Machine learning ranking

Ранжирование

Мухаметшина Даяна

Higher School of Economics

Table of contents

Задача ранжирования

Постановка №1

Пусть задано пространство объектов $\mathbb{X}$ (например, $\mathbb{R}^d$) и некоторая обучающая выборка $X = \{x_1, ..., x_l\} \subset \mathbb{X}$. Также пусть дан некоторый порядок на объектах обучающей выборке — то есть набор таких пар $(i, j) \in R \subset \{1, ..., l\}^2$, что первый объект из пары должен стоять после сортировки моделью выше второго объекта. Множество пар R заменяет собой целевую переменную.

Требуется построить такую модель $a : \mathbb{X} \to \mathbb{R}$, что для $(i, j) \in R$ (и только для них) выполнено $a(x_i) < a(x_j)$.

Постановка №2

Формально, рассмотрим N объектов $U = \{u_i\}_{i=1}^{N}$ и M элементов $E = \{e_j\}_{j=1}^{M}$. Результат работы алгоритма ранжирования элементов K для объекта $u \in U$ — это отображение $r : E \to \mathbb{R}$. Набор весов $\{r(e)\}_{e \in E}$ задает перестановку $\pi : \{1, .., M\} \to \{1, .., M\}$ на наборе элементов элементов E.

$r^{true} : E \to [0, 1]$ — эталонная функция релевантности, характеризующая «настоящую» релевантность элементов для данного объекта.

Метрики

Для бинарного ранжирования можно использовать уже известные нам метрики F–score, AUC–ROC, и другие.

Можно сказать, что нас интересует качество только релевантных объектов. Будем считать, что ими являются первые k.

Далее ”Топ k” обозначаются как ”@k”.

Precision

precision@K — точность на K элементах — базовая метрика качества ранжирования для одного объекта. Допустим, наш алгоритм ранжирования выдал оценки релевантности для каждого элемента $\{r(e)\}_{e \in E}$. Отобрав среди них первые $K \leq M$ элементов с наибольшим $r(e)$ можно посчитать долю релевантных. Именно это и делает precision at K:

$$precision@K = \frac{\sum_{k=1}^{K} r^{true}(\pi^{-1}(k))}{K}$$

Замечание: под $\pi^{-1}(k)$ понимается элемент $e \in E$, который в результате перестановки $\pi(e)$ оказался на k-ой позиции.

Проблема: не учитывает порядок в топе.

Все precision метрики используются в случае $r^{true} = \{0, 1\}$

Average and Mean Precision

- **average precision@K** —

$$a\_precision@K = \frac{\sum_{k=1}^{K} r^{true}(\pi^{-1}(k)) \cdot p@k}{K}$$

Исправляет проблему, учитывает порядок в топе.

- **mean average precision@K** —

$$MAP@K = \frac{1}{N} \sum_{j=1}^{N} AP@K_j$$

Используется в случае множества объектов (пользователей, поисковых запросов).

Cumulative Gain

Рассмотрим один объект и K элементов с наибольшим $r(e)$.
Cumulative gain at K (CG@K) — базовая метрика ранжирования, которая использует простую идею: чем выше релевантные элементы в этом топе, тем лучше.

$$CG@K = \sum_{k=1}^{K} r^{true}(\pi^{-1}(k))$$

Плюсы: может использоваться для не бинарных значений r^{true}.

Минусы: не нормализовано, не учитывает позицию.

Discounted Cumulative Gain

- Discounted Cumulative gain at K (DCG@K) —

$$DCG@K = \sum_{k=1}^{K} \frac{2^{r^{true}(\pi^{-1}(k))} - 1}{\log_2(k+1)}$$

- Ideal Cumulative gain at K (DCG@K) —

$$IDCG@K = \sum_{k=1}^{K} \frac{1}{\log_2(k+1)}$$

- Normalized Discounted Cumulative gain at K (nDCG@K) —

$$nDCG@K = \frac{DCG@K}{IDCG@K}$$

По аналогии с $map@K$ можно посчитать $nDCG@K$, усредненный по всем объектам.

Каскадная модель поведения

Ранее мы предполагали, что ”просмотр” каждого элемента независим. На самом деле мы понимаем, что ”просмотр” следующего элемента зависит от ”удовлетворенности” предыдущим. Модели поведения пользователя, где изучение предложенных ему элементов происходит последовательно и вероятность просмотра элемента зависит от релевантности предыдущих называются **каскадными**.

Expected reciprocal rank

Expected reciprocal rank (ERR):

$$ERR@K = \frac{1}{K}\sum_{k=1}^{K} P(\text{объект остановится на k-ом элементе})$$

$$P(\text{объект остановится на k-ом элементе}) = p_k \prod_{i=1}^{k-1}(1 - p_i)$$

p_k – объект удовлетворится на k-ом элементе.

Так как в нашем случае $r(e) \in [0, 1]$, то можем рассмотреть простой вариант:

$$p_k = r^{true}(\pi^{-1}(k))$$

В общем же случае чаще всего используют следующую формулу:

$$p_k = \frac{2^{r^{true}(\pi^{-1}(k))} - 1}{2^{max(r^{true})} - 1}$$

pFound

Метрика Яндекса:

$$pFound@K = \sum_{k=1}^{K} pLook(k)pRel(k)$$

$$pLook(k) = pLook(k-1) \cdot (1 - pRel(k-1)) \cdot (1 - pBreak)$$

$pRel(i) = 2^{r^{true}(\pi^{-1}(i))} - 1$, если $r^{true}(\pi^{-1}(i)) > 0$, ноль иначе.

$pBreak$ – вероятность, что просмотр прекратиться по внешним причинам.

Типы ранжирования

Pointwise approach

Научиться ставить правильные метки документам. Подходы:

1. взвесить оценки и найти функцию наиболее близкую к ним
2. построить систему взвешенных бинарных классификаторов и устроить структуру ранжирования

То есть мы можем применить уже знакомые нам регрессию и классификацию, в зависимости от типа ответов.

Применение: OPRF (1989), SLR (1992), IR-SVM (2006), McRank (2007) (очень старые)

Минусы:

- закрыли глаза на изначальную проблему
- поточечная метрика не учитывает порядок
- штраф за величину ранга

Надо:

- порядок важнее корректности предсказания!

Pairwise approach

Вспомним первую постановку задачи. Тогда функционал ошибки:

$\sum_{(i,j)\in R} [a(j) - a(i) < 0]$, R – множество пар с известным порядком.

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

- $L(M) = log(1 + e^{-M})$ это RankNet
- $L(M) = e^{-M}$ это RankBoost
- $L(M) = (1 - M)_+$ это RankSVM

Минусы:

- более медленный
- не фокусируется на топе релевантных
- не учитываются зависимости между парами

Примеры: RankSVM (2000), RankBoost (2003), LambdaRank(2006), GBRank (2007), RankRLS (2007), SortNet (2008), MatrixNet (2009), YetiRank (2010), CRR (2010)

Listwise approach

Пытаемся предсказать перестановку.

Есть два подхода:

1. Оптимизация метрик (например nDCG) (SoftRank, AdaRank)
2. Минимизация функции потерь, которая построена на понимании того, как мы хотим от ранжировать. (ListNet, ListMLE)

Минусы:

- медленно
- сложная структура
- из-за большой размерности данных можно использовать не все

E.N.D.

По идее ListWise должен работать лучше остальных, но к примеру утверждается, что в Яндексе лучше работает попарный метод.

Спасибо!

Список литературы.

https://habr.com/company/econtenta/blog/303458/
http://www.machinelearning.ru/wiki/images/8/89/Voron-ML-Ranking-slides.pdf
https://en.wikipedia.org/wiki/Evaluation_measures_(information_retrieval)