

NDCG及实现



yone

RS & NLP

51 人赞同了该文章

Normalized Discounted Cumulative Gain(归一化折损累计增益)

NDCG用作排序结果的评价指标，评价排序的准确性。

推荐系统通常为某用户返回一个item列表，假设列表长度为K，这时可以用NDCG@K评价该排序列表与用户真实交互列表的差距。

解释：

- **Gain**：表示列表中每一个item的相关性分数

$$Gain = r(i)$$

- **Cumulative Gain**：表示对K个item的Gain进行累加

$$CG@K = \sum_i^K r(i)$$

- **Discounted Cumulative Gain**：考虑排序顺序的因素，使得排名靠前的item增益更高，对排名靠后的item进行折损。

$$DCG@K = \sum_i^K \frac{r(i)}{2^{i+1}}$$

如果相关性分数 $r(i)$ 只有 (0, 1) 两种取值时， $DCG@K$ 有另一种表达。其实就是如果算法返回的排序列表中的item出现在真实交互列表中时，分子加1，否则跳过。

$$DCG@K = \sum_i^K \frac{2^{r(i)} - 1}{\log_2(i + 1)}$$

计算每个用户真实列表的DCG分数，用IDCG表示，然后用每个用户的DCG与IDCG之比作为每个用户归一化后的分值，最后对每个用户取平均得到最终的分值，即NDCG。

$$NDCG_u@K = \frac{DCG_u@K}{IDCG_u}$$

$$NDCG@K = \frac{NDCG_u@K}{|u|}$$

NDCG实现

```
import numpy as np

def getDCG(scores):
    return np.sum(
        np.divide(np.power(2, scores) - 1, np.log2(np.arange(scores.shape[0], dtype=np
            dtype=np.float32))

def getNDCG(rank_list, pos_items):
    relevance = np.ones_like(pos_items)
    it2rel = {it: r for it, r in zip(pos_items, relevance)}
    rank_scores = np.asarray([it2rel.get(it, 0.0) for it in rank_list], dtype=np.float

    idcg = getDCG(relevance)

    dcg = getDCG(rank_scores)

    if dcg == 0.0:
        return 0.0

    ndcg = dcg / idcg
    return ndcg

l1 = [1, 4, 5]
l2 = [1, 2, 3]
a = getNDCG(l1, l2)
```