Assignment-2: Evaluation Metrics

21307289 刘森元

Exercise. 1

根据 MAP@K，MRR@K 的计算公式，可写出代码（完整代码见 Exercise-1.py）

```python
def compute_AP(rankings, relevant_docs, k):
    num_relevant = 0
    precision_sum = 0.0

    for i in range(k):
        doc = rankings[i]
        if doc in relevant_docs:
            num_relevant += 1
            precision = num_relevant / (i + 1)
            precision_sum += precision

    avg_precision = precision_sum / min(k, len(relevant_docs))
    return avg_precision


def compute_RR(rankings, relevant_docs, k):
    for i in range(k):
        doc = rankings[i]
        if doc in relevant_docs:
            return 1.0 / (i + 1)

    return 0.0
```

根据问题代入数据，有

```
qiu_nangong@Somebodys-MacBook-Pro ▶ python3 Exercise-1.py
Query 1 AP@5 : 0.4833333333333333
Query 1 AP@10: 0.7329365079365079
Query 1 RR@5 : 1.0
Query 1 RR@10: 1.0
Query 2 AP@5 : 0.25
Query 2 AP@10: 0.375
Query 2 RR@5 : 0.5
Query 2 RR@10: 0.5
Query 3 AP@5 : 0.1111111111111111
Query 3 AP@10: 0.30634920634920637
Query 3 RR@5 : 0.3333333333333333
Query 3 RR@10: 0.3333333333333333
MAP@5 : 0.28148148148148144
MAP@10: 0.4714285714285715
```

```
MRR@5 : 0.611111111111111
MRR@10: 0.611111111111111
```

Exercise. 2

根据计算公式，可写出代码（完整代码见 Exercise-2.py）

```
# (a) 计算P@5和P@10
p_5 = sum(doc["binary_relevance"] for doc in ranking[:5]) / 5
p_10 = sum(doc["binary_relevance"] for doc in ranking[:10]) / 10

# (b) 计算R@5和R@10
relevant_docs = sum(doc["binary_relevance"] for doc in ranking)
r_5 = sum(doc["binary_relevance"] for doc in ranking[:5]) / relevant_docs
r_10 = sum(doc["binary_relevance"] for doc in ranking[:10]) / relevant_docs

# (c) 提供一个最大化P@5的示例排名
max_p_5_ranking = sorted(ranking,
                         key=lambda doc: doc["binary_relevance"],
                         reverse=True)[:5]

# (d) 提供一个最大化P@10的示例排名
max_p_10_ranking = sorted(ranking,
                          key=lambda doc: doc["binary_relevance"],
                          reverse=True)[:10]

# (e) 提供一个最大化R@5的示例排名
max_r_5_ranking = sorted(ranking, key=lambda doc: doc["rank"])[:5]

# (f) 提供一个最大化R@10的示例排名
max_r_10_ranking = sorted(ranking, key=lambda doc: doc["rank"])[:10]

# (g) 在这种情况下，可以使用R-Precision来设置K值。由于有7个相关文档，所以K = 7。

# (h) 计算平均准确率 (AP)
ap = sum((sum(doc["binary_relevance"] for doc in ranking[:i + 1]) / (i + 1))
         for i, doc in enumerate(ranking)
         if doc["binary_relevance"] == 1) / relevant_docs

# (i) 提供一个最大化AP的示例排名
max_ap_ranking = sorted(ranking,
                        key=lambda doc: doc["graded_relevance"],
                        reverse=True)

# (j) 计算DCG5
dcg_5 = ranking[0]["graded_relevance"] + sum(
    doc["graded_relevance"] / math.log2(doc["rank"]) for doc in ranking[1:5])
```

```
# (k) 计算NDCG5
ideal_ranking = [{
    "rank": 1,
    "graded_relevance": 4
}, {
    "rank": 2,
    "graded_relevance": 4
}, {
    "rank": 3,
    "graded_relevance": 3
}, {
    "rank": 4,
    "graded_relevance": 2
}, {
    "rank": 5,
    "graded_relevance": 1
}]
idcg_5 = ideal_ranking[0]["graded_relevance"] + sum(
    doc["graded_relevance"] / math.log2(doc["rank"])
    for doc in ideal_ranking[1:5])
ndcg_5 = dcg_5 / idcg_5

# (l) 还有其他评估指标可用于评估排名的性能，如F1得分、AP@K和NDCG@K。
```

根据问题代入数据，有

```
 qiu_nangong@Somebodys-MacBook-Pro ▶ python3 Exercise-2.py
(a) P@5: 0.8
(a) P@10: 0.7
(b) R@5: 0.5714285714285714
(b) R@10: 1.0
(c) 最大化P@5的示例排名: [51, 501, 75, 321, 38]
(d) 最大化P@10的示例排名: [51, 501, 75, 321, 38, 412, 101, 21, 521, 331]
(e) 最大化R@5的示例排名: [51, 501, 21, 75, 321]
(f) 最大化R@10的示例排名: [51, 501, 21, 75, 321, 38, 521, 412, 331, 101]
(h) 平均准确率 (AP) : 0.8333333333333333
(i) 最大化AP的示例排名: [51, 321, 75, 101, 501, 38, 412, 21, 521, 331]
(j) DCG5: 8.222706232293572
(k) NDCG5: 0.7261651480106516
```

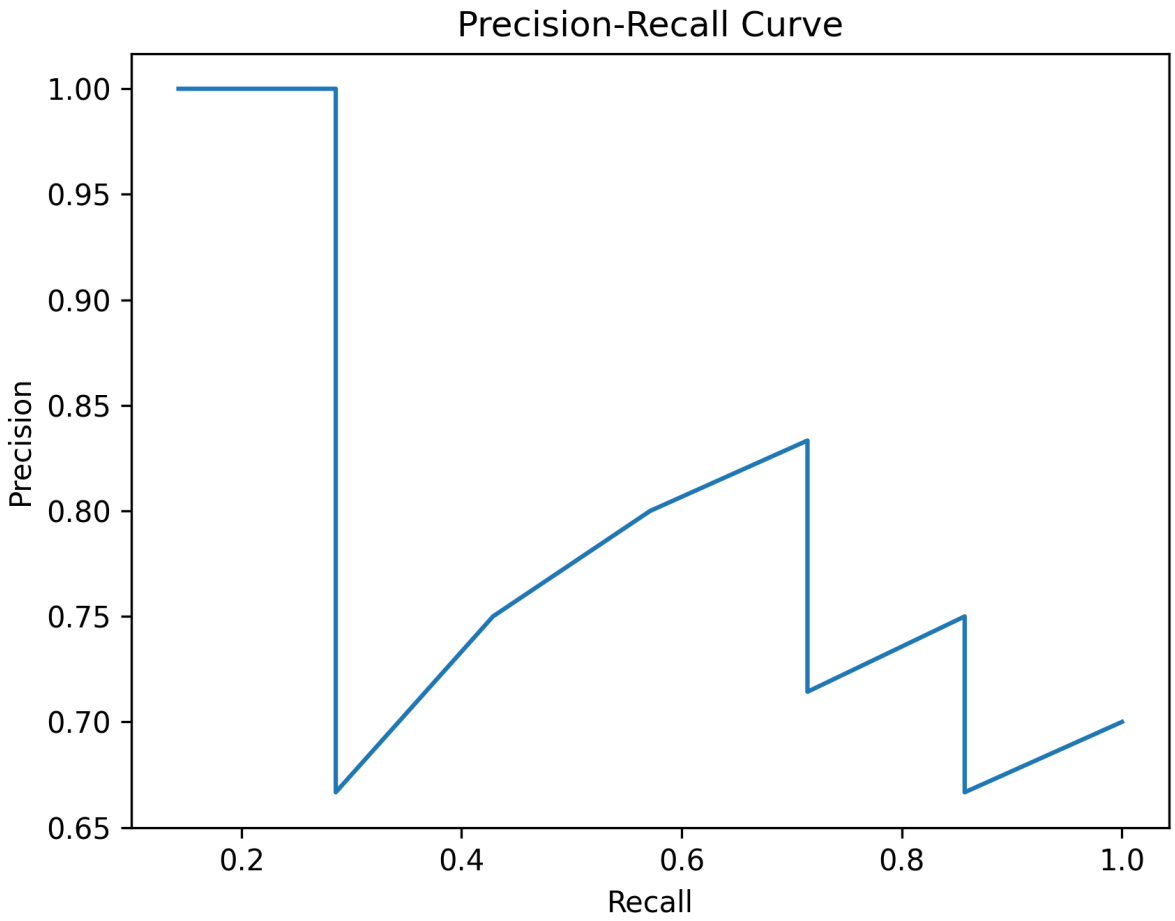
Exercise. 3

根据公式可写出计算代码

```
precision = []
recall = []
relevant_docs = sum(doc["binary_relevance"] for doc in ranking)
retrieved_docs = 0
relevant_retrieved_docs = 0

for doc in ranking:
    retrieved_docs += 1
    if doc["binary_relevance"] == 1:
        relevant_retrieved_docs += 1
    precision.append(relevant_retrieved_docs / retrieved_docs)
    recall.append(relevant_retrieved_docs / relevant_docs)
```

根据问题代入数据，有（完整代码见 Exercise-3.py）



分析可知，曲线图形符合数据。

Exercise. 4

除了在课堂中中提到的常见指标（如准确率、精确率、召回率、F1得分、AP、NDCG等），还有一些其他的评估指标可以用于性能评估。如：

1. 聚类任务：

- 轮廓系数（Silhouette Coefficient）：衡量聚类结果的紧密度和分离度。
- DB指数（Davies-Bouldin Index）：衡量聚类结果的紧密度和分离度，越小越好。
- Dunn指数（Dunn Index）：衡量聚类结果的紧密度和分离度，越大越好。

2. 分类任务：

- ROC曲线（Receiver Operating Characteristic Curve）：绘制真阳性率（True Positive Rate）和假阳性率（False Positive Rate）之间的关系曲线。
- AUC（Area Under the Curve）：计算ROC曲线下的面积，用于衡量分类器的性能。
- Cohen's Kappa系数：用于衡量分类器与人工标注之间的一致性。

3. 回归任务：

- R方（R-squared）：衡量回归模型对观测值变异的解释程度，越接近1越好。
- 均方误差（Mean Squared Error）：计算观测值与预测值之间的平均平方差，越小越好。

4. 关联规则挖掘任务：

- 支持度（Support）和置信度（Confidence）：用于衡量关联规则的频繁程度和可靠性。
- 提升度（Lift）：衡量关联规则中的项之间的相互依赖性。