Информационный поиск

Лекция 2
BM25

Дроздова Ксения
drozdova.xenia@gmail.com

Что было в прошлый раз

Индексирование (подготовка данных)

Булев поиск

Идея поиска через TF-IDF

Итоги:

Наша общая идея поиска - умножение матрицы на вектор + сортировка

Матрица это проиндексированная коллекция документов

Вектор – это проиндексированный запрос

tf-idf Weight

Bag of words

	Hitchhiker's Guide to Galaxy	Last Chance to See	Life, Universe & Everything	Restaurant at End of Universe	So Long & Thanks for all the Fish	Starship Titanic
galaxy	**0.2204**	0	**0.2140**	**0.2125**	**0.1880**	**0.1943**
zaphod	**0.5861**	0	**0.5174**	**0.6354**	**0.2288**	0
ship	**0**	**0**	**0**	**0**	**0**	**0**
arthur	**0.6230**	0	**0.6301**	**0.5931**	**0.6160**	0
fiordland	0	**1.5171**	0	0	0	0
santorini	0	0	**1.1437**	0	0	0
wordlings	0	0	0	0	**0.7780**	0

Indexed Collection

X

Arthur has Samsung Galaxy
0.46
0
0
0.51
0
0
0

Indexed Query

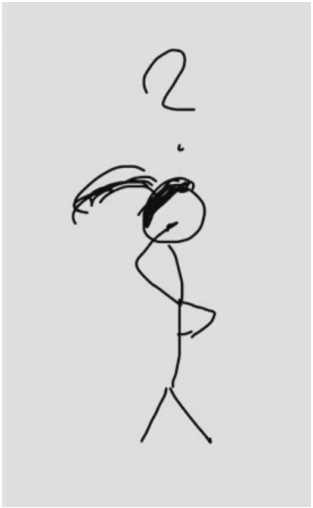
=

D1	0.43
D2	0
D3	0.42
D4	0.4
D5	0.41
D6	0.09

Scores

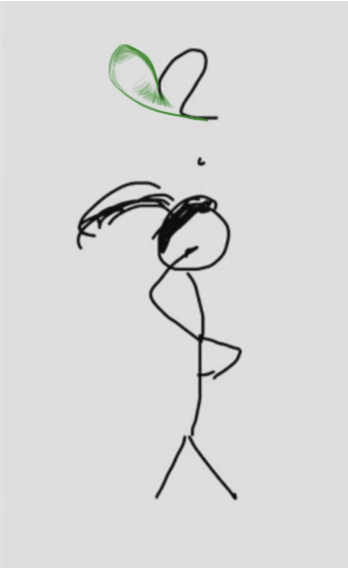
Формула BM25

$$bm25(Query, Doc) = \sum_{i=1}^{n} IDF(q_i) \cdot \frac{TF(q_i, Doc) \cdot (k+1)}{TF(q_i, Doc) + k \cdot (1 - b + b \cdot \frac{l(d)}{avgdl})}$$



$$bm25(Query, Doc) = \sum_{i=1}^{n} IDF \cdot \frac{TF \cdot (k+1)}{TF + k \cdot (1 - b + b \cdot \frac{l(d)}{avgdl})}$$

Формула BM25



$$bm25(Query, Doc) = \sum_{i=1}^{n} IDF \cdot \frac{TF \cdot (k+1)}{TF + k \cdot (1 - b + b \cdot \frac{l(d)}{avgdl})}$$

$$IDF(q_i) = \frac{N}{n(q_i)} \rightarrow \log \frac{N}{n(q_i)} \rightarrow \log \frac{N - n(q_i) + 0.5}{n(q_i) + 0.5}$$

$$bm25(Query, Doc) = \sum_{i=1}^{n} \log \left(\frac{N - n(q_i) + 0.5}{n(q_i) + 0.5} \right) \cdot \frac{TF(q_i, Doc) \cdot (k+1)}{TF(q_i, Doc) + k \cdot (1 - b + b \cdot \frac{l(d)}{avgdl})}$$

BM25 – это TF-IDF на стероидах

- BM25 – это оценка близости между запросом Query и документом Doc
- BM25 складывается из близостей каждого слова qi запроса Query и документа Doc
- формула BM25 по сути состоит из IDF * TF * Conct

Компоненты BM25

$$bm25(Query, Doc) = \sum_{i=1}^{n} \log\left(\frac{N - n(q_i) + 0.5}{n(q_i) + 0.5}\right) \cdot \frac{TF(q_i, Doc) \cdot (k+1)}{TF(q_i, Doc) + k \cdot (1 - b + b \cdot \frac{l(d)}{avgdl})}$$

Не зависит от слов запроса:

N – кол-во доков в коллекции

l(d) – длина документа Doc

avgdl – средняя длина документа в коллекции

k = 2

b = 0.75

Зависит от слов запроса:

n(qi) – кол-во доков, где есть qi

TF(qi, Doc) – частота qi в Doc

Компоненты BM25

$$IDF(q_i) = \frac{N}{n(q_i)} \rightarrow \log \frac{N}{n(q_i)} \rightarrow \log \frac{N - n(q_i) + 0.5}{n(q_i) + 0.5}$$

Чему равно значение IDF, когда слово входит в более чем половину документов?

Для того, чтобы статистика не была скошенной, можно скорректировать эту формулу:

1. Не учитывать отрицательные компоненты в итоговой сумме
2. Установить нижнюю границу: если IDF меньше e, то считать его равной e

Вариации BM25

BM11 - вариация BM25 при b=1

BM15 - вариация BM25 при b=0

BM15F (BM15Field) - вариация BM25, когда мы разбиваем документ на поля (заголовок, предисловие, пр) и присваиваем им веса в итоговой сумме

Моделируем реализацию

Все, что можно посчитать заранее, надо посчитать заранее и сохранить, чтобы на запрос тратилось минимум времени

N - количество документов в коллекции

l(d) – длина документа Doc

avgdl – средняя длина документа в коллекции

n(qi) - количество документов, содержащих слово qi

TF(qi, Doc) - частота слова qi в документе Doc

Моделируем реализацию

Сам алгоритм может быть реализован как в формуле – по циклу по словам запроса

Это нормально, и если у вас все предпосчитано, то даже займет не очень много времени

Но может быть реализован через концепт умножения матрицы на вектор, что, конечно, будет быстрее