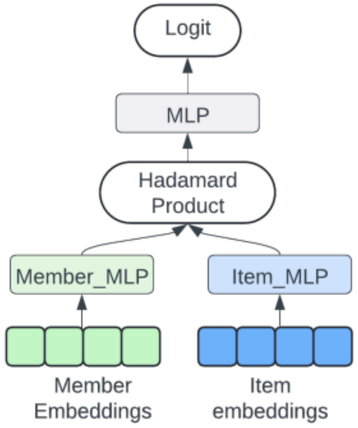
Можно использовать дистилляцию (угадывать легкой моделью скоры тяжелой/финальный скор*)

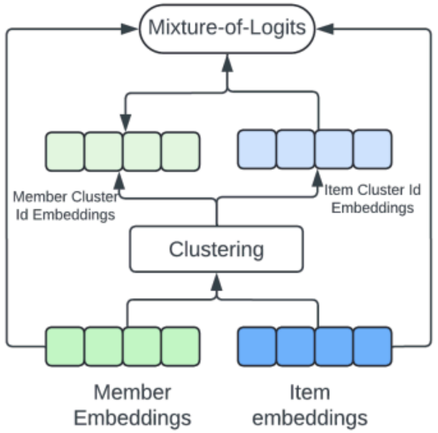
https://runtime.strm.yandex.ru/player/video/vplv5zetmiwwdym7bcv3?autoplay=0&mute=1

Retrieval, прочее

ANN -> KNN



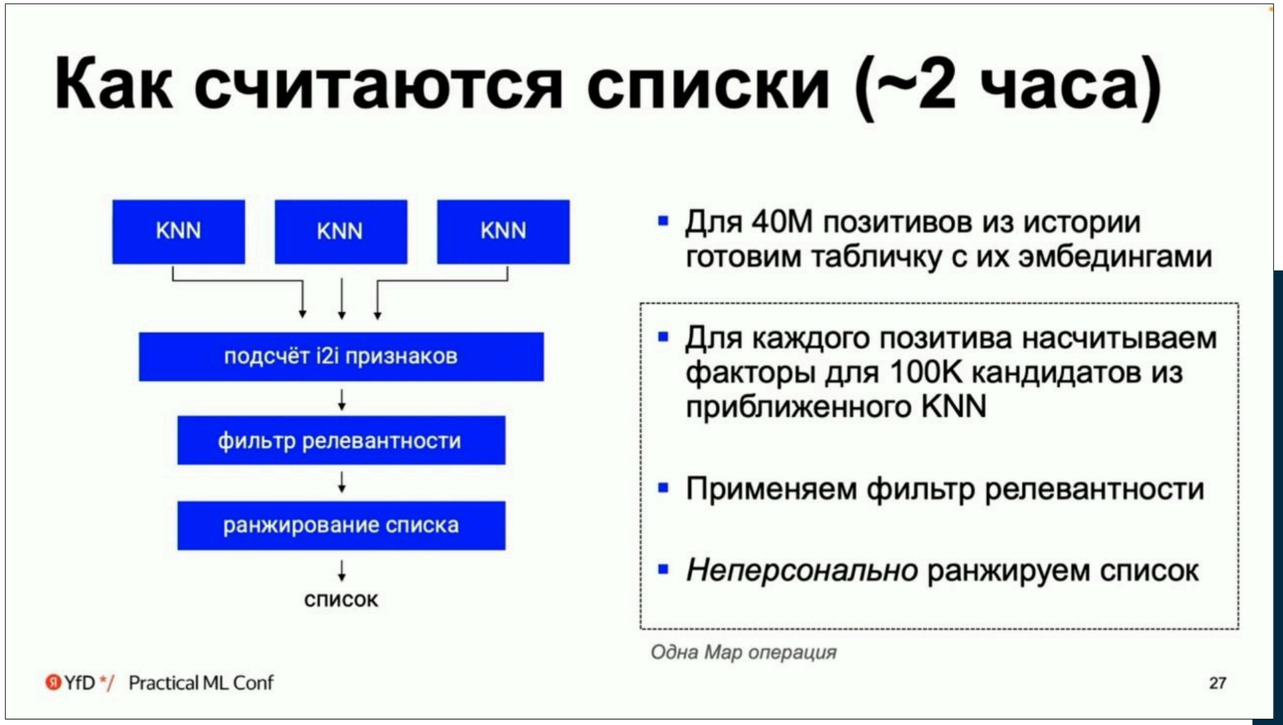




LiNR: Model Based Neural Retrieval on GPUs at LinkedIn

Retrieval, прочее

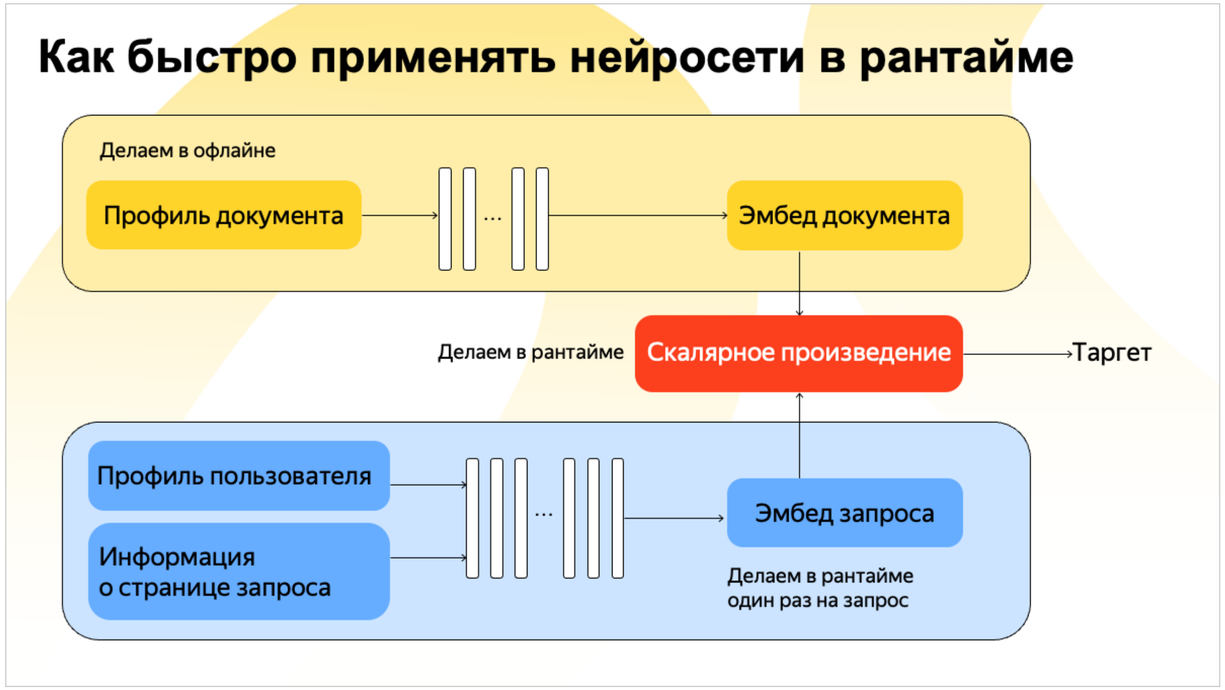
Быстрые операции не значит легкие модели

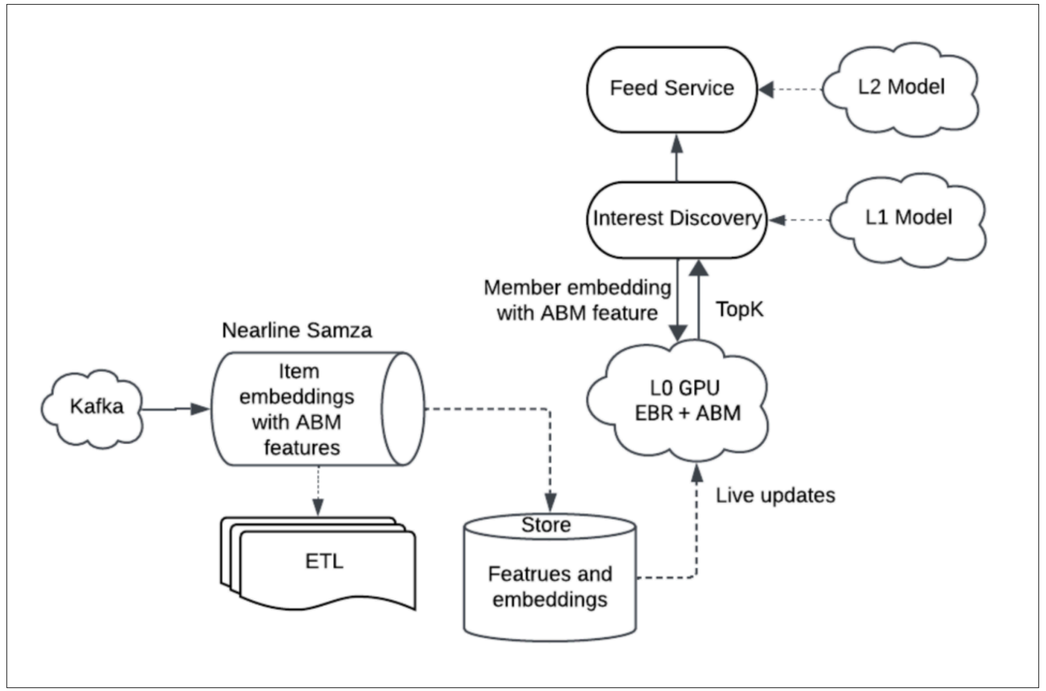


https://youtu.be/e-eZIRMgvIE?si=Ze2SSq8p17Ygye62

Retrieval, прочее

Разделяй офлайн и онлайн

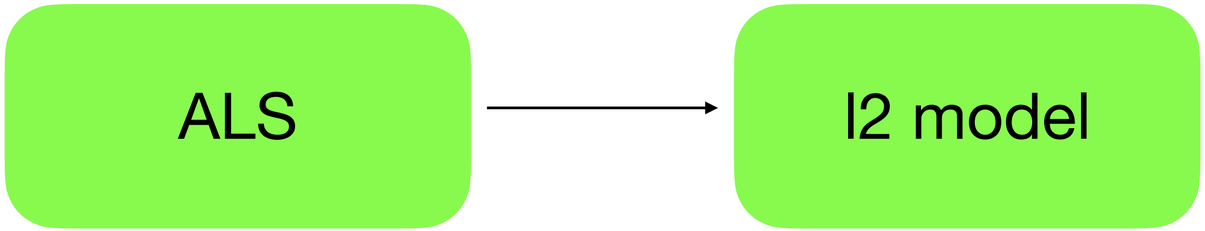




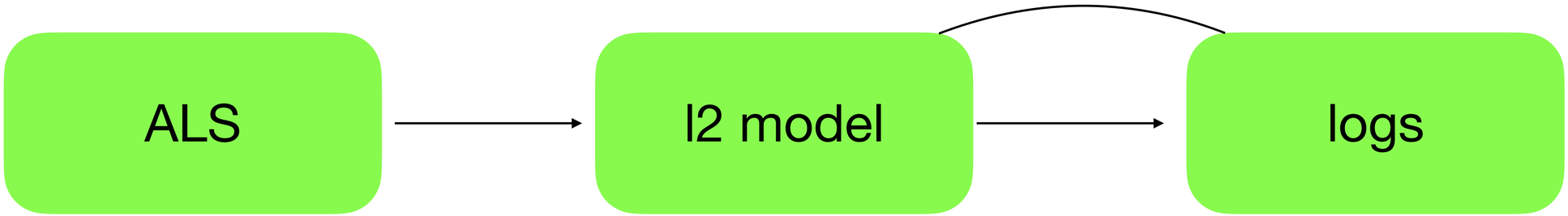
Многоуровневая система

Как это все учить?

- offline timesplit



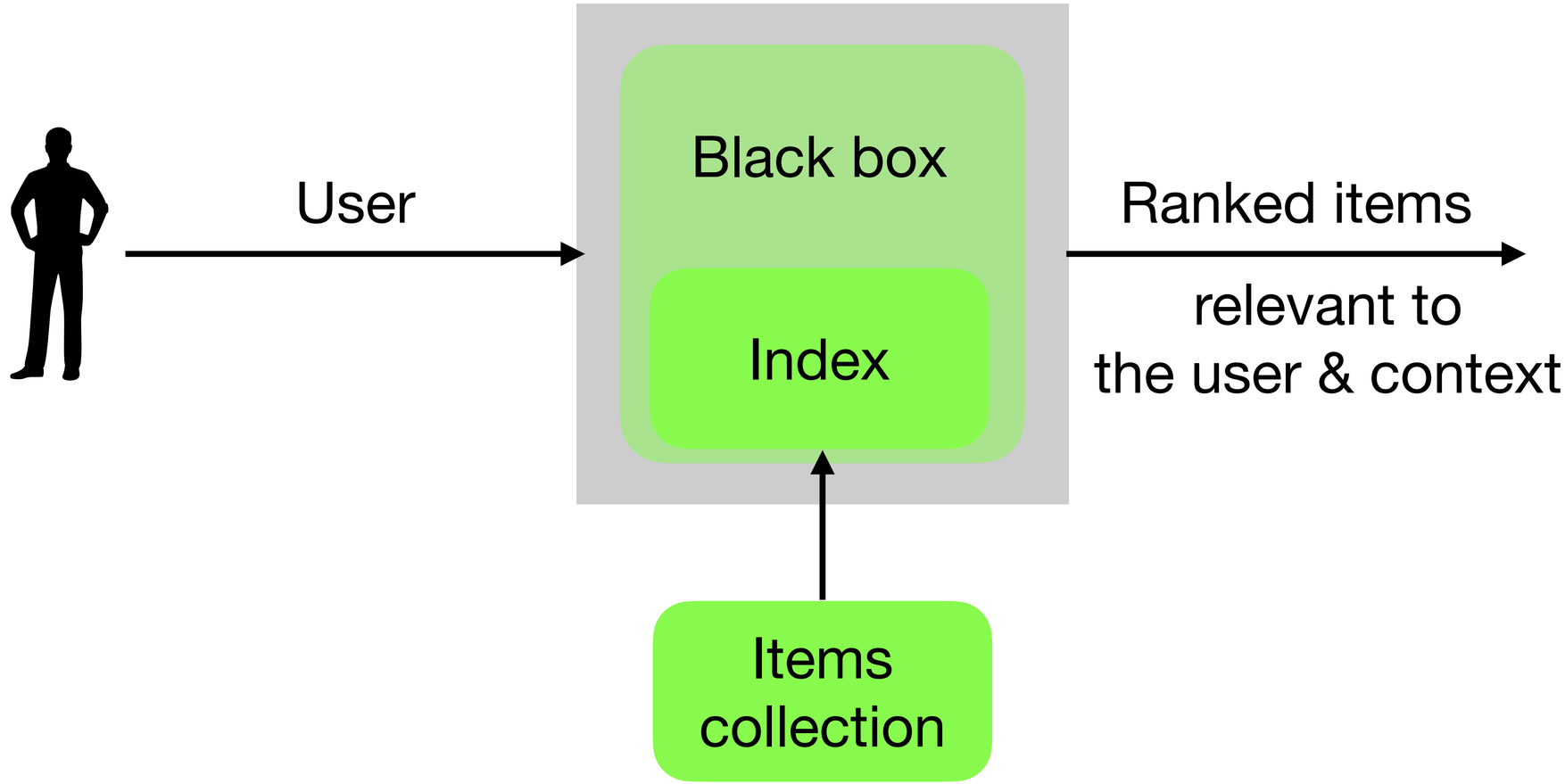
- на логах сервиса



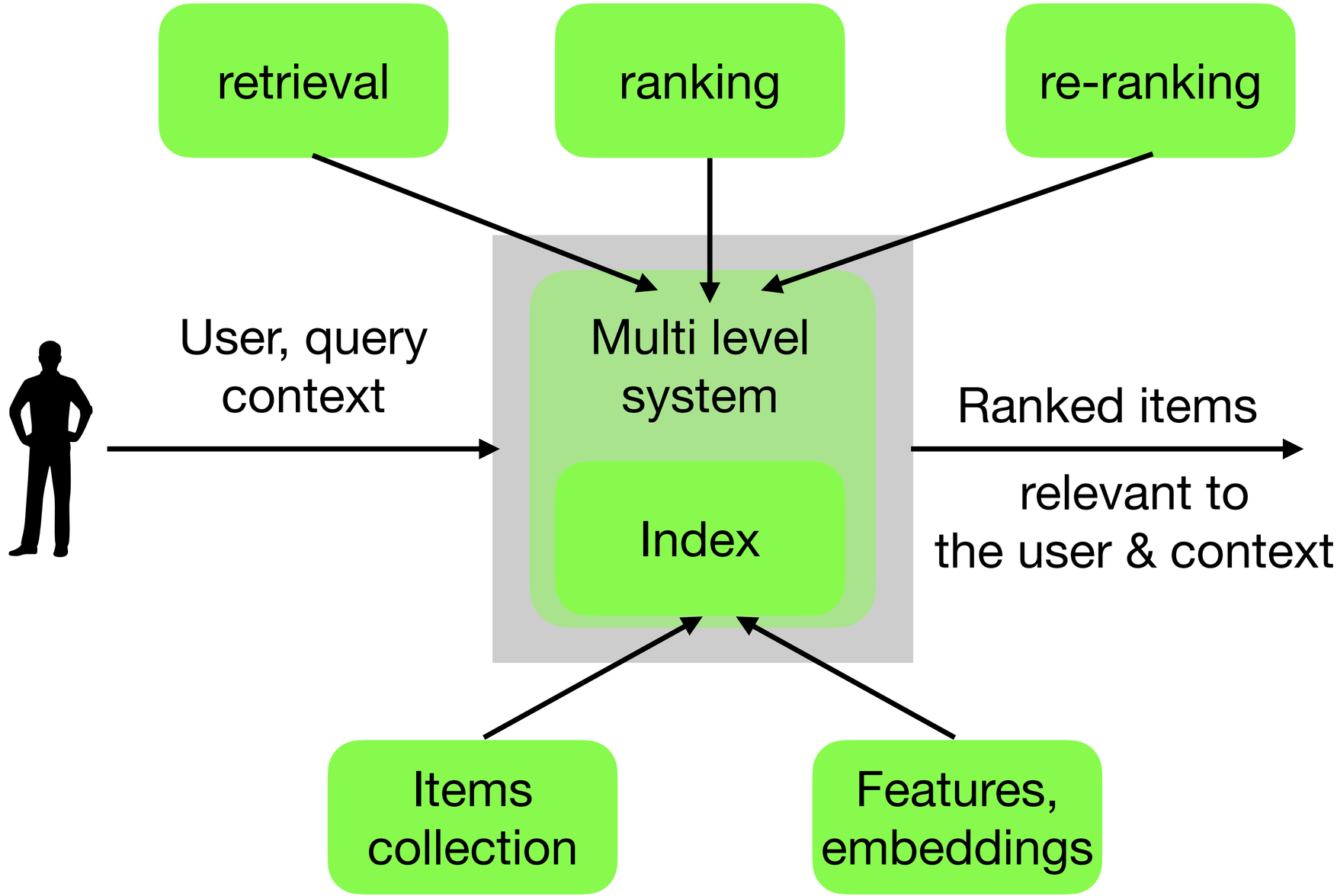
Многоуровневая система

Как это все учить?

Многоуровневая система



Многоуровневая система



Вопросы