

Ранжирование

МАКСИМОВСКАЯ
АНАСТАСИЯ

Постановка задачи

- $(i, j) \in R$ – набор пар, где первый объект из пары должен стоять после сортировки моделью выше второго объекта.
- Каждый объект может представлять собой пару (q, d) , состоящую из запроса и документа
- В этом случае порядок будет задан только на таких парах (q_1, d_1) и (q_2, d_2) , которые соответствуют одному запросу ($q_1 = q_2$)
- Хотим построить такую модель $a : X \rightarrow R$, что для $(i, j) \in R$ (и только для них) выполнено
$$a(x_i) < a(x_j)$$

mAP @ N

- **mAP @ N** (mean average precision). N – длина ленты рекомендаций. Р = число релевантных рекомендаций / общее число рекомендаций. AP @ N считается для каждого пользователя, после чего считается среднее и получаем mAP @ N

$$AP@N = \frac{1}{m} \sum_{k=1}^N (P(k) \text{ if } k^{th} \text{ item was relevant}) = \frac{1}{m} \sum_{k=1}^N P(k) \cdot rel(k).$$

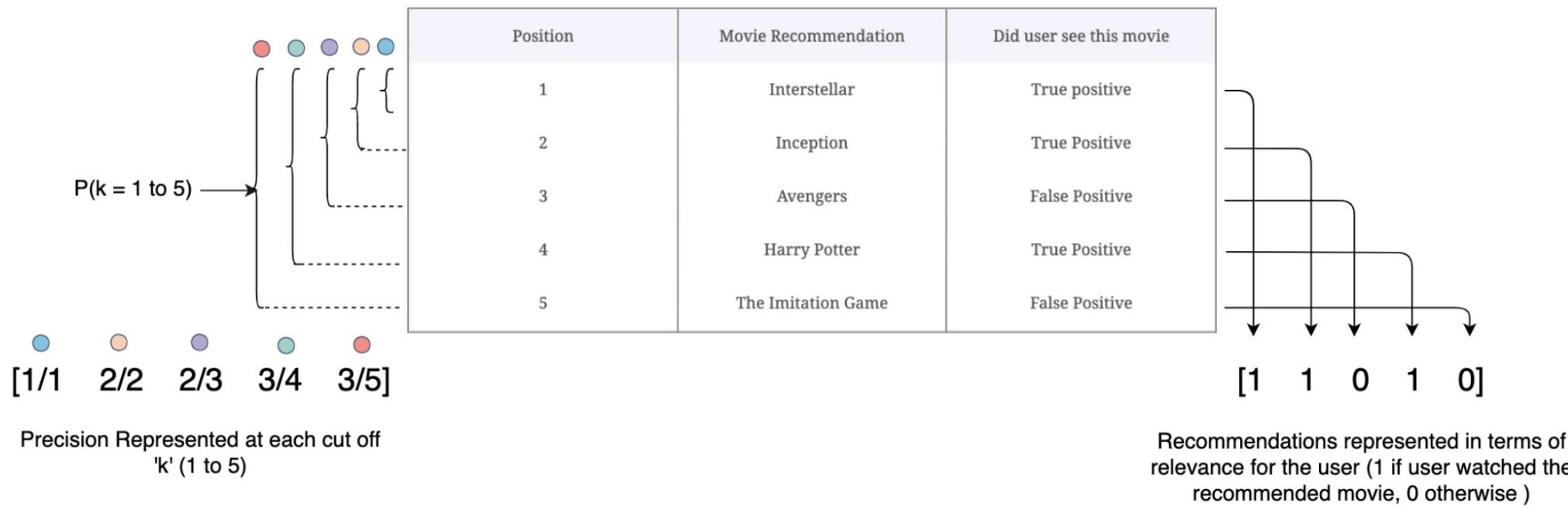
mAP@N

	Position	Movie Recommendation	Did user watch recommended movie	
$k=1 \{$	1	Interstellar	True positive	}
	2	Inception	True positive	
	3	Avengers	False positive	
	4	Harry Potter	True positive	
	5	The Imitation Game	False positive	

$P(k=1) = 1/1 \longrightarrow p @k = \text{proportion of all examples above that rank which are from the positive class}$

Calculating precision up to cutoff $k = 1$

mAP@N



mAR @ N

- **mAR @ N** (mean average recall). $r = \text{число релевантных рекомендаций} / \text{общее число возможных релевантных рекомендаций}$. В примере ниже рекомендовано 5 товаров из 10. метрика считается для каждого пользователя, после чего считается среднее и получаем mAR @ N

- **F1-score** = $2 * (\text{mAR} * \text{mAP}) / (\text{mAR} + \text{mAP})$

mAR @ N

Расчет:

1. Для каждого пользователя собираем таблицу вида рекомендация (id товара) и купил ли он ее (1 – купил, 0 – нет)
2. Считаем $r(k)$: сколько из первых k рекомендаций релевантны / N (общее число рекомендаций). Пример: Пусть у нас есть 5 товаров и список (1 1 0 1 0), где 1 означает, что пользователь купил товар, а 0 – нет. Тогда: $r(k=1) = 1/5$, $r(k=2)=2/5$, $r(k=3)=2/5$, $r(k=4)=3/5$, $r(k=5)=3/5$.
3. Перемножаем $r(k)$ на 1, если товар куплен, и 0, если нет. В примере выше: $1/5 * 1 + 2/5 * 1 + 2/5 * 0 + 3/5 * 1 + 3/5 * 0 = 1.2$ Это мы посчитали AR @N для одного пользователя.
4. Считаем метрику для всех пользователей по схеме выше и усредняем. Ответ – итоговое среднее число

Качество ранжирования

- Обозначим через a_{ui} предсказание модели для пользователя u и товара i . Отсортируем все товары по убыванию предсказания a_{ui} . Тогда для товара i_p на позиции p можно вычислить его полезность $g(r_{ui_p})$ и штраф за позицию $d(p)$.
- Введем метрику DCG (Discounted cumulative gain):

$$\text{DCG}@k(u) = \sum_{p=1}^k g(r_{ui_p})d(p)$$

Качество ранжирования

- Примерами конкретных функций могут служить такие $g(r)$ и $d(p)$:

$$g(r) = 2^r - 1 \quad d(p) = \frac{1}{\log(p+1)}.$$

- Чтобы значение метрики легче было интерпретировать, её можно поделить на значение DCG при идеальном ранжировании — в этом случае получим метрику nDCG (normalized DCG):

$$\text{nDCG}@k(u) = \frac{\text{DCG}@k(u)}{\max \text{DCG}@k(u)}.$$

Метрики качества

$$p_{i+1} = p_i(1 - y_{(i)})(1 - p_{\text{out}}),$$

где p_{out} — вероятность того, что пользователь уйдет, не узнав ответ на свой запрос.

Метрика pFound равна вероятности найти ответ среди первых k документов:

$$\text{pFound}@k(q) = \sum_{i=1}^k p_i y_{(i)}.$$

Признаки в моделях ранжирования

3 типа признаков

- Запросные — зависят только от запроса.
- Статические — зависят только от документа и могут быть рассчитаны заранее.
- Динамические — зависят от запроса и документа.

BM25

$$\text{BM25}(q, d) = \sum_{i=1}^n \text{IDF}(q_i) \frac{\text{tf}(q_i, d)(k_1 + 1)}{\text{tf}(q_i, d) + k_1 \left(1 - b + b \frac{|D|}{\bar{n}_d}\right)}$$

PageRank

- Число, которое показывает, насколько документ подходит под запрос
- Считаем, основываясь на ссылках между документами в нашем корпусе
- $PR(A)$ – PageRank документа A
- $L(B)$ – число ссылок с документа B, ссылающегося в том числе и на документ A

$$d \text{ -- коэффициент затухания, } d \in [0,1]$$
$$PR(A) = 1 - d + d \cdot \left(\frac{PR(B_1)}{L(B_1)} + \frac{PR(B_2)}{L(B_2)} + \dots + \frac{PR(B_N)}{L(B_N)} \right)$$

Методы ранжирования

Поточечные методы

- В поточечном (pointwise) подходе предлагается забыть про то, что мы решаем задачу ранжирования, и независимо для каждого объекта x предсказывать ответ y .
- В зависимости от типа ответов получим задачу классификации или регрессии.

Попарные методы

Функционал ошибки:

$$\sum_{(i,j) \in R} [a(x_j) - a(x_i) < 0]$$

Заменим индикатор ошибки $[z < 0]$ на его гладкую верхнюю оценку $L(z)$:

$$\sum_{(i,j) \in R} [a(x_j) - a(x_i) < 0] \leq \sum_{(i,j) \in R} L(a(x_j) - a(x_i)) .$$

- RankNet: В качестве оценки $L(z)$ можно брать, например, логистическую функцию $L(x) = \log(1 + e^{-\sigma z})$ с параметром $\sigma > 0$

Попарные методы: Lambda Rank

- Оптимизируем с помощью SGD, возьмем линейную модель $a(x) = \langle w, x \rangle$. Тогда шаг имеет вид:

$$w := w + \eta \frac{\sigma}{1 + \exp(\sigma \langle x_j - x_i, w \rangle)} (x_j - x_i)$$

- Домножим на смещение на изменение метрики ΔF_{ij} , которое произойдет при перестановке x_i и x_j местами в ранжировании:

$$w := w + \eta \frac{\sigma}{1 + \exp(\sigma \langle x_j - x_i, w \rangle)} |\Delta F_{ij}| (x_j - x_i)$$

Парные методы: RankSVM

$$\begin{cases} \frac{1}{2} \|w\|^2 + C \sum_{(i,j) \in R} \xi_{ij} \rightarrow \min_{w, \xi} \\ \langle w, x_j - x_i \rangle \geq 1 - \xi_{ij}, \quad (i, j) \in R; \\ \xi_{ij} \geq 0, \quad (i, j) \in R. \end{cases}$$

Списочные методы: ListNet

Предположим, что на самом деле модель выдает распределение на всех возможных перестановках документов, причем вероятность конкретной перестановки π определяется как:

$$P_z(\pi) = \prod_{j=1}^{n_q} \frac{\varphi(z_{\pi(j)})}{\sum_{k=j}^{n_q} \varphi(z_{\pi(k)})}$$

Списочные методы: ListNet

- Вероятности $P_z(\pi)$ задают распределение на множестве всех перестановок n_q элементов.
- Пусть перестановка π ставит объект x_i выше объекта x_j , и $z_i > z_j$. Если поменять эти объекты местами в перестановке (то есть поставить x_j выше, чем x_i), то новая перестановка будет иметь меньшую вероятность, чем старая. Иными словами, чем ближе перестановка к оптимальной, тем выше её вероятность.
- Максимальную вероятность имеет перестановка, которая сортирует объекты по убыванию z_i ; минимальную вероятность имеет обратная к ней перестановка.

Списочные методы: ListNet

- Всего перестановок объектов $nq!$, и посчитать по ним всем матожидание не представляется возможным.
- Чтобы упростить подсчеты, рассмотрим вероятность $P_z(j)$ попадания объекта x_j на первое место после перестановки. Можно показать, что они вычисляются по формуле:

$$P_z(j) = \frac{\varphi(z_j)}{\sum_{k=1}^n \varphi(z_k)}$$

Списочные методы: ListNet

- Данные вероятности образуют распределение на всех n_q документах.
- Также для объектов с $z_i > z_j$ выполнено $P_z(i) > P_z(j)$ — то есть введенные вероятности задают на объектах порядок, совпадающий с ранжированием по оценкам модели.
- Теперь мы можем сравнить истинное ранжирование документов и ранжирование по оценкам модели, посчитав кросс-энтропию между соответствующими им распределениями:

$$Q(y, z) = - \sum_{j=1}^{n_q} P_y(j) \log P_z(j)$$

Дополнительные материалы

➤ <https://www.coursera.org/learn/vvedeniye-informatsionnyy-poisk/home/welcome>

Подготовка к собеседованиям

План

1. Структура технического собеседования
2. Подготовка рассказа о предыдущих проектах
3. Теоретические вопросы: как готовиться и какие бывают
4. Пример задачи по Python
5. Примеры задач по SQL
6. Кейс по ML

Подготовка

1. Изучаем вакансию (не только требования, но и обязанности)
2. Узнаем у HR структуру отбора (сколько интервью, с кем)
3. Спрашиваем про содержание технического собеса
4. При подготовке делаем особый акцент на те требования, которые были озвучены в вакансии + готовим пару рассказов про свои успешные проекты (в деталях: что использовали, как работают модели/фреймворки)

Общие вопросы

1. Расскажите о своем опыте/реализованных проектах
2. Какие модели и фреймворки Вы использовали?
3. Как они работают? В чем их плюсы и минусы?
4. Расскажите как реализован Ваш последний проект

Python

1. Какие Ваши любимые модули/библиотеки? Почему?
2. Как реализован X в них?
3. Как происходит управление памятью в Python?
4. Что такое декораторы и как они работают?
5. В чем отличие списков от кортежей, множеств?
6. Какие есть методы работы со словарями?

Еще: <https://habr.com/ru/company/mailru/blog/506824/>

Несколько категорий вопросов

1. Общая теория: как работают модели, плюсы-минусы-сравнения друг с другом
2. Программирование: либо просто питон, либо алгоритмы
3. Общий интерес к сфере, тренды сейчас
4. Специфичные для области/индустрии вопросы

Примеры вопросов по теории

1. В чем разница между смещением и разбросом?
2. Что такое норма?
3. В чем разница между ошибками 1 и 2 рода?
4. Как работает метод SVD?
5. Что такое QR-разложение?

Примеры популярных вопросов по теории

1. Как работает логистическая регрессия?
2. Почему оптимизация logloss приводит к оценке вероятностей?
3. Как работает градиентный бустинг в задачах регрессии и классификации?
4. Вопросы про глубину и количество деревьев в случайном лесе и градиентном бустинге
5. Плюсы и минусы ROC-AUC, его уместность в работе с несбалансированными выборками и задаче ранжирования
6. Как оценивается статистическая значимость в A/B тестировании?

Источники вопросов

1. <https://ds-interviews.org/>
2. Python: <https://www.hackerrank.com/interview/interview-preparation-kit>,
<https://leetcode.com/>, Марк Лутц “Изучаем Python” (скачать можно,
например, тут https://vk.com/wall-51126445_43964)
3. SQL: <https://www.sql-ex.ru/>, <http://www.sql-tutorial.ru/>
4. ML: C. Bishop “Pattern Recognition and Machine Learning
(<https://github.com/peteflorence/MachineLearning6.867/blob/master/Bishop%20-%20Pattern%20Recognition%20and%20Machine%20Learning.pdf>),
<https://online.stanford.edu/search-catalog>

Источники вопросов

1. <https://use-the-index-luke.com/3-minute-test>
2. <https://www.interviews.ai/>
3. <https://interviewing.fyi/>
4. <https://github.com/alexeygrigorev/data-science-interviews/>
5. <https://www.springboard.com/blog/machine-learning-interview-questions/>
6. <https://medium.com/analytics-vidhya/test-your-skills-26-data-science-interview-questions-answers-69cb2b223e57>
7. <https://drive.google.com/file/d/1r2bmQtIxlr8J-TpBeFi4CWzbChJG4iHe/view>
8. <https://github.com/slgero/testovoe> – сборник тестовых
9. <https://www.pramp.com/#/> – для мок-интервью
10. Для алгоритмов: Cracking the Coding Interview

Примеры задач

Python

1. Реализуйте подсчет n-ого числа из последовательности Фибоначчи с помощью генераторов.

Python

```
def fib(num):
    if num < 2:
        return num
    fib_list = [0] * num
    fib_list[0] = 1
    fib_list[1] = 1
    for i in range(2, num):
        fib_list[i] = fib_list[i-1] + fib_list[i-2]
    return fib_list[num-1]

for num in range(1, 11):
    print(fib(num))
```

1
1
2
3
5
8
13
21
34
55

Python

```
: def fibonacci_generator(n):
    first_fib, second_fib = 1, 1
    for _ in range(n):
        yield first_fib
        first_fib, second_fib = second_fib, first_fib + second_fib
```

```
: n = 10
: for num in fibonacci_generator(n):
:     print(num)
```

```
1
1
2
3
5
8
13
21
34
55
```

SQL

1. Вывести отдел с наибольшим числом сотрудников
2. Вывести список сотрудников, получающих заработную плату выше, чем у руководителя

SQL

Department

Id	name
1	Финансы
2	Риски
3	Розница
4	Безопасность
..	..
1000	ДКК

Personal

id_head – id – руководителя
id_dep – id департамента

Id	Id_head	Id_dep	name	sal
1	1	2	Бегинс	45 000
2	1	2	Поттер	80 000
3	2	2	Чапаев	100 000
4	4	4	Шилов	65 000
...
10000	5	3	Наумов	64 500

SQL

2.1 Вывести отдел с наибольшим числом сотрудников

```
SELECT d.name  
FROM Personal AS p JOIN Department AS d ON p.Id_dep = d.Id  
GROUP BY d.name  
ORDER BY count(d.name) DESC LIMIT 1;
```

2.2 Вывести список сотрудников, получающих заработную плату выше, чем у руководителя

```
SELECT p.name  
FROM Personal p JOIN Personal p2 ON p.Id_head = p2.Id  
WHERE p2.sal < p.sal;
```

SQL

У Вас есть две таблицы: t1 с количеством строк M и t2 с количеством строк N.
Какое максимальное количество строк в итоговой таблице при INNER JOIN?

SQL

У Вас есть две таблицы: $t1$ с количеством строк M и $t2$ с количеством строк N .
Какое максимальное количество строк в итоговой таблице при INNER JOIN?

- Рассмотрим краевой случай: в примере ниже все индексы одинаковые –
1. В первой таблице 3 строки, в последней 4. Но колонки разные, и inner join заметчил все возможные варианты, отсюда $M*N$.

SQL

```
1  DELETE FROM table1;
2  DELETE FROM table2;
3
4  INSERT INTO table1 (ID, Name) VALUES (1, "Katya");
5  INSERT INTO table1 (ID, Name) VALUES (1, "Pavel");
6  INSERT INTO table1 (ID, Name) VALUES (1, "Jack");
7
8  INSERT INTO table2 (ID, LastName) VALUES (1, "Petrova");
9  INSERT INTO table2 (ID, LastName) VALUES (1, "Smirnova");
10 INSERT INTO table2 (ID, LastName) VALUES (1, "Ivanova");
11 INSERT INTO table2 (ID, LastName) VALUES (1, "Kolmogorov");
12
13 SELECT count(*) FROM table1 t1 INNER JOIN table2 t2 ON t1.ID=t2.ID;
```

SQL

	ID	Name	ID	LastName	
1	1	Katya	1	Ivanova	
2	1	Katya	1	Kolmogorov	
3	1	Katya	1	Petrova	
4	1	Katya	1	Smirnova	
5	1	Pavel	1	Ivanova	
6	1	Pavel	1	Kolmogorov	
7	1	Pavel	1	Petrova	
8	1	Pavel	1	Smirnova	
9	1	Jack	1	Ivanova	
10	1	Jack	1	Kolmogorov	
11	1	Jack	1	Petrova	
12	1	Jack	1	Smirnova	

Итоговая таблица

Кейс с реального интервью

Вы работаете в HR-блоке крупной компании. Перед Вами поставили следующие задачи:

1. Определять то, что сотрудник перегружен
2. Определять, что сотрудник выгорает и может уволиться

Вопросы:

1. Какие данные Вам будут нужны?
2. Какими моделями/метриками вы воспользуетесь?