Информационный поиск и ранжирование
Качество поиска

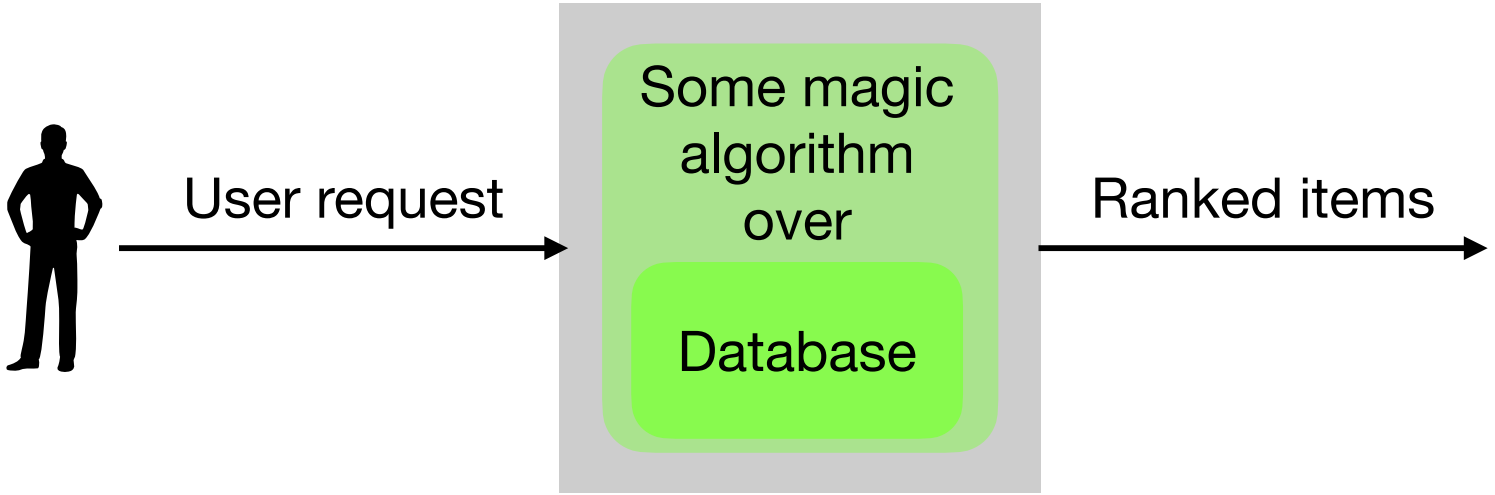
Андросов Дмитрий, 03.03.2025, AI Masters

План лекции

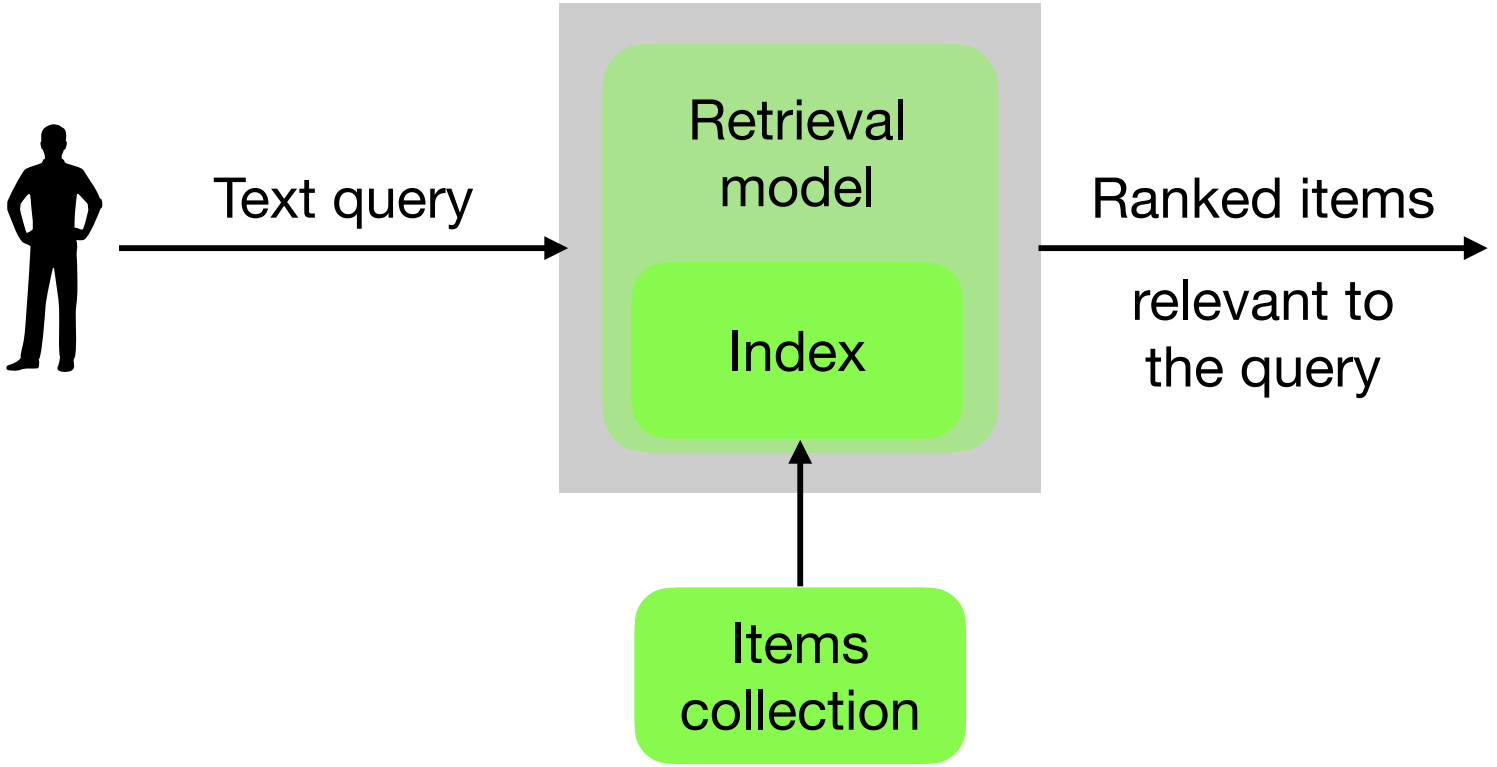
- Вспоминаем основы
- Введение в ранжирование
- Векторное представление текста
- Оценка качества поиска

Вспоминаем

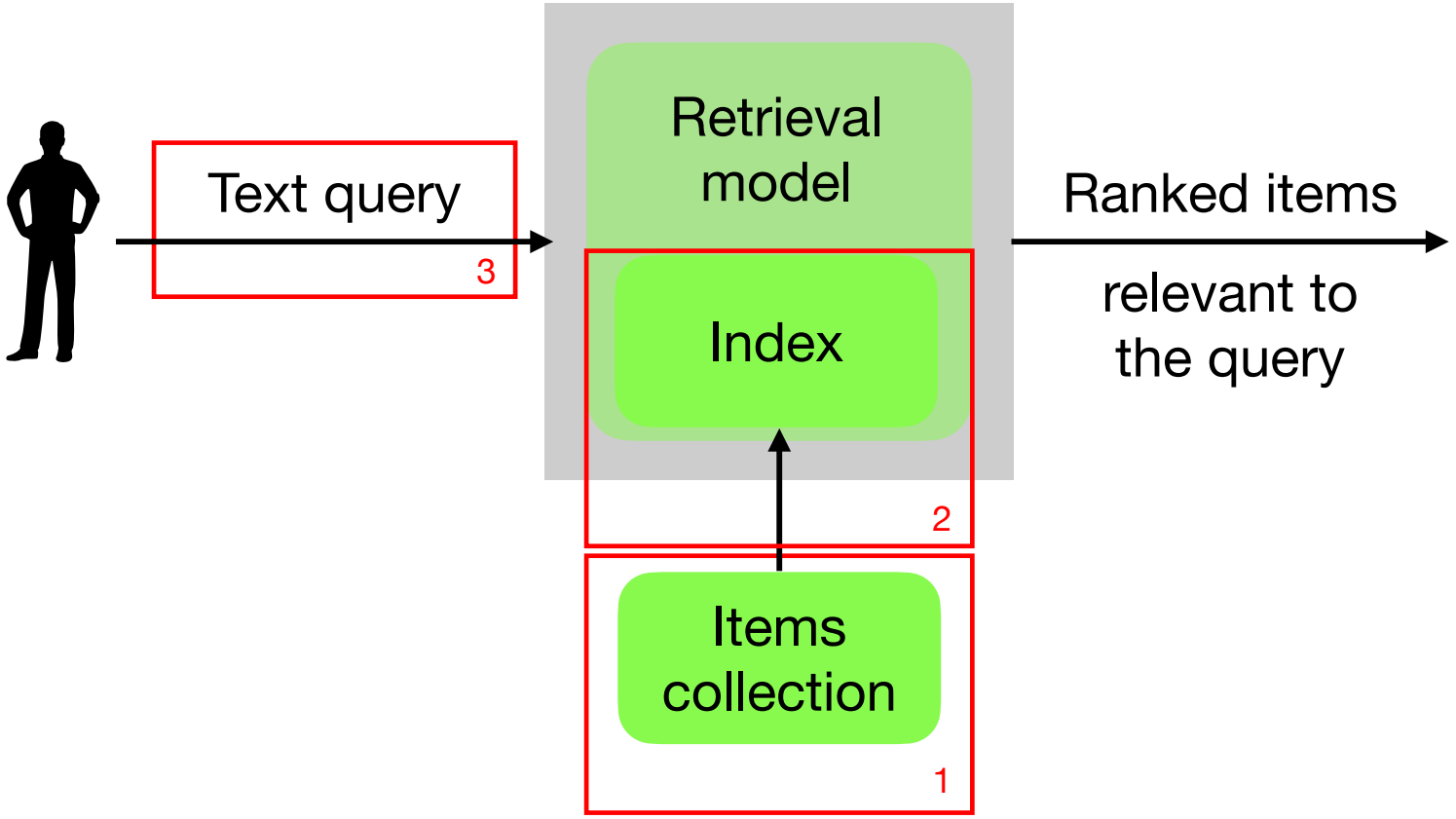
Ранжирующая система



Поисковая система



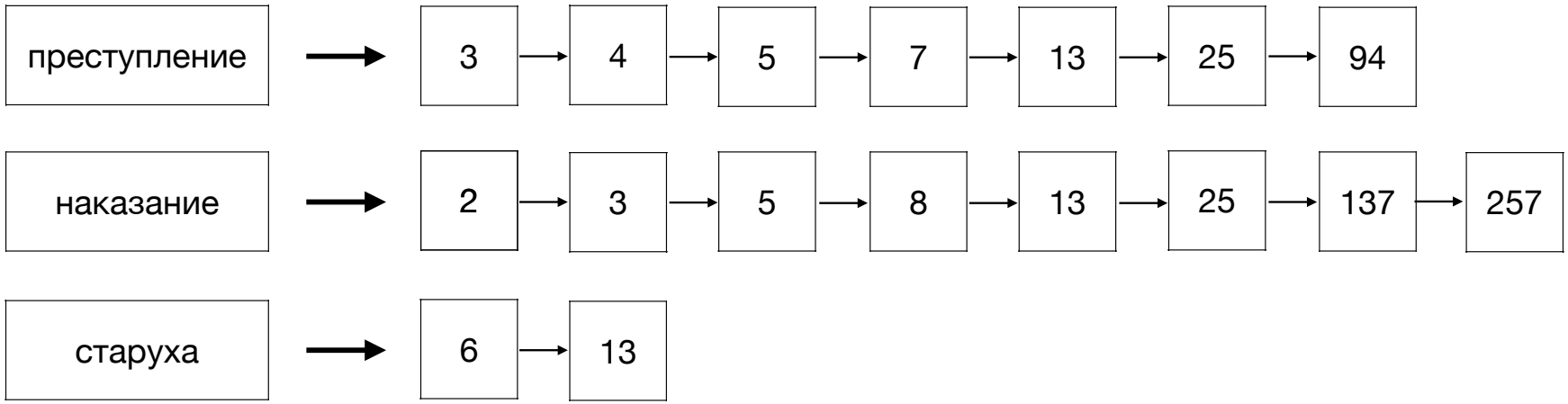
Поисковая система



Задача информационного поиска

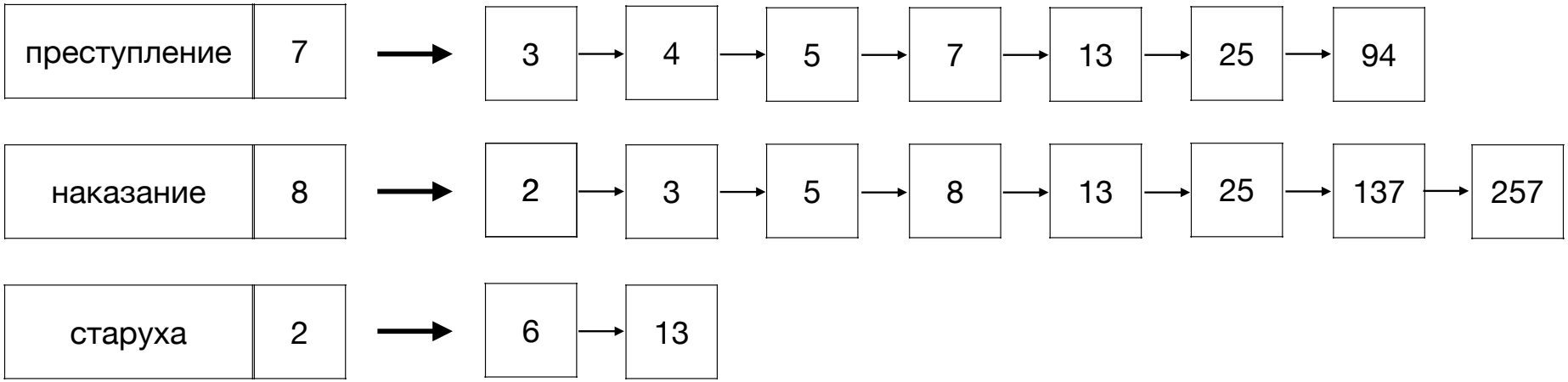
- Дано:
  - Набор документов $D = \{d_1, d_2, \ldots, d_n\}$
  - Запрос $Q = \{q_1, q_2, \ldots, q_m\}$
- Найти:
  - Набор документов $D* \subset D$, релевантных запросу Q

Inverted Index



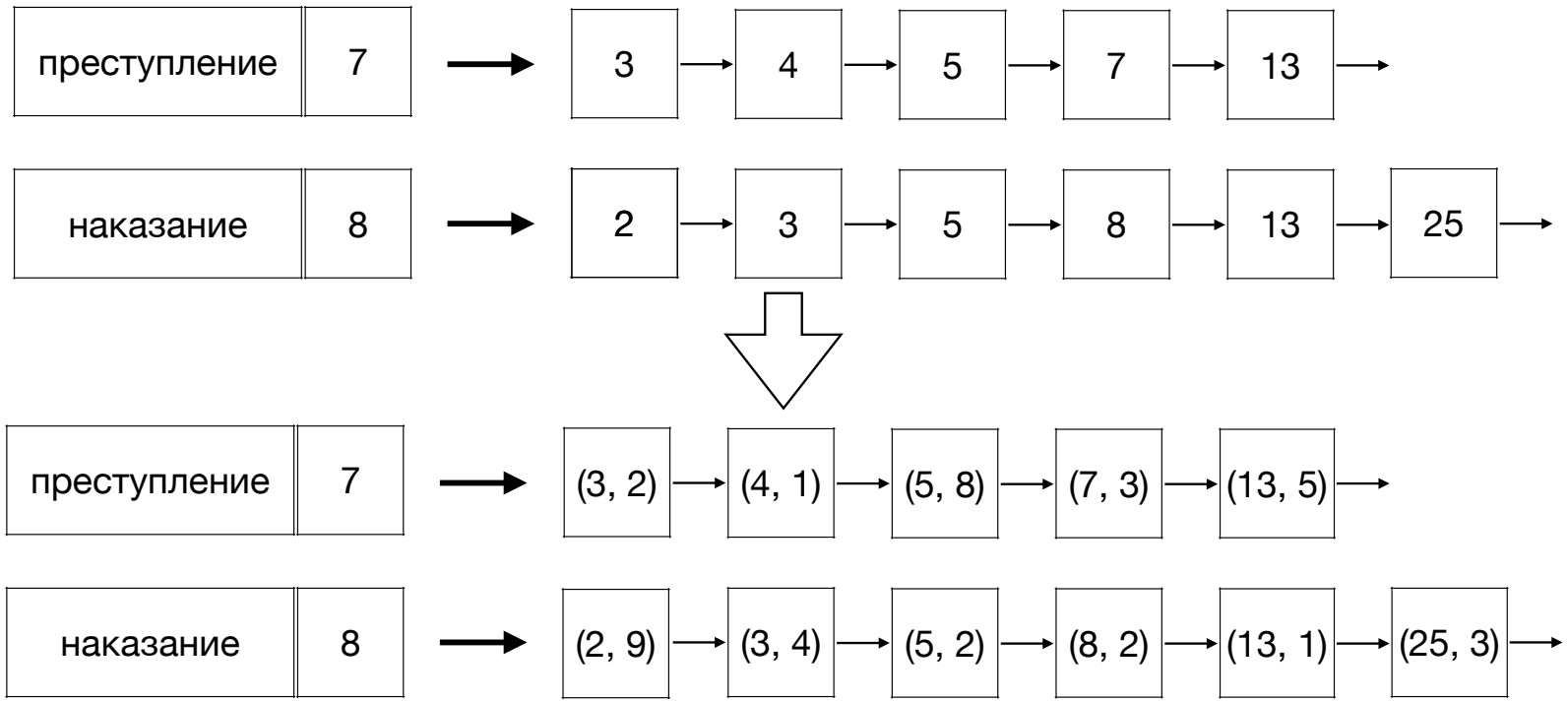
Inverted Index

with document frequency

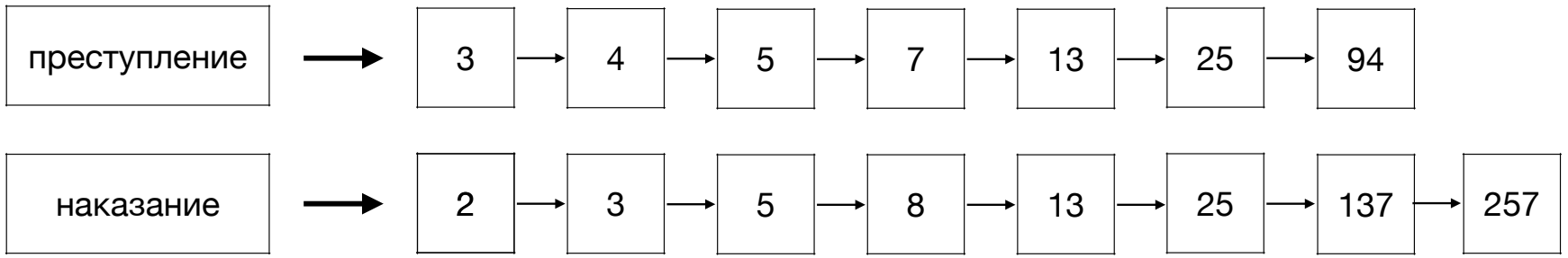


Inverted Index

with document frequency and term frequency



Boolean Retrieval Model



Query: «преступление AND наказание»

Result:

3	5	13	25

Введение в ранжирование

Boolean Retrieval Model

- Хорошо работает для пользователей-«экспертов» — тех кто точно понимает свои потребности, знает, как их сформулировать в виде запроса, хорошо знаком с коллекцией документов
- Не очень хорошо работает для большинства пользователей:
  - Сложно написать / сформулировать правильный запрос
  - Пользователям не нужны тысячи документов в ответе системы

Ranked Boolean Retrieval Model

2 этапа:

- Получение документов, удовлетворяющих ограничениям (запросу)
- Сортировка (ранжирование) полученных документов на основе величины, определяющей степень соответствия документа запросу

Ranked Boolean Retrieval Model

2 этапа:

- Получение документов, удовлетворяющих ограничениям (запросу)
- Сортировка (ранжирование) полученных документов на основе величины, определяющей степень соответствия документа запросу
- Но это не ранжирование по релевантности! Это ранжирование по соответствию условиям-ограничениям

Boolean Retrieval Model

Advantages and disadvantages

- **Простота** с точки зрения работы поисковой системы
- **Интерпретируемость**: легко понять почему документ был получен / не получен
- **Контролируемость**: легко определить полноту выдачи — мало документов (AND) или много документов (OR)

Boolean Retrieval Model

Advantages and disadvantages

- **Простота** с точки зрения работы поисковой системы
- **Интерпретируемость**: легко понять почему документ был получен / не получен
- **Контролируемость**: легко определить полноту выдачи — мало документов (AND) или много документов (OR)
- Ответственность за качество и эффективность лежит **на пользователе**
- Не поддерживается ранжирование документов **по релевантности**

Понятие релевантности

- На соответствие документа информационным потребностям пользователя влияет множество факторов: актуальность, свежесть, авторство, оформление, сложность, новизна и т.д.
- Релевантность теме: документ относится к той же тематике, что и запрос
- Релевантность пользователю: прочие факторы

Понятие релевантности

- На соответствие документа информационным потребностям пользователя влияет множество факторов: актуальность, свежесть, авторство, оформление, сложность, новизна и т.д.
- Релевантность теме: документ относится к той же тематике, что и запрос
- Релевантность пользователю: прочие факторы
- Цель: предсказать релевантность теме (topical relevance)

Понятие релевантности

- **Best-match retrieval model** — модель, предсказывающая степень релевантности документа запросу
- Наилучшая модель: $relevance(Q, d)$

Понятие релевантности

- **Best-match retrieval model** — модель, предсказывающая степень релевантности документа запросу
- Наилучшая модель: $relevance(Q, d)$
- На практике: $similarity(Q, d)$

Понятие релевантности

- **Best-match retrieval model** — модель, предсказывающая степень релевантности документа запросу
- Наилучшая модель: $relevance(Q, d)$
- На практике: $similarity(Q, d)$
- Основываясь на функции $similarity(Q, d)$ можно составить **отранжированный список (ranked list)** документов:
    - Некоторые документы более релевантны запросу, некоторые документы менее релевантны запросу

Понятие релевантности

- **Best-match retrieval model** — модель, предсказывающая степень релевантности документа запросу
- Наилучшая модель: $relevance(Q, d)$
- На практике: $similarity(Q, d)$
- Основываясь на функции $similarity(Q, d)$ можно составить **отранжированный список (ranked list)** документов:
    - Некоторые документы более релевантны запросу, некоторые документы менее релевантны запросу

Ranked retrieval

Ranked retrieval

- Запросы: текст (естественный язык)
- Документы ранжируются по релевантности / близости запросу
- Результаты обработки запроса контролируемы по объему:
    - Отдаем десятки / сотни результатов

Ranked retrieval

- Запросы: текст (естественный язык)
- Документы ранжируются по релевантности / близости запросу
- Результаты обработки запроса контролируемы по объему:
    - Отдаем десятки / сотни результатов
- Предположения:
    - Топ отранжированных документов более вероятно удовлетворяют пользовательскому запросу
    - Скор документа основывается на его релевантности запросу
- Главный вопрос в определении $score(d, Q) = f(similarity(d, Q))$

Ranked retrieval

Мера (коэффициент) Жаккара

- Определяет пересечение 2 наборов токенов A и B:

$$Jaccard(A, B) = \frac{A \cap B}{A \cup B}$$

- Документы и запрос представляются как мешок слов (BoW)
- $score(d, Q) = Jaccard(d, Q)$

Ranked retrieval

Мера (коэффициент) Жаккара

- Определяет пересечение 2 наборов токенов A и B:

$$Jaccard(A, B) = \frac{A \cap B}{A \cup B}$$

- Пример:
  - d = «Раскольников совершил преступление»
  - Q = «Преступление и наказание»
  - $score(d, Q) = Jaccard(d, Q) = \frac{1}{5} = 0.2$

Ranked retrieval

Мера (коэффициент) Жаккара — недостатки

- Не учитывает частота встречаемости токена в документе (term frequency)

Ranked retrieval

Мера (коэффициент) Жаккара — недостатки

- Не учитывает частота встречаемости токена в документе (term frequency)
- Все токены считаются одинаково важными:
    - Нет понятия важности токена

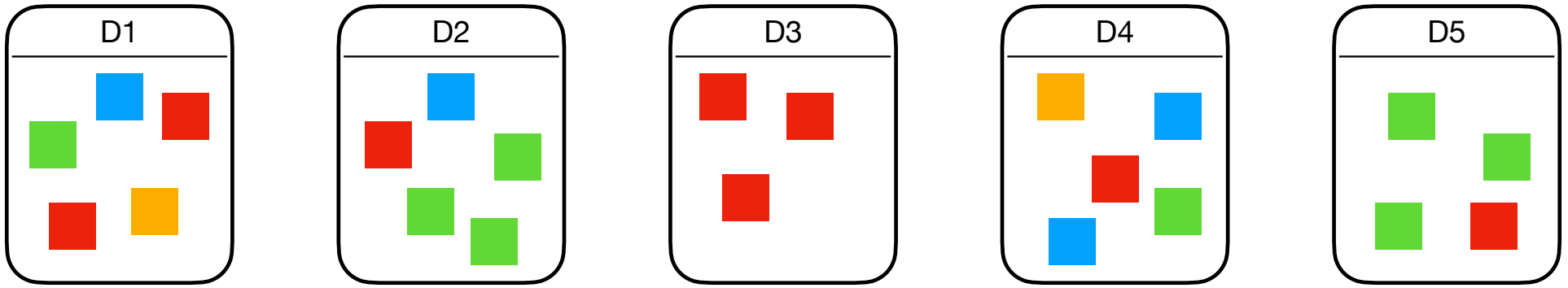
Ranked retrieval

Мера (коэффициент) Жаккара — недостатки

- Не учитывает частота встречаемости токена в документе (term frequency)
- Все токены считаются одинаково важными:
  - Нет понятия важности токена
- Нет нормализации на длину документа:
  - $|d_1| = 5, |d_2| = 100, |Q| = 3,\ d_1 \subset d_2$
  - $score(d_1, Q) >> score(d_2, Q)$

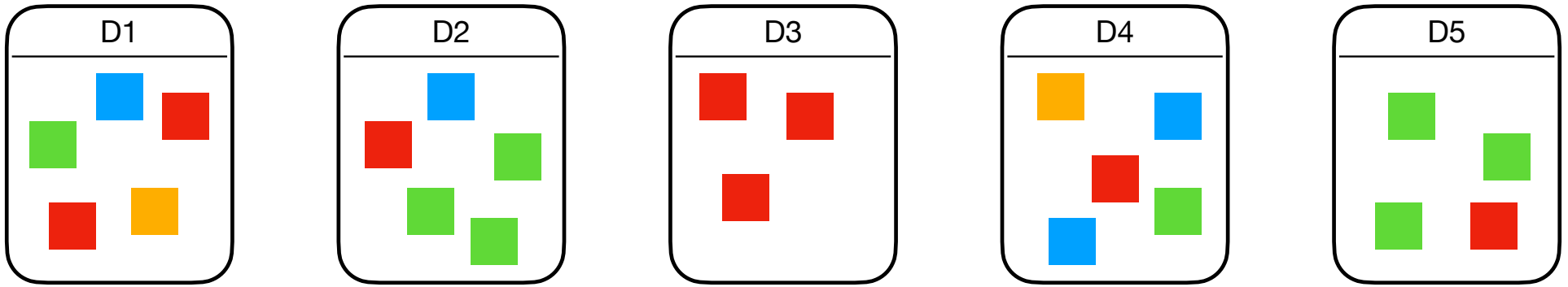
Важность токенов

Почему нужно считать важность?



- Запрос = 
- Какие более и какие менее релевантные документы?

Почему нужно считать важность?



- Запрос = 
- Какие более и какие менее релевантные документы?

Нужна мера важности токена

Document scoring

$$score(d, Q) = \begin{cases} \sum_{t \in Q} f(d, t) \\ \prod_{t \in Q} g(d, t) \\ \dots \end{cases}$$

Document scoring

$$score(d, Q) = \begin{cases} \sum_{t \in Q} f(d, t) \\ \prod_{t \in Q} g(d, t) \\ \dots \end{cases}$$

- Запрос: «преступление и наказание»
- $score(d, Q) = f(d, \text{«преступление»}) + f(d, \text{«и»}) + f(d, \text{«наказание»})$

Document scoring

$$score(d, Q) = \begin{cases} \sum_{t \in Q} f(d, t) \\ \prod_{t \in Q} g(d, t) \\ \dots \end{cases}$$

- Запрос: «преступление и наказание»
- $score(d, Q) = f(d, \text{«преступление»}) + f(d, \text{«и»}) + f(d, \text{«наказание»})$
- $f(\cdot), g(\cdot)$ — определяют важность токенов

Важность токенов

Term frequency (tf)

- $tf(t, d)$ терма t в документе d определяется как количество раз, которое терм встречается в документе
- Можно использовать tf для определения важности документа для запроса

Важность токенов

Term frequency (tf)

- $tf(t, d)$ терма t в документе d определяется как количество раз, которое терм встречается в документе
- Можно использовать tf для определения важности документа для запроса
- Допустим $tf(t, d_1) = 10,\ tf(t, d_2) = 100$. Значит ли это, что $relevance(d_2) = 10 \cdot relevance(d_1)$?

Важность токенов

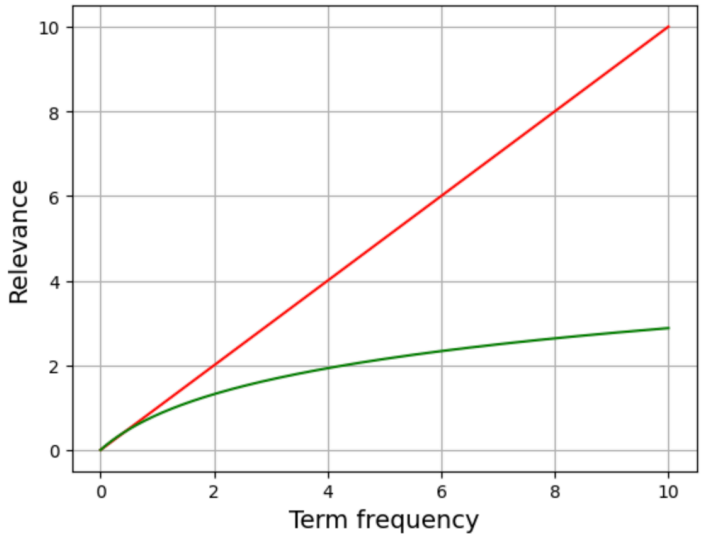
Term frequency (tf)

- $tf(t, d)$ терма t в документе d определяется как количество раз, которое терм встречается в документе
- Можно использовать tf для определения важности документа для запроса
- Допустим $tf(t, d_1) = 10,\ tf(t, d_2) = 100$. Значит ли это, что $relevance(d_2) = 10 \cdot relevance(d_1)$? Нет! Но значит, что $relevance(d_2) > relevance(d_1)$

Важность токенов

Term frequency (tf)

- Релевантность зависит от tf нелинейно
- Логично предположить, что рост релевантности замедляется при достаточно больших значениях tf :



Важность токенов

Term frequency (tf) — logarithm scaling

- Эвристика:

$$w(t,d) = \begin{cases} 1 + log_{10}\ tf(t,d) \\ 0,\ otherwise \end{cases}$$

Важность токенов

Term frequency (tf) — logarithm scaling

- Эвристика:

$$w(t,d) = \begin{cases} 1 + log_{10}\ tf(t,d) \\ 0,\ otherwise \end{cases}$$

- $tf(t,d) \implies w(t,d): \ 0 \implies 0; \ 1 \implies 1; \ 2 \implies 1.3; \ 10 \implies 2; \ 1000 \implies 4$

Важность токенов

Term frequency (tf) — logarithm scaling

- Эвристика:

$$w(t,d) = \begin{cases} 1 + log_{10}\ tf(t,d) \\ 0,\ otherwise \end{cases}$$

- $tf(t,d) \implies w(t,d): \ 0 \implies 0; \ 1 \implies 1; \ 2 \implies 1.3; \ 10 \implies 2; \ 1000 \implies 4$
- Скор пары запрос-документ — сумма по всем термам запроса:

$$scaled\_tf\_score(Q,d) = \sum_{t \in Q} (1 + log_{10}\ tf(t,d))$$

- Если ни один терм $t \in Q$ не присутствует в документе d, то $scaled\_tf\_score(Q,d) = 0$

Важность токенов

Редкие токены

- Редко встречающиеся токены более информативны, чем частые токены

Важность токенов

Редкие токены

- Редко встречающиеся токены более информативны, чем частые токены
- Пример: «Преступление и наказание»
- Документы содержащие термы «преступление», «наказание» более релевантны запросу, чем документы, содержащие «и»

Важность токенов

Редкие токены

- Редко встречающиеся токены более информативны, чем частые токены
- Пример: «Преступление и наказание»
- Документы содержащие термы «преступление», «наказание» более релевантны запросу, чем документы, содержащие «и»
- Ещё пример: «купить большой холодильник»

Важность токенов

Редкие токены

- Редко встречающиеся токены более информативны, чем частые токены
- Пример: «Преступление и наказание»
- Документы содержащие термы «преступление», «наказание» более релевантны запросу, чем документы, содержащие «и»
- Ещё пример: «купить большой холодильник»
- Почему бы тогда не давать больший вес редким токенам?

Важность токенов

Редкие токены

- Редко встречающиеся токены более информативны, чем частые токены
- Пример: «Преступление и наказание»
- Документы содержащие термы «преступление», «наказание» более релевантны запросу, чем документы, содержащие «и»
- Ещё пример: «купить большой холодильник»
- Почему бы тогда не давать больший вес редким токенам?

Нужна мера редкости токена

Важность токенов

- Мы хотим задавать больший вес редким токенам
- Мы хотим задавать меньший (но положительный) вес частым токенам, поскольку найти частый токен в документе всё ещё лучше, чем его не найти

Важность токенов

- Мы хотим задавать больший вес редким токенам
- Мы хотим задавать меньший (но положительный) вес частым токенам, поскольку найти частый токен в документе всё ещё лучше, чем его не найти
- Будем использовать *document frequency* как меру редкости токена
- *document frequency* ($df(t)$) токена t определяется как количество документов коллекции, в которых хотя бы раз встретился токен t

Важность токенов

Inverse document frequency (idf)

- $df(t)$ токена t определяется как количество документов коллекции, в которых хотя бы раз встретился токен t
- $df(t)$ является обратной мерой информативности токена t

Важность токенов

Inverse document frequency (idf)

- $df(t)$ токена t определяется как количество документов коллекции, в которых хотя бы раз встретился токен t
- $df(t)$ является обратной мерой информативности токена t
- $Inverse\ document\ frequency$ определяется как:

$$idf(t) = log_{10} \frac{N}{df(t)}$$

- где N — количество документов коллекции
- $idf(t)$ является мерой информативности токена t

Важность токенов

TF-IDF term weighting

- Важность тем больше, чем чаще токен встречается в документе — tf
- Важность тем больше, чем более редкий токен в коллекции — idf
- Комбинация tf и idf:

$$w(t, d) = (1 + log_{10}\ tf(t, d)) \cdot log_{10}\ \frac{N}{df(t)}$$

Vector Space Model

Incidence vectors

- Вектор представления токена (term incidence vector, TIV) — вектор-индикатор встречаемости данного токена в документах корпуса
- Вектор представления документа (document incidence vector, DIV) — вектор-индикатор встречаемости токенов словаря в данном документе

DIV(doc_2)

TIV(мама)

	мама мыла раму	мама мыла пол	деревянная рама
мама	1	1	0
мыть	1	1	0
рама	1	0	1
пол	0	1	0
деревянный	0	0	1

Incidence vectors

- Вектор представления токена (term incidence vector, TIV) — вектор-индикатор встречаемости данного токена в документах корпуса
- Вектор представления документа (document incidence vector, DIV) — вектор-индикатор встречаемости токенов словаря в данном документе

DIV(doc_2)

TIV(мама)

	мама мыла раму	мама мыла пол	деревянная рама
мама	1	1	0
мыть	1	1	0
рама	1	0	1
пол	0	1	0
деревянный	0	0	1

- Нет представления о частотности токена, его позиции и т.д.

Vector Space Model

- Представляем запрос и документ в виде векторов в едином пространстве
- Вычисляем близость между векторами

$$score(Q, d) = vector\_similarity(\vec{Q}, \vec{d})$$

- Ранжируем документы по близости вектора к вектору запроса

Близость векторов

Евклидово расстояние

Для данных $\vec{X} = \{x_1, x_2, \ldots, x_n\}$, $\vec{Y} = \{y_1, y_2, \ldots, y_n\}$ найти близость

- Евклидово расстояние (*Euclidean_distance*):

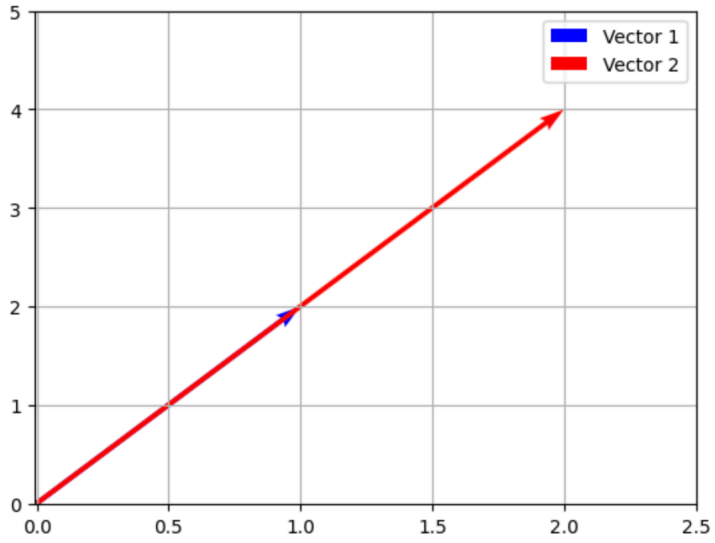
$$Euclidean\_distance(\vec{X}, \vec{Y}) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

- Вычисляется, насколько близки векторы в данном векторном пространстве

Близость векторов

Евклидово расстояние

- Схожесть документов обратно пропорциональна расстоянию
- Расстояние большое для векторов разных длин
  - Представим $\overrightarrow{d_2} = 2 \cdot \overrightarrow{d_1}$
  - Семантически $\overrightarrow{d_1}$ и $\overrightarrow{d_2}$ близки
  - Однако Евклидово расстояние велико



Близость векторов

Скалярное произведение

Для данных $\overrightarrow{X} = \{x_1, x_2, \ldots, x_n\}$, $\overrightarrow{Y} = \{y_1, y_2, \ldots, y_n\}$ найти близость

- Скалярное произведение ($dot\_product$):

$$vector\_similarity(\overrightarrow{X}, \overrightarrow{Y}) = \sum_{i=1}^{n} x_i \cdot y_i = X^T Y$$

- $dot\_product(\overrightarrow{X}, \overrightarrow{Y}) = 0 \implies \overrightarrow{X} \perp \overrightarrow{Y}$
- Если мы работаем с бинарным представлением (1 - если токен встретился, 0 - иначе), то скалярное произведение равно количеству токенов, встретившихся (хотя бы один раз) в X и в Y

Близость векторов

Скалярное произведение

- Какой документ более релевантен запросу $Q = \{q_1, q_2, q_3\}$:
  - Документ, состоящий из 20 токенов, среди которых есть 3 токена запроса?
  - Документ, состоящий из 200 токенов, среди которых есть 3 токена запроса?

Близость векторов

Скалярное произведение

- Какой документ более релевантен запросу $Q = \{q_1, q_2, q_3\}$:
  - Документ, состоящий из 20 токенов, среди которых есть 3 токена запроса?
  - Документ, состоящий из 200 токенов, среди которых есть 3 токена запроса?
- При прочих равных, длинные документы более вероятно содержат в себе токены запроса, чем более короткие документы
- Скалярное произведение не учитывает, что документы могут сильно отличаться по длине
- Таким образом, скалярное произведение склоняет к более длинным документам

Близость векторов

Нормализация на длину вектора

- Можно нормализовать вектор, поделив его компоненты на длину вектора — его L_2-норму:

$$\|\vec{x}\| = \sqrt{\sum_i x_i^2}$$

- Таким образом, мы приводим векторы к одинаковой длине
- Длинные и короткие документы теперь имеют сравнимые представления

Близость векторов

Косинусная близость

Для данных $\vec{X} = \{x_1, x_2, \ldots, x_n\}$, $\vec{Y} = \{y_1, y_2, \ldots, y_n\}$ найти близость

- Косинусная близость ($cosine\_similarity$):

$$cosine\_similarity(\vec{X}, \vec{Y}) = \frac{\sum_i x_i \cdot y_i}{\sqrt{\sum_i x_i^2} \cdot \sqrt{\sum_i y_i^2}}$$

- $cosine\_similarity \in [0, 1]$

Близость векторов запроса и документа

$$score(Q, d) = cosine\_similarity(\vec{Q}, \vec{d}) = \frac{\sum_{i=1}^{|vocab|} Q_i \cdot d_i}{\sqrt{\sum_i Q_i^2} \cdot \sqrt{\sum_i d_i^2}} = \sum_{t \in Q} w(t, Q) \cdot w(t, d)$$

- где $w(t, Q),\ w(t, d)$ — веса токена t в запросе Q и документе d

Взвешивание токенов

Подход 1 — бинарный

- $w(t, d) = \begin{cases} 1, \ if \ t \in d \\ 0, \ otherwise \end{cases}$
- Документ — бинарный вектор (множество токенов)
- Близость — пересечение токенов

$$score(Q, d) = \sum_t I_{t \in Q} \cdot I_{t \in d}$$

- где I — индикатор

Взвешивание токенов

Подход 2 — взвешенный

- $w(t, d) = tf(t, d)$:

$$score(Q, d) = \sum_{t} tf(t, Q) \cdot tf(t, d)$$

Взвешивание токенов

Подход 2 — взвешенный

- $w(t, d) = tf(t, d)$:

$$score(Q, d) = \sum_{t} tf(t, Q) \cdot tf(t, d)$$

- Проблема — длинные документы имеют преимущества над короткими, поскольку вероятно имеют больший $tf(t, d)$

Взвешивание токенов

Подход 3 — взвешенный, нормализованный на длину

- $w(t, d) = \frac{tf(t, d)}{|d|}$:

$$score(Q, d) = \sum_{t} tf(t, Q) \cdot \frac{tf(t, d)}{|d|}$$

Взвешивание токенов

Подход 3 — взвешенный, нормализованный на длину

- $w(t, d) = \frac{tf(t, d)}{|d|}$:

$$score(Q, d) = \sum_t tf(t, Q) \cdot \frac{tf(t, d)}{|d|}$$

- Почему не делается нормализация на длину запроса $\frac{tf(t, Q)}{|Q|}$?

Взвешивание токенов

Подход 3 — взвешенный, нормализованный на длину

- $w(t, d) = \frac{tf(t, d)}{|d|}$:

$$score(Q, d) = \sum_t tf(t, Q) \cdot \frac{tf(t, d)}{|d|}$$

- Почему не делается нормализация на длину запроса $\frac{tf(t, Q)}{|Q|}$?

- Скор будет отличаться, но ранжирование останется прежним, поскольку мы для всех пар запрос-документ поделим скор на одно и то же число

Взвешивание токенов

Подход 4 — редкие токены

- $w(t,d) = \frac{tf(t,d)}{|d|} \cdot log\ \frac{N}{df(t)}$:

$$score(Q,d) = \sum_t tf(t,Q) \cdot \frac{tf(t,d)}{|d|} \cdot log\ \frac{N}{df(t)}$$

- Больший вес получают редкие токены за счет IDF
- Проблема — линейная зависимость от частоты встречаемости (tf)

Взвешивание токенов

Подход 5 — TF logarithm scaling

- $w(t,d) = (1 + log\ \frac{tf(t,d)}{|d|}) \cdot log\ \frac{N}{df(t)}$:

$$score(Q,d) = \sum_{t} tf(t,Q) \cdot (1 + log\ \frac{tf(t,d)}{|d|}) \cdot log\ \frac{N}{df(t)}$$

Взвешивание токенов

Подход 5 — TF logarithm scaling

- $w(t,d) = (1 + log\ \frac{tf(t,d)}{|d|}) \cdot log\ \frac{N}{df(t)}$:

$$score(Q,d) = \sum_t \underbrace{tf(t,Q)}_{w(t,Q)} \cdot \underbrace{(1 + log\ \frac{tf(t,d)}{|d|}) \cdot log\ \frac{N}{df(t)}}_{w(t,d)}$$

BM25 Model (Best Matching 25)

BM25 Model

Вспоминаем TF-IDF

- $TF - IDF(t, d) = TF(t, d) \cdot IDF(t)$

BM25 Model

Вспоминаем TF-IDF

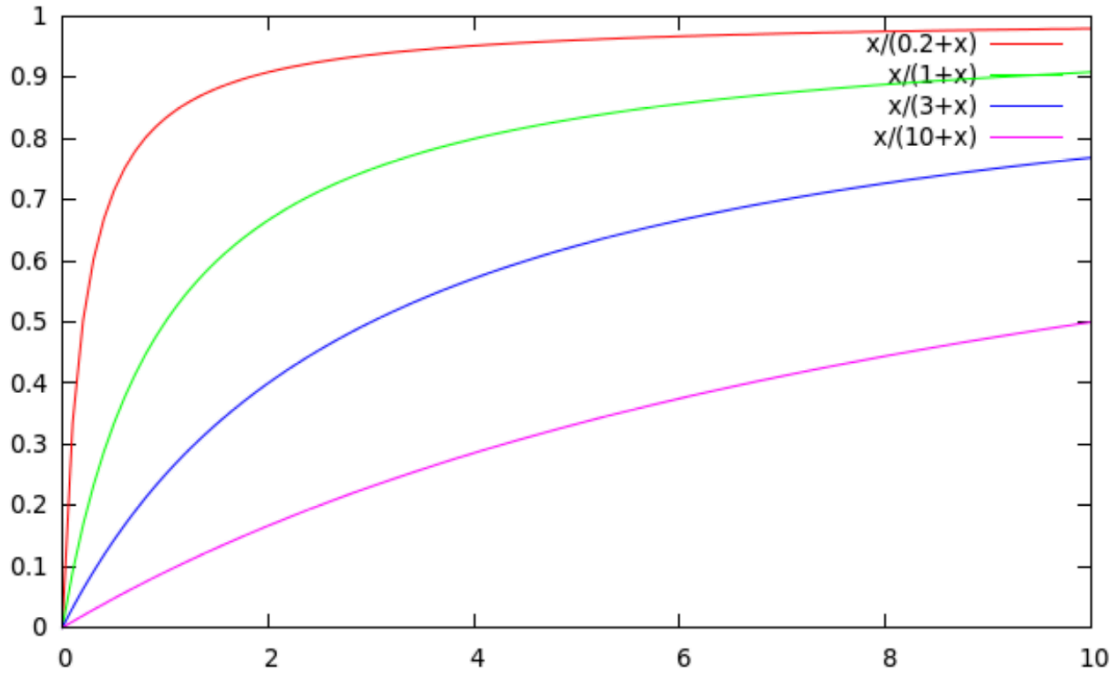
- $TF - IDF(t, d) = TF(t, d) \cdot IDF(t)$
- **Цель**: быть чувствительным к $TF(t, d)$ и длине документа, при этом не добавляя большого количества параметров в модель.

BM25 Model

Функция насыщаемости (saturation function)

- Функция насыщаемости:

$$TF(t,d) = \frac{tf(t,d)}{k_1 + tf(t,d)}$$



BM25 Model

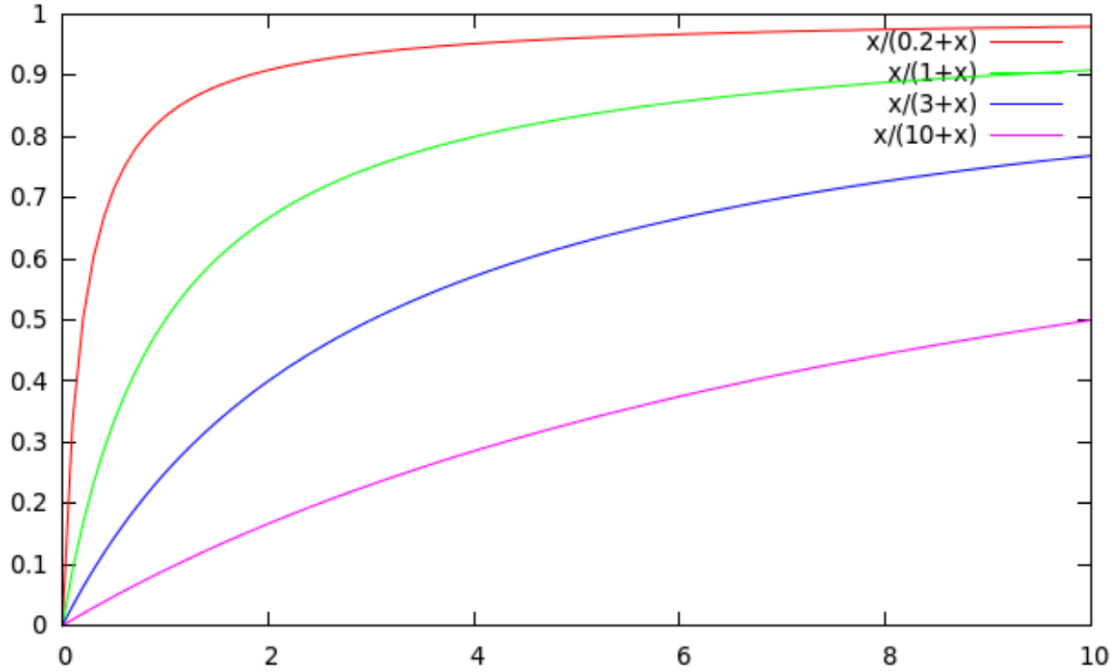
Функция насыщаемости (saturation function)

- Функция насыщаемости:

$$TF(t,d) = \frac{tf(t,d)}{k_1 + tf(t,d)}$$

- Дополнительно:

$$TF(t,d) = \frac{(k_1 + 1) \cdot tf(t,d)}{k_1 + tf(t,d)}$$



- $(k_1 + 1)$ не изменяет ранжирование, но выравнивает скор:
  при $tf(t,d) = 1 \implies TF(t,d) = 1$

BM25 Model

Нормализация на длину документа

- Длина документа:

$$document\_length = dl = \sum_{t \in d} tf(t, d)$$

- $avdl$ — средняя длина документов коллекции

BM25 Model

Нормализация на длину документа

- Длина документа:

$$document\_length = dl = \sum_{t \in d} tf(t, d)$$

- $avdl$ — средняя длина документов коллекции
- Компонента нормализации на длину:

$$B = \left(1 - b + b \cdot \frac{dl}{avdl}\right), \quad 0 \leq b \leq 1$$

BM25 Model

Нормализация на длину документа

- Длина документа:

$$document\_length = dl = \sum_{t \in d} tf(t, d)$$

- $avdl$ — средняя длина документов коллекции
- Компонента нормализации на длину:

$$B = \left(1 - b + b \cdot \frac{dl}{avdl}\right), \;\; 0 \leq b \leq 1$$

- $b = 0$ — нет нормализации
- $b = 1$ — полная нормализация на среднюю длину документа

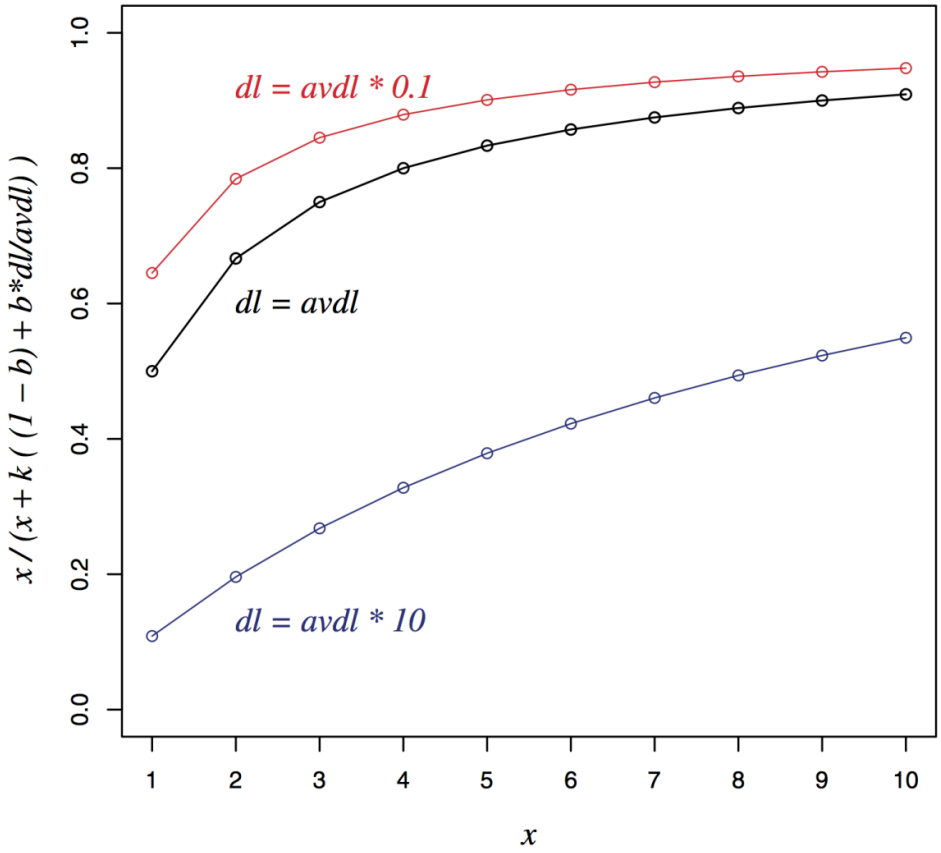
BM25 Model

TF с насыщаемостью и нормализацией на длину

$$TF(t,d) = \frac{(k_1 + 1) \cdot tf(t,d)}{k_1 \cdot \left(1 - b + b \cdot \frac{dl}{avdl}\right) + tf(t,d)}$$

BM25 Model

Нормализация на длину документа



BM25 Model

TF с насыщаемостью и нормализацией на длину

$$BM25(t,d) = IDF(t) \cdot \frac{(k_1+1)\cdot tf(t,d)}{k_1 \cdot \left(1 - b + b \cdot \frac{dl}{avdl}\right) + tf(t,d)}$$

- k_1 контролирует масштаб tf:
  - $k_1 = 0$ — бинарная модель (0 — терм не встретился, 1 — встретился)
  - при увеличении k_1 мы приближаемся к tf
- b контролирует нормализацию на длину
- Обычно на практике используется:
  - $k_1 \in [1.2,\ 2],\ b \approx 0.75$

BM25 Model

IDF эвристики

$$IDF(t) = log\left(\frac{N}{df(t)}\right)$$

BM25 Model

IDF эвристики

$$IDF(t) = log\left(\frac{N}{df(t)}\right)$$

- Недостатки:
  - деление на $df(t) = 0$
  - излишнее влияние редких токенов
  - нет сглаживания

BM25 Model

IDF эвристики

- Эвристика: $IDF(t) \rightarrow \frac{P(document\ does\ not\ contain\ t)}{P(document\ contains\ t)}$
- Пришли к:

$$IDF(t) = log\left(\frac{N}{df(t)}\right) \rightarrow log\left(\frac{N - df(t) + 0.5}{df(t) + 0.5} + 1\right)$$

BM25 Model

Ранжирующая функция

$$BM25(t,d) = log\left(\frac{N - df(t) + 0.5}{df(t) + 0.5} + 1\right) \cdot \frac{(k_1 + 1) \cdot tf(t,d)}{k_1 \cdot \left(1 - b + b \cdot \frac{dl}{avdl}\right) + tf(t,d)}$$

BM25 Model

Ранжирующая функция

$$BM25(t,d) = log\left(\frac{N - df(t) + 0.5}{df(t) + 0.5} + 1\right) \cdot \frac{(k_1 + 1) \cdot tf(t,d)}{k_1 \cdot \left(1 - b + b \cdot \frac{dl}{avdl}\right) + tf(t,d)}$$

$$score(Q,d) = \sum_{t \in Q \cap d} weight(t,d) = \sum_{t \in Q \cap d} BM25(t,d)$$

BM25 Model

Ранжирующая функция

$$score(Q,d) = \sum_{t \in Q \cap d} log\left(\frac{N - df(t) + 0.5}{df(t) + 0.5} + 1\right) \cdot \frac{(k_1 + 1) \cdot tf(t,d)}{k_1 \cdot \left(1 - b + b \cdot \frac{dl}{avdl}\right) + tf(t,d)}$$

- $\sum_{t \in Q \cap d}$ — чем больше общих слова в запросе и документе, тем лучше

BM25 Model

Ранжирующая функция

$$score(Q, d) = \sum_{t \in Q \cap d} log\left(\frac{N - df(t) + 0.5}{df(t) + 0.5} + 1\right) \cdot \frac{(k_1 + 1) \cdot tf(t, d)}{k_1 \cdot \left(1 - b + b \cdot \frac{dl}{avdl}\right) + tf(t, d)}$$

- $\sum_{t \in Q \cap d}$ — чем больше общих слова в запроса и документе, тем лучше
- $tf(t, d)$ — чем больше встречаемость токена в документе, тем лучше

BM25 Model

Ранжирующая функция

$$score(Q, d) = \sum_{t \in Q \cap d} log\left(\frac{N - df(t) + 0.5}{df(t) + 0.5} + 1\right) \cdot \frac{(k_1 + 1) \cdot tf(t, d)}{k_1 \cdot \left(1 - b + b \cdot \frac{dl}{avdl}\right) + tf(t, d)}$$

- $\sum_{t \in Q \cap d}$ — чем больше общих слова в запроса и документе, тем лучше
- $tf(t, d)$ — чем больше встречаемость токена, тем лучше
- $log\left(\frac{N - df(t) + 0.5}{df(t) + 0.5} + 1\right)$ — более частые токены — менее важные

Оценка поисковой системы

Оценка поисковой системы

Мотивация

- Какой алгоритм сортировки лучше: «быстрая» или «пузырьком»?

Оценка поисковой системы

Мотивация

- Какой алгоритм сортировки лучше: «быстрая» или «пузырьком»?
  - Быстрая: $O(n \cdot log(n))$
  - Пузырьком: $O(n^2)$

Оценка поисковой системы

Мотивация

- Какой алгоритм сортировки лучше: «быстрая» или «пузырьком»?
  - Быстрая: $O(n \cdot log(n))$
  - Пузырьком: $O(n^2)$
- Но результат работы одинаков!

Оценка поисковой системы

Мотивация

- Какой алгоритм сортировки лучше: «быстрая» или «пузырьком»?
- Какая IR модель лучше: TF-IDF или BM25?

Оценка поисковой системы

Мотивация

- Какой алгоритм сортировки лучше: «быстрая» или «пузырьком»?
- Какая IR модель лучше: TF-IDF или BM25?
- Можно оценивать по:
    - скорости работы системы
    - качеству результатов работы

Оценка поисковой системы

Мотивация

- Какой алгоритм сортировки лучше: «быстрая» или «пузырьком»?
- Какая IR модель лучше: TF-IDF или BM25?
- Можно оценивать по:
    - скорости работы системы — понятно как измерять
    - качеству результатов работы — непонятно как измерять

Оценка качества поиска

- Данные:
  - Коллекция документов
  - Набор запросов
  - Разметка — информация о том, для какого запроса какой документ является релевантным

Оценка качества поиска

- Данные:
  - Коллекция документов
  - Набор запросов
  - Разметка — информация о том, для какого запроса какой документ является релевантным
- Предположения:
  - Релевантность документа запросу объективно понятна
  - Все релевантные документы коллекции известны
  - Все релевантные документы одинаково сложно найти (нет влияния на перфоманс)
  - Релевантность документа не зависит от релевантности других документов

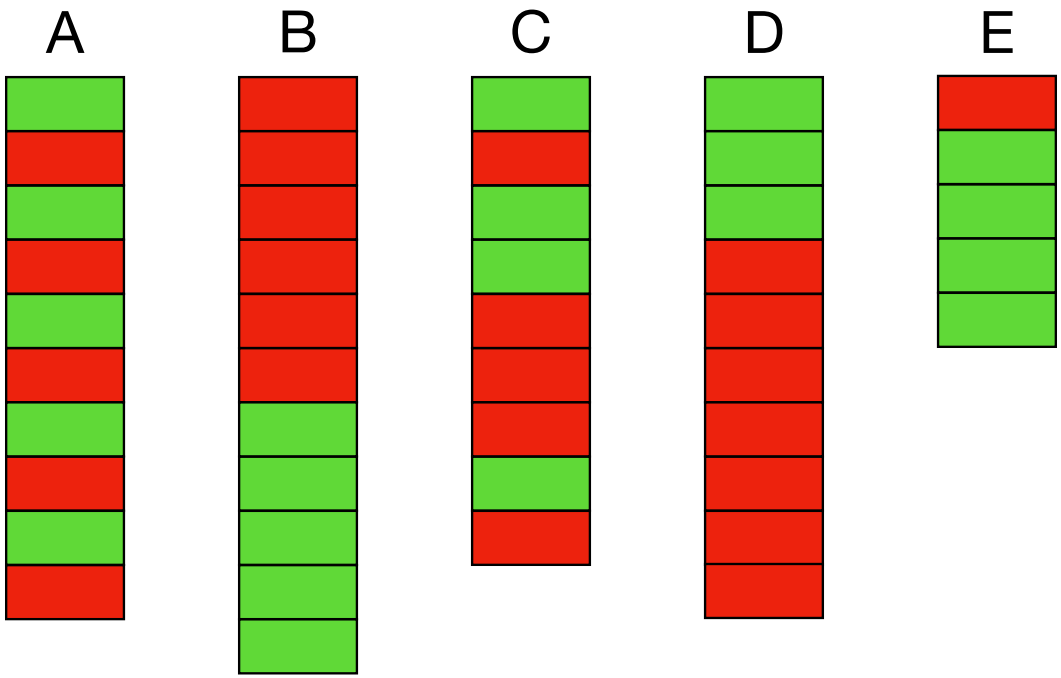
Оценка качества поиска

- У пользователя есть потребность
- Потребность формулируется в виде запроса
- Документы могут быть релевантными и нерелевантными
- Идеальная поисковая система возвращается все и только релевантные документы

Оценка качества поиска

Сравнение нескольких IR систем

- Запрос $Q = \{q_1, q_2, \ldots, q_n\}$
- В коллекции есть 8 документов, релевантных запросу

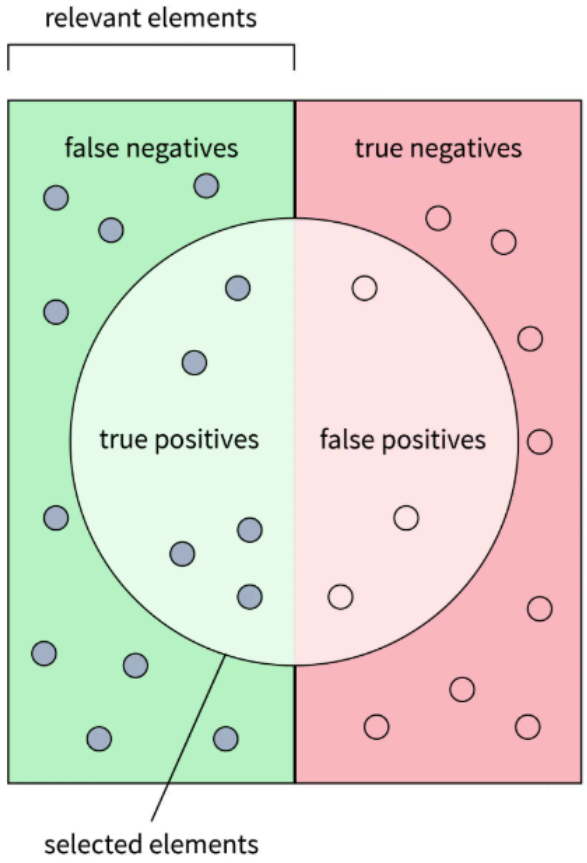


Set-based Metrics

Precision, recall

- $$precision = \frac{retrieved\_relevant}{retrieved} = \frac{\#TP}{\#TP + \#FP}$$
- $$recall = \frac{retrieved\_relevant}{total\_relevant} = \frac{\#TP}{\#TP + \#FN}$$

Confusion matrix	Retrieved	Not retrieved
Relevant	TP	FN
Irrelevant	FP	TN



Комбинация Precision и Recall

- $F_1 - score$ — среднее гармоническое Precision и Recall:

$$F_1 = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall}$$

Комбинация Precision и Recall

- $F_1 - score$ — среднее гармоническое Precision и Recall:

$$F_1 = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall}$$

- Больше гибкости — $F_\beta - score$:

$$F_1 = \frac{(\beta^2 + 1) \cdot Precision \cdot Recall}{\beta^2 \cdot Precision + Recall}$$

Комбинация Precision и Recall

- $F_1 - score -$ среднее гармоническое Precision и Recall:

$$F_1 = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall}$$

- Больше гибкости $- F_\beta - score$:

$$F_\beta = \frac{(\beta^2 + 1) \cdot Precision \cdot Recall}{\beta^2 \cdot Precision + Recall}$$

- β контролирует относительную важность $Precision$ и $Recall$:
  - $\beta = 1 \implies Precision$ и $Recall$ равно важны ($F_1 - score$)
  - $\beta = 3 \implies Recall$ в 3 раза важнее, чем $Precision$

Set-based Metrics

Precision, recall

- *Precision*: измеряет способность возвращать релевантные документы в топе выдачи
- *Recall*: измеряет способность находить все релевантные документы коллекции

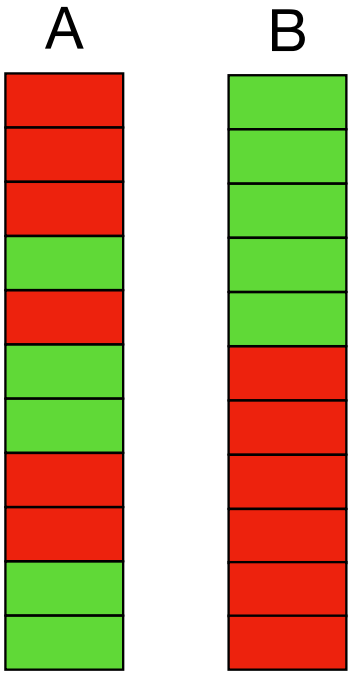
Set-based Metrics

Precision, recall

- $Precision$: измеряет способность возвращать релевантные документы в топе выдачи
- $Recall$: измеряет способность находить все релевантные документы коллекции
- С увеличением количества возвращаемых системой документов:
  - больше шансов найти все релевантные документы: $Recall \uparrow$
  - больше шансов найти больше нерелевантных документов: $Precision \downarrow$

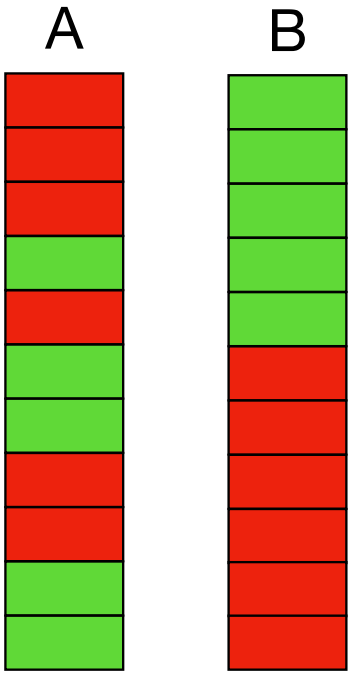
Set-based metrics

- Рассмотрим 2 системы A и B
- Обе системы вернули 10 документов
- Обе системы вернули 5 релевантных документов
- $Precision$, $Recall$ и $F_1 - score$ для этих двух систем равны



Set-based metrics

- Рассмотрим 2 системы A и B
- Обе системы вернули 10 документов
- Обе системы вернули 5 релевантных документов
- $Precision$, $Recall$ и $F_1 - score$ для этих двух систем равны
- Но правда ли, что они имеют одинаковое качество?



Метрики ранжирования

Precision@K

- K — параметр
- Измеряет $Precision$ для $top - K$ найденных документов
- Важно, что K зафиксирован для всех систем одинаковым

Метрики ранжирования

R—Precision

- Предположим, что мы знаем R — число документов, релевантных запросу, в коллекции
- R — параметр, специфичный для запроса
- Важно, что R разный для разных запросов
- Мера «идеальности» системы

Метрики ранжирования

Average Precision (AP)

Rank	Type	Recall	Precision
1	Rel	0.2	1.0
2	Non-Rel		
3	Rel	0.4	0.67
4	Non-Rel		
5	Non-Rel		
6	Rel	0.6	0.5
-	Rel	0.8	0.0
-	Rel	1.0	0.0

Метрики ранжирования

Average Precision (AP)

Rank	Type	Recall	Precision
1	Rel	0.2	1.0
2	Non-Rel		
3	Rel	0.4	0.67
4	Non-Rel		
5	Non-Rel		
6	Rel	0.6	0.5
-	Rel	0.8	0.0
-	Rel	1.0	0.0

$$AP = \frac{1}{5} \cdot \left(1 + \frac{2}{3} + \frac{3}{6}\right)$$

Метрики ранжирования

Average Precision (AP)

Rank	Type	Recall	Precision
1	Rel	0.2	1.0
2	Non-Rel		
3	Rel	0.4	0.67
4	Non-Rel		
5	Non-Rel		
6	Rel	0.6	0.5
∞	Rel	0.8	0.0
∞	Rel	1.0	0.0

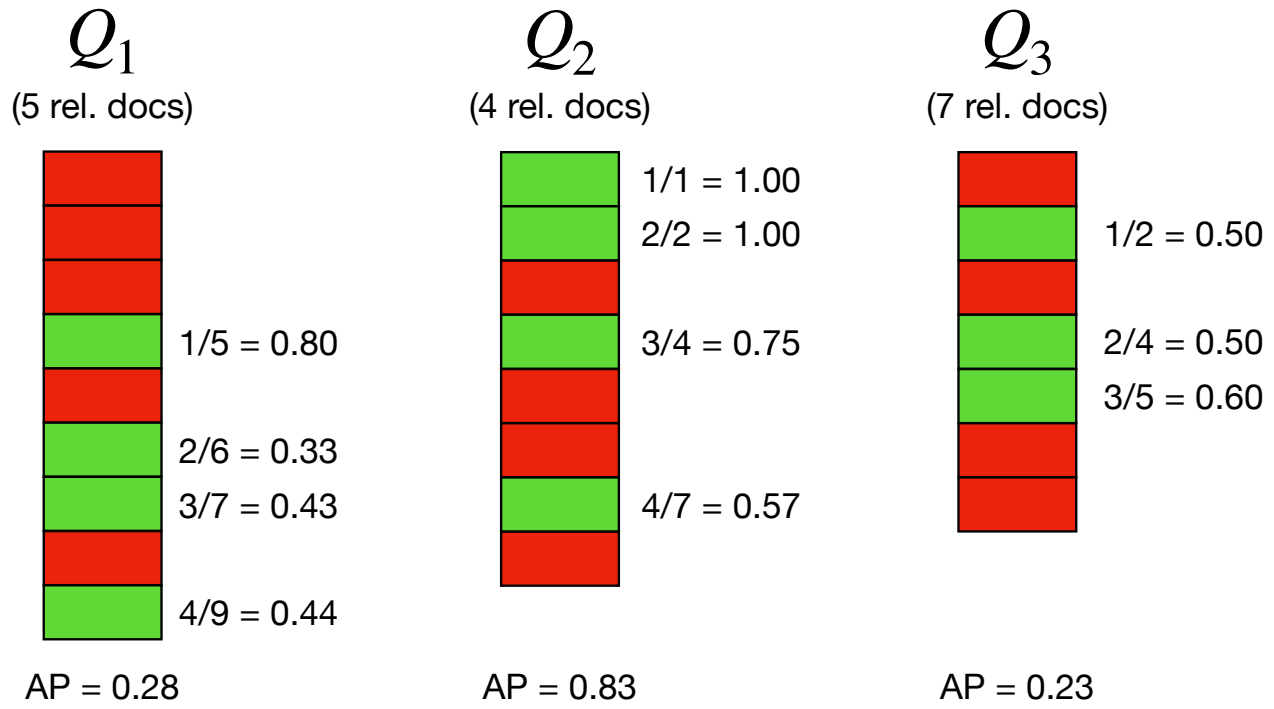
$$AP = \frac{1}{5} \cdot \left(1 + \frac{2}{3} + \frac{3}{6}\right)$$

$$AP = \frac{1}{N_{rel}} \sum_{d_i \in Rel} \frac{i}{Rank(d_i)}$$

Нацелена на то, чтобы релевантные документы находились как можно выше

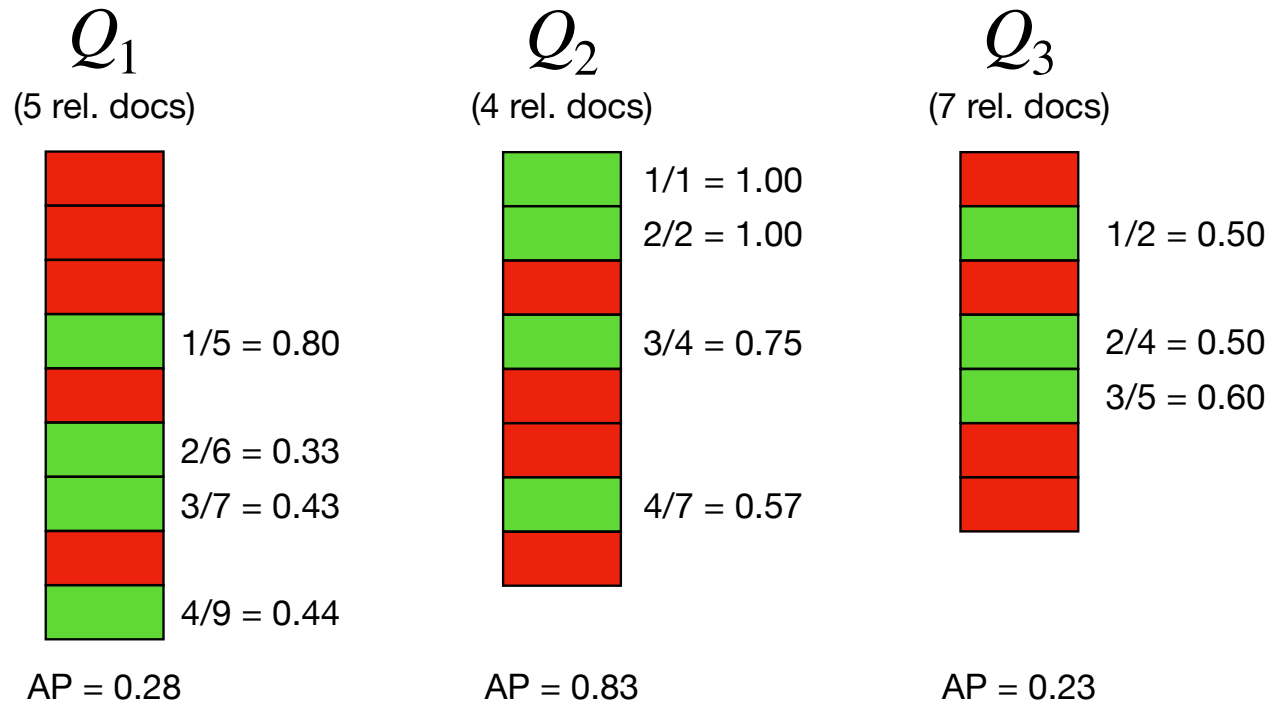
Метрики ранжирования

Mean Average Precision (MAP)



Метрики ранжирования

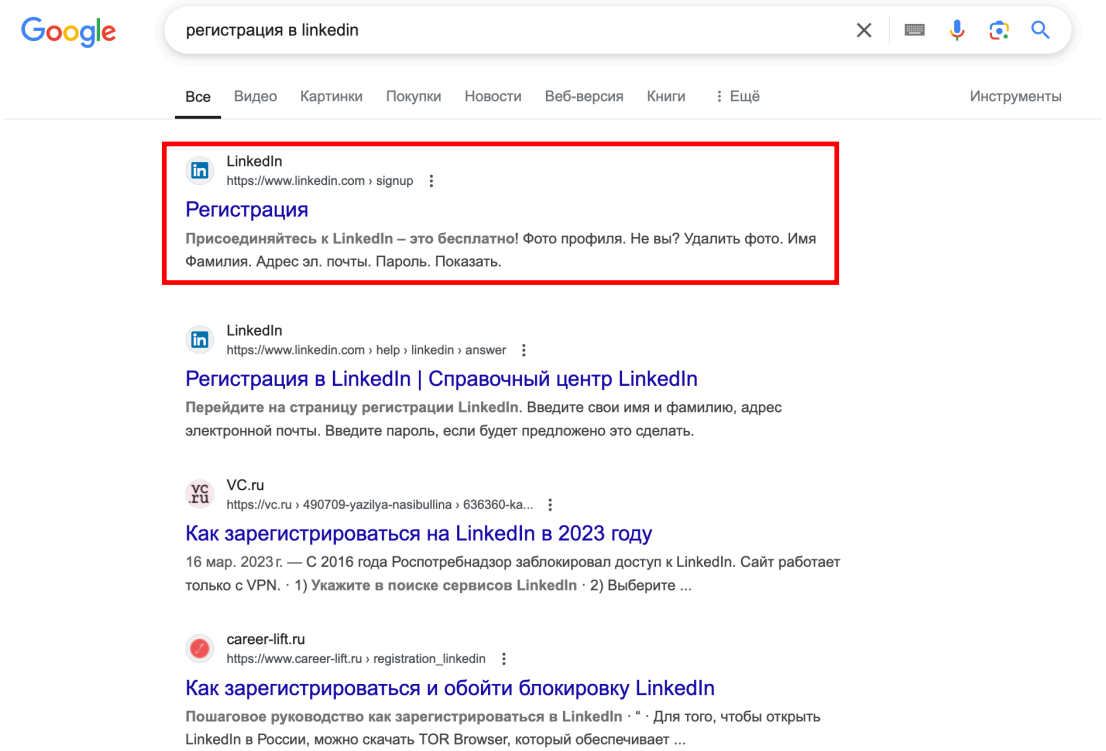
Mean Average Precision (MAP)



$$MAP = \frac{1}{N_Q} \sum_{Q} AP_Q = \frac{0.28 + 0.83 + 0.23}{3} = 0.45$$

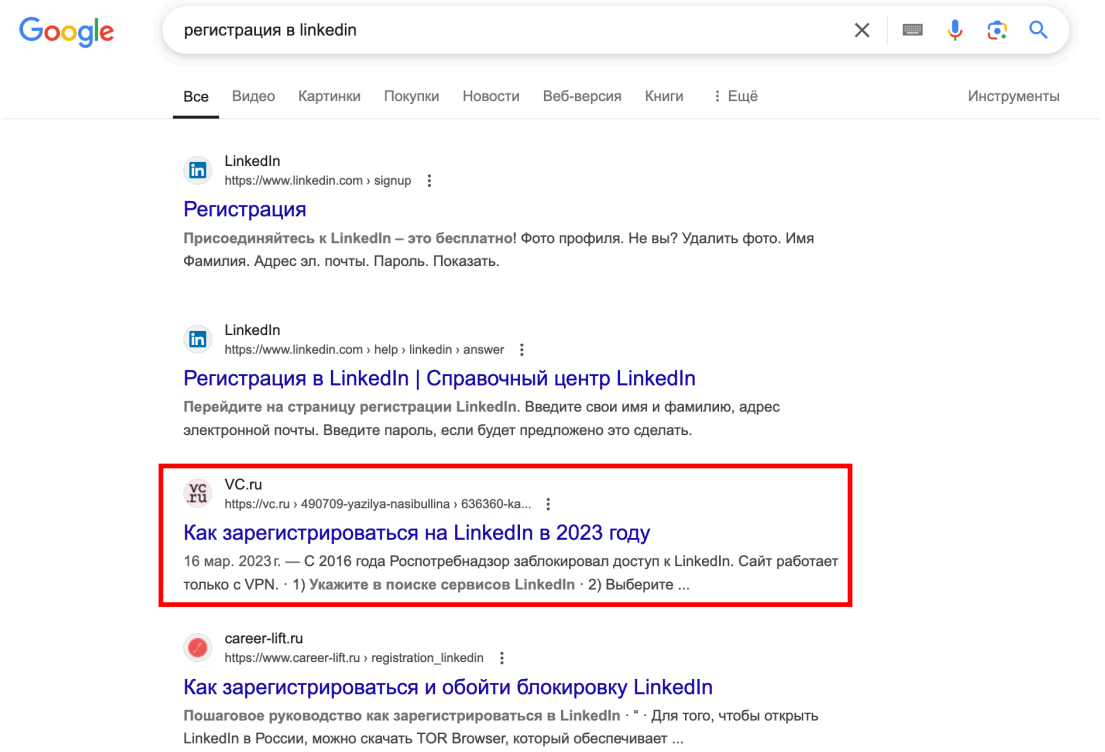
Небинарная релевантность

- **Vital**: документы, без которых запрос неполноценен



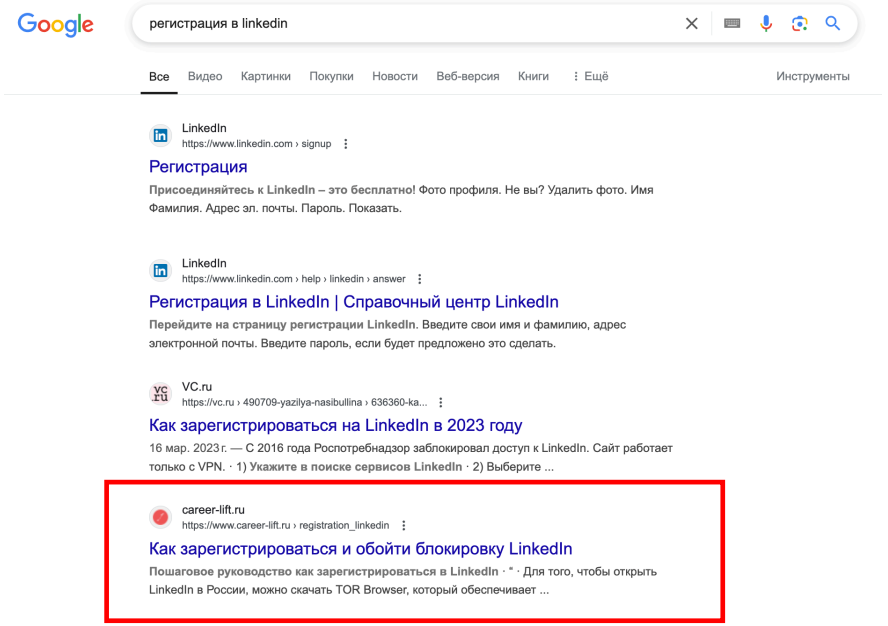
Небинарная релевантность

- **Vital**: документы, без которых запрос неполноценен
- **Useful**: документы, которые дают полный ответ на запрос и обладают полезными свойствами (например, авторитетность автора)



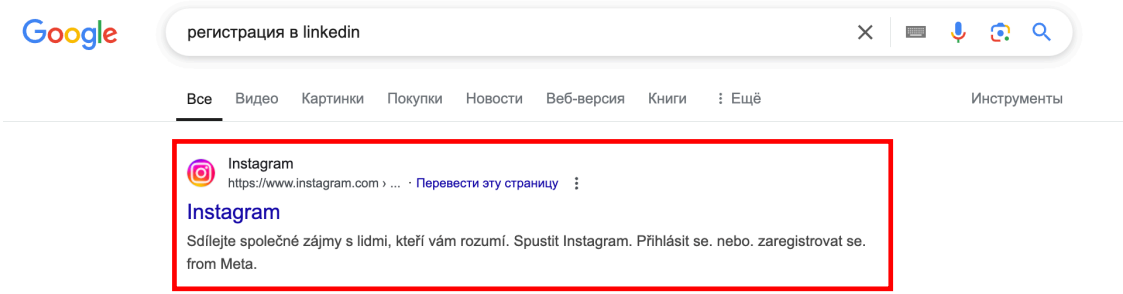
Небинарная релевантность

- **Vital**: документы, без которых запрос неполноценен
- **Useful**: документы, которые дают полный ответ на запрос и обладают полезными свойствами (например, авторитетность автора)
- **Relevant+**: документы, подходящие запросу



Небинарная релевантность

- **Vital**: документы, без которых запрос неполноценен
- **Useful**: документы, которые дают полный ответ на запрос и обладают полезными свойствами (например, авторитетность автора)
- **Relevant+**: документы, подходящие запросу
- **Relevant—**: документы, не подходящие запросу, но имеющие отношение к теме запроса



Небинарная релевантность

- **Vital**: документы, без которых запрос неполноценен
- **Useful**: документы, которые дают полный ответ на запрос и обладают полезными свойствами (например, авторитетность автора)
- **Relevant+**: документы, подходящие запросу
- **Relevant—**: документы, не подходящие запросу, но имеющие отношение к теме запроса
- **Irrelevant**: документы, совсем не подходящие запросу

Небинарная релевантность

- **Vital**: документы, без которых запрос неполноценен
- **Useful**: документы, которые дают полный ответ на запрос и обладают полезными свойствами (например, авторитетность автора)
- **Relevant+**: документы, подходящие запросу
- **Relevant—**: документы, не подходящие запросу, но имеющие отношение к теме запроса
- **Irrelevant**: документы, совсем не подходящие запросу

- N.B.: в ответе не обязательно содержатся все типы релевантностей

Инструкция для асессоров

1. Цель оценки

Определить, насколько документ соответствует поисковому запросу пользователя, используя пятибалльную шкалу релевантности. Оценка должна быть объективной и основываться на содержании документа, его полезности и соответствии запросу.

Инструкция для асессоров

2. Шкала релевантности

- **Vital (5 баллов)**:
  Документ является критически важным для запроса. Без него ответ на запрос будет неполным или недостаточным.
  Пример: Запрос «как сделать СЛР» — официальное руководство от Красного Креста с пошаговой инструкцией.

- **Useful (4 балла)**:
  Документ полностью отвечает на запрос и обладает дополнительными полезными свойствами, такими как авторитетность источника, структурированность, актуальность или уникальность информации.
  Пример: Запрос «лучшие практики SEO» — статья от известного эксперта с примерами и исследованиями.

- **Relevant+ (3 балла)**:
  Документ подходит запросу, но может быть недостаточно полным, авторитетным или полезным.
  Пример: Запрос «как вырастить помидоры» — статья с базовыми советами, но без деталей или научного обоснования.

- **Relevant— (2 балла)**:
  Документ не полностью подходит запросу, но имеет косвенное отношение к теме.
  Пример: Запрос «рецепт пиццы» — статья о истории пиццы без конкретных рецептов.

- **Irrelevant (1 балл)**:
  Документ совсем не подходит запросу и не имеет отношения к теме.
  Пример: Запрос «как починить велосипед» — статья о покупке автомобиля.

Инструкция для асессоров

3. Критерии оценки

- **Соответствие запросу:**
    - Документ должен напрямую отвечать на вопрос или соответствовать теме запроса
    - Учитывайте синонимы, близкие по смыслу формулировки и контекст
- **Полнота информации**:
    - Документ должен содержать достаточно информации для ответа на запрос
    - Если запрос требует конкретных данных (например, инструкция, статистика), они должны быть представлены
- **Авторитетность и надежность**:
    - Оценивайте источник информации. Официальные, экспертные или научные источники имеют больший вес
    - Учитывайте наличие ссылок на исследования, цитаты экспертов или подтвержденные данные
- **Актуальность**:
    - Информация должна быть актуальной на момент оценки. Устаревшие данные снижают релевантность
- **Полезные свойства**:
    - Дополнительные преимущества, такие как структурированность, наличие иллюстраций, видео, таблиц или интерактивных элементов

Инструкция для асессоров

4. Примеры оценки

Запрос	Документ	Оценка	Обоснование
Как испечь торт	Пошаговая инструкция с фото и видео от известного кулинарного блога	5 (Vital)	Полный ответ, полезные свойства (фото, видео), авторитетный источник
Как испечь торт	Подробный рецепт с объяснением техник выпечки от профессионального повара	4 (Useful)	Полный ответ, полезные свойства (техники, объяснения), авторитетный источник
Как испечь торт	Статья с общими советами по выпечке без конкретного рецепта	3 (Relevant+)	Подходит запросу, но не дает полного ответа
Как испечь торт	Статья о истории тортов	2 (Relevant—)	Косвенно относится к теме, но не отвечает на запрос
Как испечь торт	Новости о футбольном матче	1 (Irrelevant)	Не имеет отношения к запросу

Инструкция для асессоров

5. Рекомендации

- Внимательно читайте запрос и документ. Учитывайте контекст и возможные интерпретации запроса.
- Не оценивайте документ только по заголовку. Изучайте содержание.
- Если запрос допускает несколько интерпретаций, оценивайте документ по наиболее вероятному смыслу.
- Избегайте субъективных оценок. Ориентируйтесь на факты и объективные критерии.

Инструкция для асессоров

6. Частые ошибки

- **Переоценка**: Присвоение высокой оценки (Vital/Useful) документу, который не полностью отвечает на запрос.
- **Недооценка**: Присвоение низкой оценки (Relevant—/Irrelevant) документу, который частично или полностью отвечает на запрос.
- **Игнорирование контекста**: Неучет синонимов или альтернативных формулировок в запросе.

Инструкция для асессоров

7. Проверка и контроль

- Регулярно сверяйте свои оценки с эталонными примерами.
- Если возникают сомнения, обсудите их с командой или руководителем.
- Следите за обновлениями в критериях оценки.

Cumulative Gain

- Пусть наши документы могут иметь важности: $G = \{0,\ 1,\ 2,\ 3\}$
- Пусть наша выдача состоит из документов важностей соответственно:

$$G = \{3,\ 2,\ 3,\ 0,\ 0,\ 1,\ 2,\ 2,\ 3,\ 0,\ \dots\}$$

- Cumulative Gain:

$$CG@K = \sum_{i=1}^{K} G[i]$$

- $CG@1 = 3,\ CG@2 = 3 + 2 = 5,\ CG@5 = 3 + 2 + 3 + 0 + 0 = 8,\ \dots$

Discounted Cumulative Gain

- Пусть наши документы могут иметь важности: $G = \{0,\ 1,\ 2,\ 3\}$
- Пусть наша выдача состоит из документов важностей соответственно:

$$G = \{3,\ 2,\ 3,\ 0,\ 0,\ 1,\ 2,\ 2,\ 3,\ 0,\ \dots\}$$

- Discounted Cumulative Gain:

$$DCG@K = \sum_{i=1}^{K} \frac{G[i]}{log(i+1)}$$

- $DCG@1 = \frac{3}{log(2)},\ \ DCG@2 = \frac{3}{log(2)} + \frac{2}{log(3)},\ \ \dots$

Normalized Discounted Cumulative Gain

- Пусть наши документы могут иметь важности: $G = \{0,\ 1,\ 2,\ 3\}$
- Пусть наша выдача состоит из документов важностей соответственно:

$$G = \{3,\ 2,\ 3,\ 0,\ 0,\ 1,\ 2,\ 2,\ 3,\ 0,\ \dots\}$$

- Normalized Discounted Cumulative Gain:

$$nDCG@K = \frac{DCG@K}{iDCG@K}, \text{ где}$$

- $iDCG@K = DCG@K$ при отсортированном по убыванию G:
  $G_{ideal} = \{3,\ 3,\ \dots,\ 3,\ 2,\ 2,\ \dots,\ 2,\ 1,\ \dots,\ 1,\ 0,\ \dots\}$

DCG and nDCG

- В поисковой системе важно сначала оптимизировать DCG, а уже затем оптимизировать $nDCG$:
    - При оптимизации DCG мы обращаем больше внимания на те запросы, по которым существуют хорошие результаты, и мы стараемся хорошо работать на них
    - Оптимизация $nDCG$ осуществляет оптимизацию «длинного хвоста» запросов

Информационный поиск и ранжирование
Качество поиска

Андросов Дмитрий, 03.03.2025, AI Masters