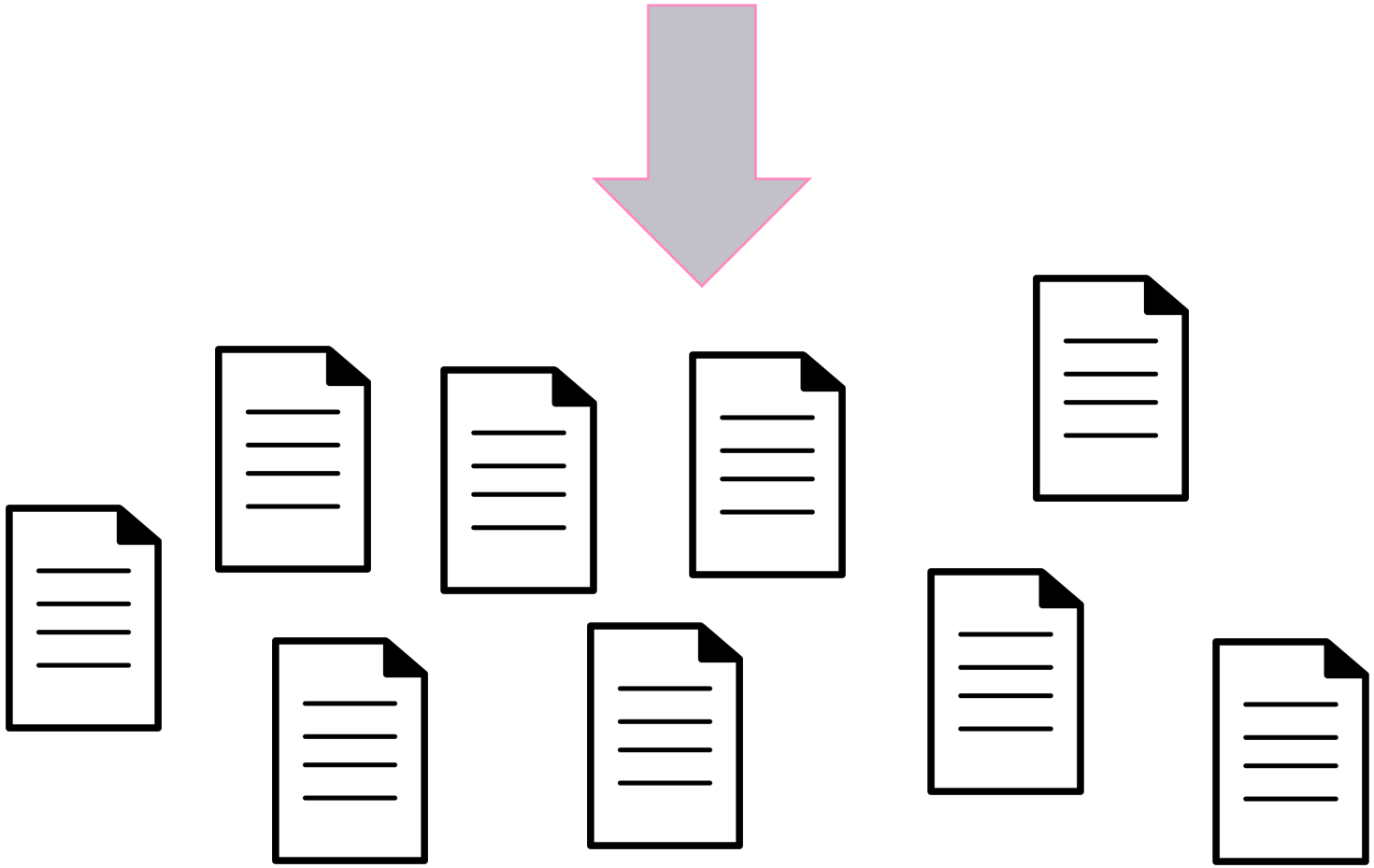
Information retrieval. Learning 2 rank.

Сбертех, МФТИ

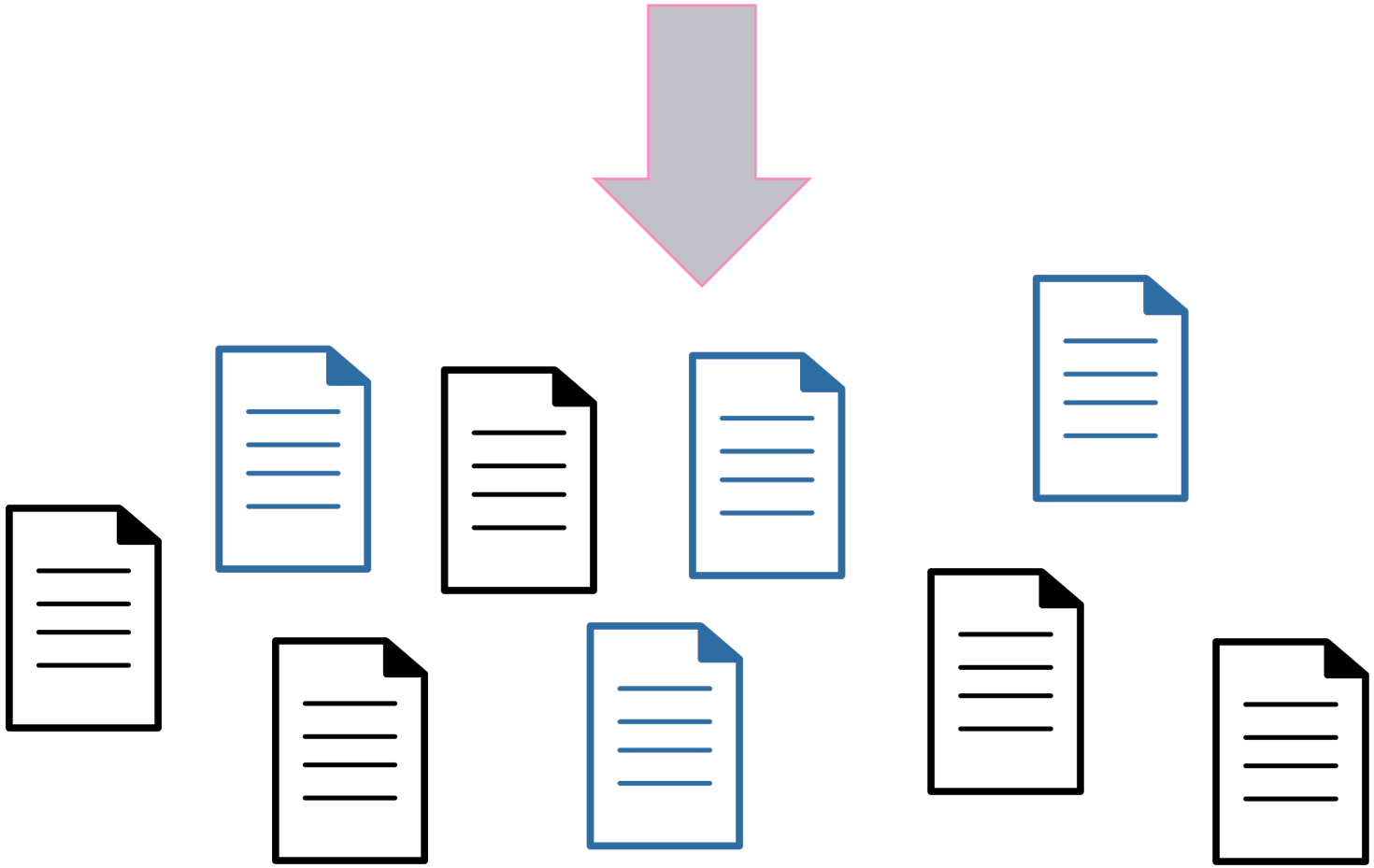
Задача информационного поиска

Кто снял фильм Титаник?



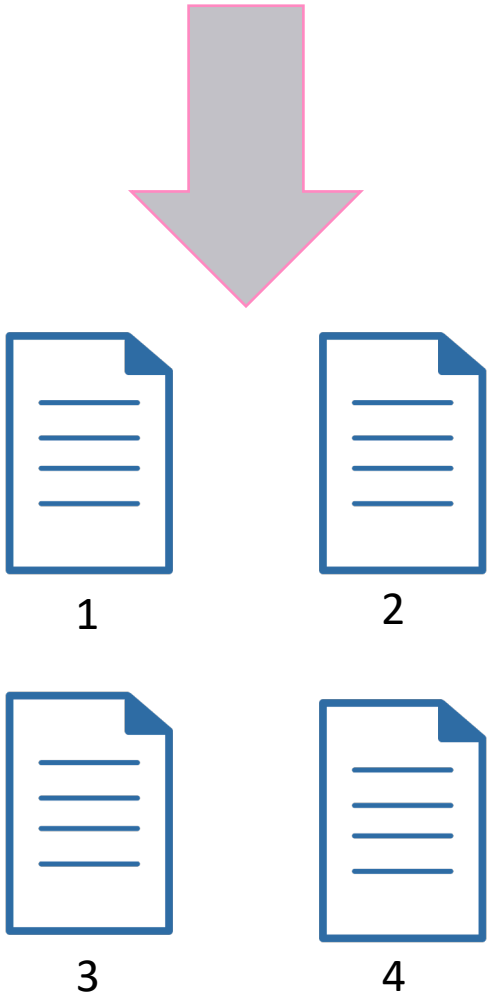
Задача информационного поиска – Поиск(retrieval)

Кто снял фильм Титаник?



Задача информационного поиска – Ранжирование(ranking)

Кто снял фильм Титаник?



Постановка задачи ранжирования

Пускай существует множество запросов $q_1, q_2, \dots, q_n$, где для каждого запроса существует m документов $d_1, d_2, \dots, d_m$. Каждому документу проставлена некоторая оценка из множества $\{S_1, S_2, \dots, S_k\}$, которая задает порядок выдачи документов для запроса q_i так что:

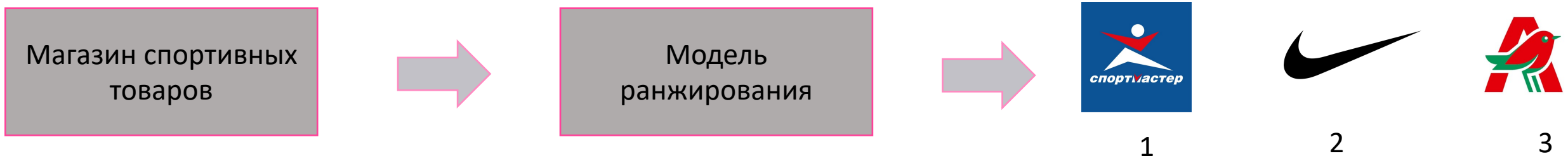
$$(q_i, d_1) \succ (q_i, d_2) \succ \cdots \succ (q_i, d_m)$$

Выражение $x \succ x'$ соответствует предпочтению объекта x объекту x'.

Задача ранжирования – построить такую функцию $a(q, \{d_1, d_2, \dots, d_m\})$, которая для каждой пары запрос-документ формирует оценку S, которая позволит упорядочить документы

$$S_1 > S_2 > \cdots > S_n \rightarrow (q, d_1) \succ (q, d_2) \succ \cdots \succ (q, d_m)$$

для запроса в последовательности, удовлетворяющей бизнес логике. Обычно данные для обучения ранжированию готовят асессоры.



Качество поиска

Метрика оценивания

Постановка задачи

Алгоритм ранжирования (pointwise, pairwise, listwise)

Данные

Признаки ранжирования

Признаки ранжирования

- **Текстовые**
  - BM25, TF-IDF, fuzzy…
- **Графовые**
  - С какими сущностями системы связан запрос?
- **Поведенческие**
  - Информация об истории взаимодействия пользователя с системой
- **Социальные**
  - Как пользователь связан с другими пользователями системы?
- **Временные**
  - В каком временном периоде живет запрос/документ?

Метрики ранжирования

Метрики оценивания ранжирования – сравнение предсказанного упорядоченного списка документов и истинного.

Точность ранжирования – если целевая переменная $y(q,d) \in \{0,1\}$.

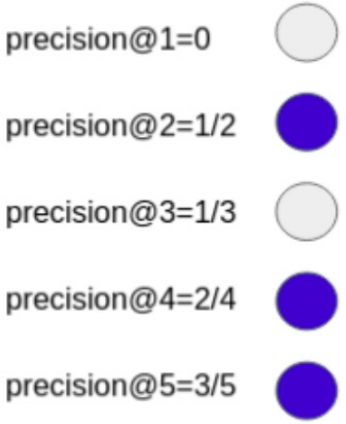
$\boldsymbol{Precision@k(q)}$ - доля релевантных документов среди отобранных k кандидатов

$$Precision@k(q) = \frac{1}{k}\sum_{i=1}^{k} y(q, d_q^i)$$

- Не учитываются позиции релевантных документов. Если у нас 10 кандидатов, то 5 наиболее релевантных документов могут находится где угодно среди этих кандидатов.

$\boldsymbol{AP@k(q)}$ - достигает максимума, если все релевантные документы находятся вверху ранжированного списка. Значение метрики уменьшается, если релевантные документы смещаются ниже.

$$AP@k(q) = \frac{\sum_{i=1}^{k} y\left(q, d_q^i\right) Precision@i(q)}{\sum_{i=1}^{k} y(q, d_q^i)}$$



Если необходимо вычислить $AP@k(q)$ для всей выборки, можно вычислить AP для каждого запроса и потом усреднить.

$$MAP@k = \frac{1}{|Q|}\sum_{q \in Q} AP@k(q)$$

https://medium.com/swlh/rank-aware-recsys-evaluation-metrics-5191bba16832

Метрики ранжирования

Если $y(q,d) \in R$ – метрика ранжирования дисконтированного совокупного прироста.

$$DCG(k) = \sum_{j:\pi_i(j) \le k} G(j)D(\pi_i(j))$$

DCG – дисконтируемый совокупный прирост информации от документа первого документа до k документа с учетом позиций документа.

$G(\cdot)$ – функция усиления(как мы оцениваем текущую оценку документа)

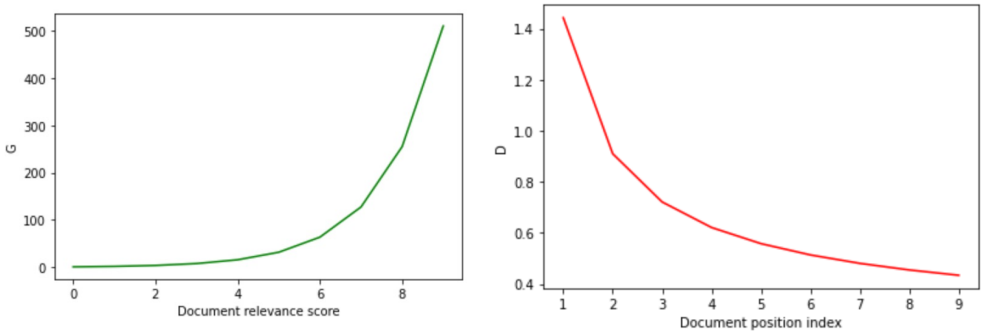
- $G(i) = 2^{y(q,d_i)} - 1$
- Чем больше релевантность документа, тем больше G

$D(\cdot)$- функция позиции(штраф за позицию)

- $D(\pi_i) = \frac{1}{\log_2(1+\pi_i(j))}$
- Чем ниже позиция документа, тем выше штраф
- $\pi_i(j)$ - j позиция документа d_j для запроса q_i в списке кандидатов π_i

k– количество документов относительно которых производится расчёт.

Сложно оптимизировать напрямую так как учитывается порядок документа.



- *Релевантный документ с низкой позицией – большой штраф*
- *Релевантный документ с высокой позицией – низкий штраф*
- *Если вверху списка находятся все релевантные документы – метрика достигает максимума*

Для нормировки метрики показатель DCG нормируют на DCG при идеальном ранжировании:

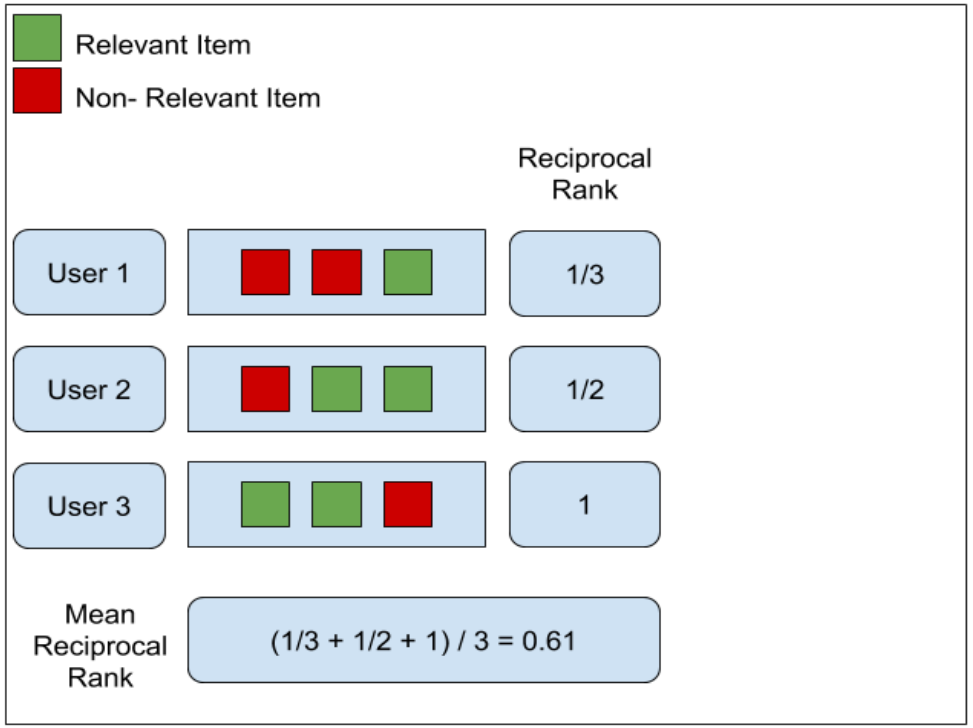
$$NDCG = \frac{DCG(k)}{\max(DCG(k))} \in [0;1]$$

Метрики ранжирования

Mean Reciprocal Rank(MRR) – метрика, оценивающая местоположение первого релевантного элемента относительно 1ого.

$$r_q = \frac{1}{p_{q_i}}$$

$$MRR@k = \frac{1}{|Q|}\sum_{q \in Q} r_q$$



Pointwise

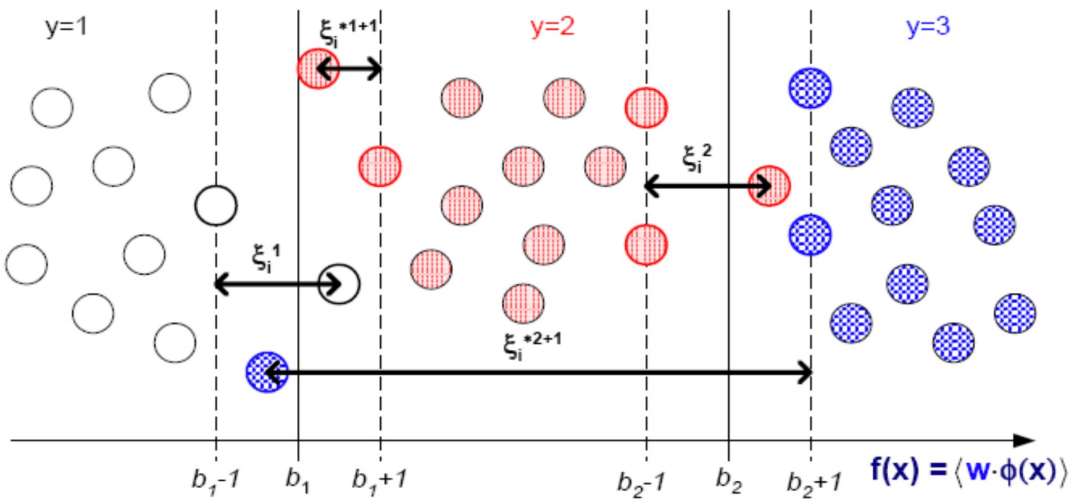
Минимизация ошибки между оценкой модели и оценкой асессора:

- Регрессия

$$L(a) = \sum_{q} \sum_{(q,d_i)} \big(y(q,d_i) - a(q,d_i)\big)^2 \to min$$

- Классификация

$$L(a) = \sum_{q} \sum_{(q,d_i)} [y(q,d_i) \neq a(q,d_i)] \to min$$



Плюсы

+ Прост в реализации
+ Уверенно работает на простых запросах

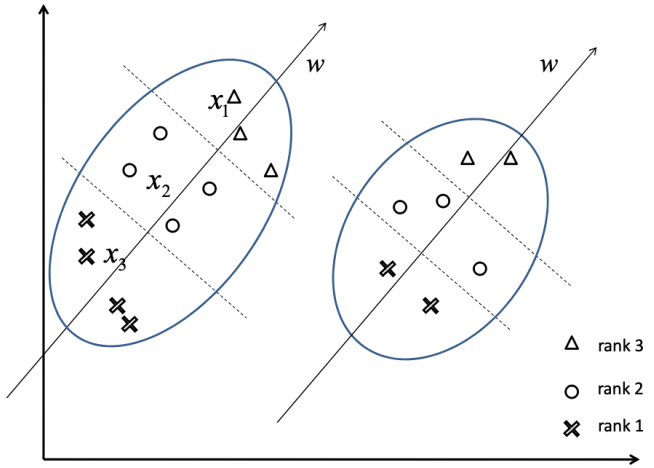
Минусы

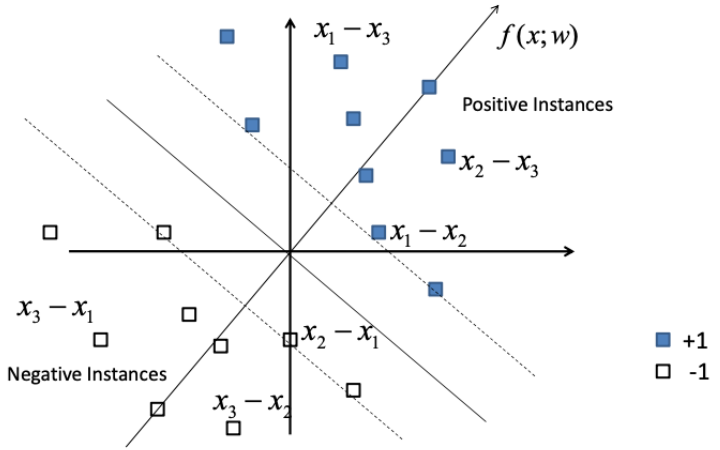
- Нет оптимизации порядка документов
- Появляется разная уверенность на разных положительных документах
- Если на 1 запрос 5 документов, на другой 10 – то модель склонна в будущем давать большую уверенность второму запросу

Pairwise (RankSVM)

Задача – научить модель правильно упорядочивать пары документов(минимизировать количество неправильно упорядоченных пар). Если $(q, d_j) \succ (q, d_i)$, то $h(x_j) - h(x_i) > 0$.

$$L(h) = \sum_h \sum_{(q,d_i)\prec(q,d_j)} [h(x_j) - h(x_i) < 0] \leq \sum_h \sum_{(q,d_i)\prec(q,d_j)} L\left(h(x_j) - h(x_i)\right) \to min$$





$$\begin{cases} \frac{1}{2}||w||^2 + C\sum_{i=1}^{m}\xi_i \to \min\limits_{w,b} \\ y_i(< w, x_i > +b) \geq 1 - \xi_i, i = 1, \dots, m \\ \xi_i \geq 0, i = 1, \dots, m \end{cases}$$

$$\min_w \sum_{i=1}^{m} \max\left(1 - y_i\langle w, x_i^1 - x_i^2\rangle, 0\right) + \lambda||w||^2$$

Pairwise (IR SVM)

Оценки релевантности – 3, 2, 1(чем выше, тем лучше)

Проблема № 1

РанжированиеNo1 – 2, 3, 2, 1, 1, 1,1
РанжированиеNo2 – 3, 2, 1, 2, 1, 1, 1

Где серьезнее ошибка?

Проблема №2

Выдача для запроса 1 – 3, 2, 2, 1, 1, 1, 1
Выдача для запроса 2 – 3, 3, 2, 2, 2, 1, 1, 1, 1, 1

Запрос 1

- 2 пары запросов для оценок 3-2
- 4 пары запросов для оценок 3-1
- 8 пар запросов для оценок 2-1

Итого – 14 пар документов

Запрос 2

- 6 пары запросов для оценок 3-2
- 10 пары запросов для оценок 3-1
- 15 пар запросов для оценок 2-1

Итого – 31 пар документов

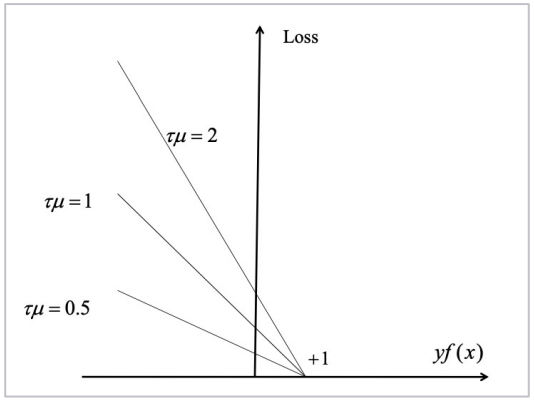
Ошибка на запросе 1 < Ошибка на запросе 2

$$\min_w \sum_{i=1}^{m} \tau_{k(i)}\mu_{q(i)}\max\left(1 - y_i\langle w, x_i^1 - x_i^2\rangle, 0\right) + \lambda\left||w|\right|^2$$

$\tau_{k(i)}$ - вес пары при оценке ранжирования $k(i)$ (чем выше оценка у пары, тем выше ошибка)

- *Рассчитывается, как изменение в NDCG при перестановке пары в списке ранжирования*

$\mu_{q(i)}$ - нормирование значение ошибки по кол-ву пар документов на запрос($= \frac{1}{|n_q|}$)



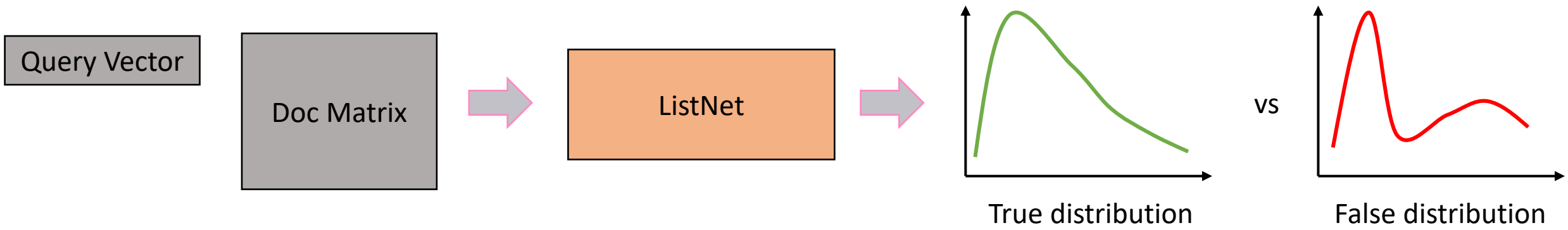
Модифицированный Hinge Loss

Listwise (ListNet)

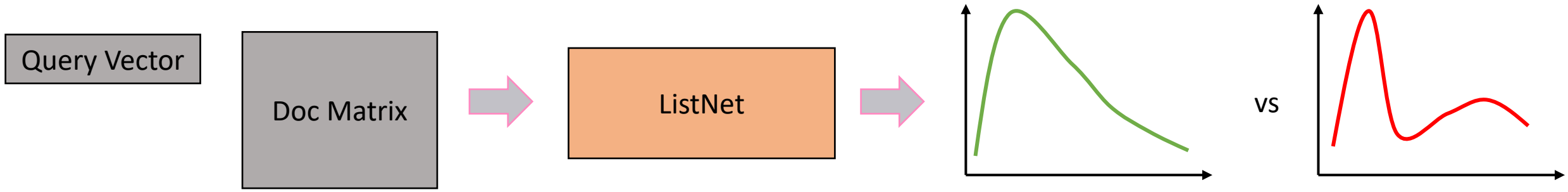
1) Вектор признаков $x_j^i = \phi(q^i, d_j^i)$ формируется для каждой пары запрос-документ. Набору векторов признаков $x^j = \{x_1^i, x_2^i, \dots, x_3^i\}$ запрос-документ соответствует список оценок $y_i == \{y_1^i, y_2^i, \dots, y\}$.

2) Необходимо построить такую функцию ранжирования f, так что документы d_j^i для запроса q^i будут упорядочены по построенным оценкам функцией ранжирования. Истинные оценки релевантности и оценки построенной функции порождают вероятностное распределение на множестве перестановок документов.

3) Задача модели на этапе обучения – минимизировать расстояние между распределением истинными оценками релевантности и предсказанным распределением оценок.



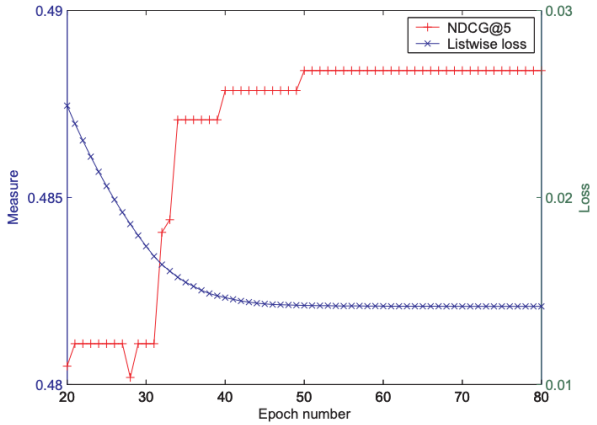
Listwise (ListNet)



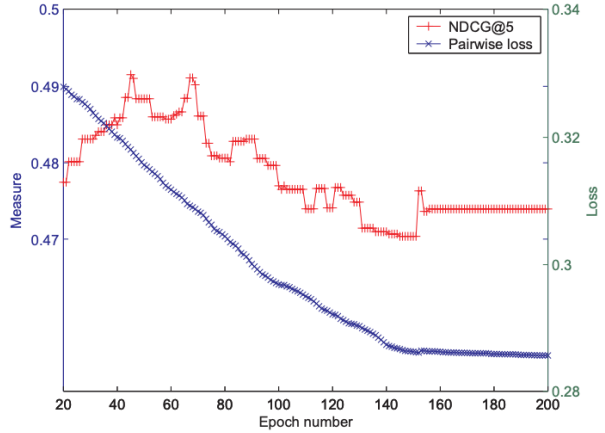
1. Готовим матрицу признаков Q и D
   - Признаки могут быть бинарными, категориальными, непрерывными
   - Документы отсортированы по оценке релевантности
2. В рамках модели конкатенируем признаки каждый документа с признаками запроса
3. Строим неглубокую модель(*качество ранжирования зависит от признаков*)
4. Обучаем модель на ошибку разницы между истинным и предсказанным распределением.
   - Истинное распределение считать, как softmax по оценкам релевантности документов
   - Предсказанное распределение считать как softmax по документам запроса
   - Функция ошибки – кросс-энтропия, KL

https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/tr-2007-40.pdf

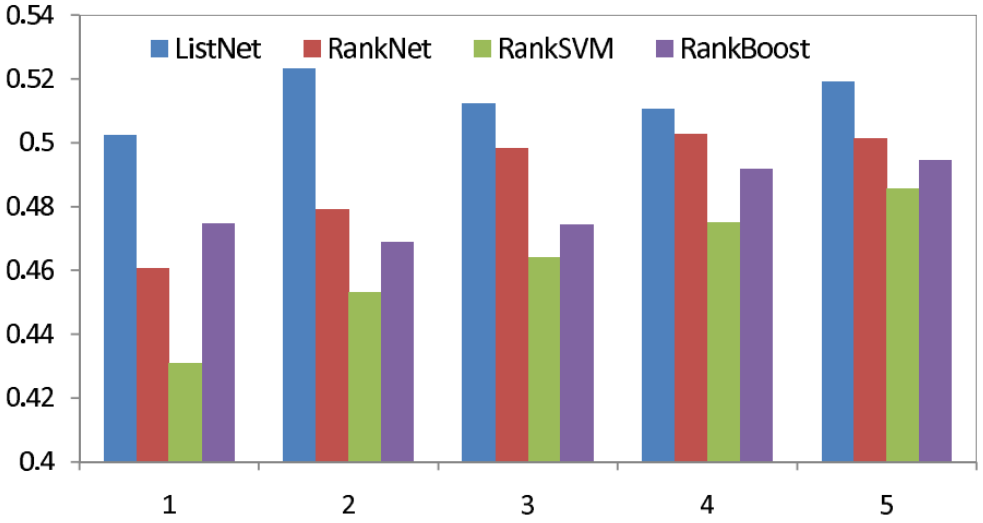
Метрики listwise модели



Listwise loss vs NDCG@5

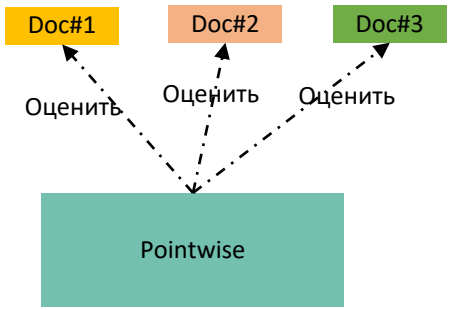


Pairwise loss vs NDCG@5

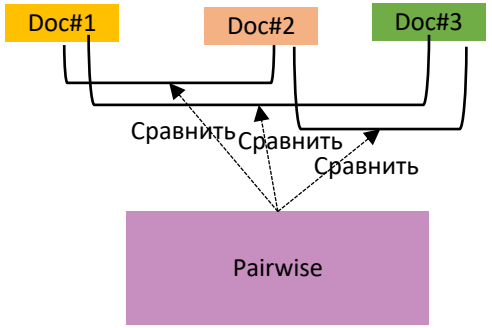


NDCG@N at TREC DATASET

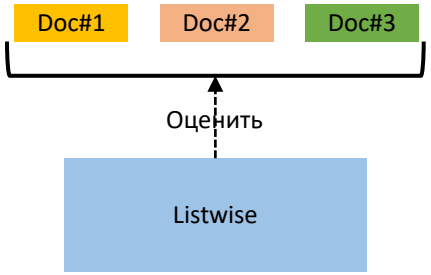
Какой алгоритм выбрать?



- **Pointwise** - предсказание релевантности запроса документу
  - Объем данных – $\mathrm{m} * N_{max}$ точек
  - Задача – предсказать, релевантен ли данный запрос данному документу
  - Если у нас много разных документов(положительных/отрицательных) на один запрос, то работает хорошо.
  - Где использовать – отделить релевантные документы от нерелевантных.
  - Нет возможности оптимизировать порядок выдачи



- **Pairwise** – предсказание релевантности одной пары над другой
  - Объем данных – $\mathrm{m} * N^2$ точек
  - Задача – предсказание порядка выдачи с помощью попарного предсказания релевантности документов
  - Если много оцененных документов на запрос – то работает хорошо, но медленно
  - Где использовать – упорядочить документы для выдачи пользователю



- **Listwise** – наиболее прямая оптимизация метрики NDCG
  - Объем данных – $\mathrm{m} * N_{\max}$ точек
  - Задача – минимизировать разницу между распределением оценки асессоров и предсказанным распределением
  - Если есть много запросов – то работает хорошо, но быстро
  - Где использовать - упорядочить документы для выдачи пользователю

Датасеты

- https://www.microsoft.com/en-us/research/project/mslr/
- https://microsoft.github.io/msmarco/
- https://www.microsoft.com/en-us/research/project/letor-learning-rank-information-retrieval/
- https://www.kaggle.com/c/yandex-personalized-web-search-challenge/data
- https://trec.nist.gov/data/session2013.html
- http://proceedings.mlr.press/v14/chapelle11a/chapelle11a.pdf - Yahoo Learning 2 rank challenge