Get unlimited access to the best of Medium for less than $1/week. Become a member

Ranking Evaluation Metrics for Recommender Systems

Various evaluation metrics are used for evaluating the effectiveness of a recommender. We will focus mostly on ranking related metrics covering HR (hit ratio), MRR (Mean Reciprocal Rank), MAP (Mean Average Precision), NDCG (Normalized Discounted Cumulative Gain).

Benjamin Wang · Follow
Published in Towards Data Science · 5 min read · Jan 18, 2021

224

Recommender systems eventually output a ranking list of items regardless of different modelling choices. So it is important to look at how to evaluate directly ranking quality instead of other proxy metrics like mean squared error, etc.

HR (Hit Ratio)

In recommender settings, the hit ratio is simply the fraction of users for which the correct answer is included in the recommendation list of length L.

$$HR = \frac{|U_{hit}^L|}{|U_{all}|}$$

where $|U_{hit}^L|$ is the number of users for which the correct answer is included in the top L recommendation list, $|U_{all}|$ is the total number of users in the test dataset.

As one can see, the larger *L* is, the higher hit ratio becomes, because there is a higher chance that the correct answer is included in the recommendation list. Therefore, it is important to choose a reasonable value for *L*.

MRR (Mean Reciprocal Rank)

MRR is short for *mean reciprocal rank.* It is also known as *average reciprocal hit ratio (ARHR).*

$$MRR = \frac{1}{|U_{all}|} \sum_{u=1}^{|U_{all}|} RR(u)$$

$$RR(u) = \sum_{i \leq L} \frac{relevance_i}{rank_i}$$

where $RR(u)$ is the reciprocal rank of a user u, and it is defined by the sum of relevance score of top L items weighted by reciprocal rank. MRR is simply the mean of all users in the test dataset.

Note that there are different variations or simplifications for calculating *RR(u)*. **For implicit dataset, the relevance score is either 0 or 1, for items not bought or bought (not clicked or clicked, etc.).**

Another simplification is only to look at **one top relevant** item in the recommendation list, instead of summing up for all. in case of implicit dataset, there is no ordering of relevance per se, it is sufficient just to consider any one relevant item on top of the list.

One could argue that hit ratio is actually a special case of MRR, when *RR(u)* is binary, as it becomes 1 if there is a relevant item in the list, 0 otherwise.

MAP (Mean Average Precision)

Let's first refresh our memory on precision and recall, especially in the Information Retrieval are.

What are precision and recall?

In short, precision is the fraction of relevant items in all the retrieved items. It is used to answer how many items among all recommendations are correct.

And recall is the fraction of relevant items in all relevant items. It is to answer the coverage question, among all those considered relevant items, how many are captured in the recommendations.

$$\text{precision} = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{retrieved documents}\}|}$$

Precision from Wikipedia

$$\text{recall} = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{relevant documents}\}|}$$

Recall from Wikipedia

What is precision@k?

Building upon it, we can also define **precison@k** and also **recall@k** similarly. Precision@k would be the fraction of relevant items in the top k recommendations, and recall@k would be the coverage of relevant times in the top k.

What is Mean Average Precision?

Now back to MAP.

MAP is the mean of *Average Precision.* If we have the AP for each user, it is trivial just to average it over all users to calculate the MAP.

By computing a precision and recall at every position in the ranked sequence of documents, one can plot a precision-recall curve, plotting precision *p(r)* as a function of recall *r*. Average precision computes the average value of *p(r)* over the interval from 0 to 1.

This is essential the area under the precision-recall curve. In a discrete manner, it can be calculated as follows

$$AP = \sum_{k=1}^{n} p(k) rel(k)$$

where $p(k)$ denotes precision@k, n is the number of items in the recommendation list, and $rel(k)$ is an indicator function which equals 1 if the rank k item is relevant, 0 otherwise.

Open in app ↗

Search Write

$$MAP = \frac{1}{|U_{all}|} \sum_{u=1}^{|U_{all}|} AP(u)$$

NDCG (Normalized Discounted Cumulative Gain)

NDCG stands for *normalized discounted cumulative gain.* We will build up this concept backwards answering the following questions:

- What is gain?
- What is cumulative gain?
- How to discount?
- How to normalize?

Gain for an item is essentially the same as the relevance score, which can be numerical ratings like search results in Google which can be rated in scale from 1 to 5, or binary in case of implicit data where we only know if a user has consumed certain item or not.

Naturally Cumulative Gain is defined as the sum of gains up to a position k in the recommendation list

$$CG(k) = \sum_{i=1}^{k} G_i$$

One obvious drawback of CG is that it does not take into account of **ordering**. By swapping the relative order of any two items, the CG would be unaffected. This is problematic when ranking order is important. For example, on Google Search results, you would obviously not like placing the most relevant web page at the bottom.

To penalize highly relevant items being placed at the bottom, we introduce the **DCG**

$$DCG(k) = \sum_{i=1}^{k} \frac{G_i}{log_2(i+1)}$$

By diving the gain by its rank, we sort of push the algorithm to place highly relevant items to the top to achieve the best DCG score.

There is still a drawback of DCG score. It is that DCG score adds up with the length of recommendation list. Therefore, we cannot consistently compare the DCG score for system recommending top 5 and top 10 items, because the latter will have higher score not because its recommendation quality but pure length.

We tackle this issue by introducing **IDCG (ideal DCG).** IDCG is the DCG score for the most ideal ranking, which is ranking the items top down according their relevance up to position k.

$$IDCG(k) = \sum_{i=1}^{|I(k)|} \frac{G_i}{log_2(i+1)}$$

where $I(k)$ represents the ideal list of items up to position k, $|I(k)| = k$

And NDCG is simply to normalize the DCG score by IDCG such that its value is always between 0 and 1 regardless of the length.

$$NDCG(k) = \frac{DCG(k)}{IDCG(k)}$$

Notes

1. Wikipedia on NDCG is pretty good: https://en.wikipedia.org/wiki/Discounted_cumulative_gain
2. Wikipedia has a very nice list of evaluation metrics used in IR: https://en.wikipedia.org/w/index.php?title=Information_retrieval&oldid=793358396#Average_precision
3. This article explains MAP very well both in IR and object detection: https://towardsdatascience.com/breaking-down-mean-average-precision-map-ae462f623a52

Recommendation System | Machine Learning | Artificial Intelligence | Mathematics

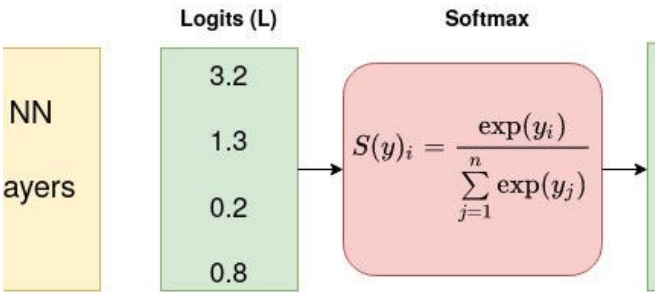
Written by Benjamin Wang

Follow

100 Followers · Writer for Towards Data Science

Machine Learning & Software Engineer in Amsterdam, Holland

More from Benjamin Wang and Towards Data Science



Benjamin Wang in The Startup

Cross Entropy Loss in PyTorch

A small tutorial for newbie using cross entropy loss in PyTorch.

4 min read · Jan 13, 2021

235

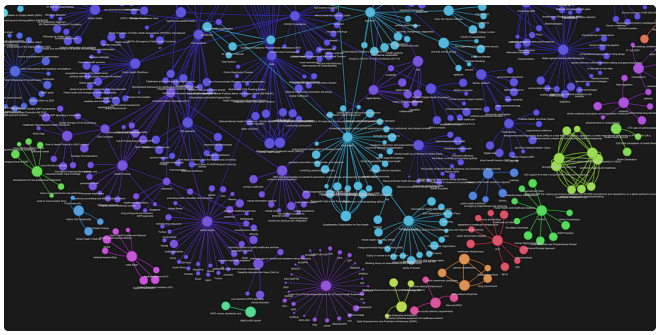
Marco Peixeiro in Towards Data Science

TimeGPT: The First Foundation Model for Time Series Forecasting

Explore the first generative pre-trained forecasting model and apply it in a project...

· 12 min read · Oct 24

2.2K 21



Rahul Nayak in Towards Data Science

How to Convert Any Text Into a Graph of Concepts

A method to convert any text corpus into a Knowledge Graph using Mistral 7B.

12 min read · Nov 10

1.2K 20

Benjamin Wang in Towards Data Science

Monte Carlo Tree Search: An Introduction

MCTS is the cornerstone of AlphaGo and many AI applications. We aim to build some...

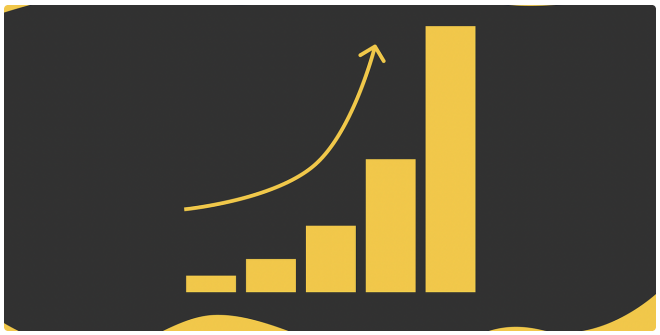
6 min read · Jan 10, 2021

190 1

See all from Benjamin Wang

See all from Towards Data Science

Recommended from Medium



Vyacheslav Efimov in Towards Data Science

Comprehensive Guide to Ranking Evaluation Metrics

Explore an abundant choice of metrics and find the best one for your problem

13 min read · Jul 29

205 6

Everton Gomede, PhD

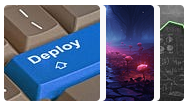
Understanding the BM25 Ranking Algorithm

Introduction

5 min read · May 27

98 1

Lists



Predictive Modeling w/ Python
20 stories · 600 saves

AI Regulation
6 stories · 186 saves

Natural Language Processing
856 stories · 398 saves

Practical Guides to Machine Learning
10 stories · 680 saves

Lakshma Reddy Induri

Evaluation Metrics for Search Ranking and Recommendation...

photo taken by me :)

Bharathi Priyaa

Part 2—How to Crack Machine learning Interviews at FAANG :...

In this article, I provide some some suggestions on how levels are calibrated an...

· 13 min read · Aug 7

11

8 min read · Jun 14

432 5

Eren Kızılırmak

Text Summarization: How To Calculate Rouge Score

This article is written by Eren Kızılırmak and Alparslan Mesri.

6 min read · Aug 2

18

Kushal Shah

How to find Sentence Similarity using Transformer Embeddings :...

A computer program can pretend to take text and images as input, and give text and imag...

5 min read · Sep 15

21

See more recommendations