NDCG and MRR

CS5154/6054

Yizong Cheng

9/29/2022

8 *Evaluation in information retrieval*

A final approach that has seen increasing adoption, especially when employed with machine learning approaches to ranking (see Section 15.4, page 341) is measures of *cumulative gain*, and in particular *normalized discounted cumulative gain* (*NDCG*). NDCG is designed for situations of non-binary notions of relevance (cf. Section 8.5.1). Like precision at k, it is evaluated over some number k of top search results. For a set of queries Q, let $R(j, d)$ be the relevance score assessors gave to document d for query j. Then,

CUMULATIVE GAIN
NORMALIZED DISCOUNTED CUMULATIVE GAIN
NDCG

(8.9)
$$\mathrm{NDCG}(Q,k) = \frac{1}{|Q|}\sum_{j=1}^{|Q|} Z_{kj} \sum_{m=1}^{k} \frac{2^{R(j,m)}-1}{\log_2(1+m)},$$

where Z_{kj} is a normalization factor calculated to make it so that a perfect ranking's NDCG at k for query j is 1. For queries for which $k' < k$ documents are retrieved, the last summation is done up to k'.

File Edit View History Bookmarks Tools Help

CS 276: Information Retrieval an

https://web.stanford.edu/class/cs276/

CS 276 / LING 286: Information Retrieval and Web Search

Image credit

Course Description

Information retrieval is the process through which a computer system can respond to a user's query for text-based information on a

File Edit View History Bookmarks Tools Help

CS 276: Information Retrieval an

https://web.stanford.edu/class/cs276/

	Tues. 4/23	PA2 release	**Programming assignment #2 released**	
	Thurs. 4/25	Lecture (Chris)	**Probabilistic IR: the binary independence model, BM25, BM25F** • Videos: "Vector Space Model" • Slides: PPT | PDF/6 | PDF/1	• IIR chapter 6 • IIR chapter 11
Week 5	Tues. 4/30	PS1 due	**Problem set #1 due**	
	Tues. 4/30	Lecture (Chris)	**Evaluation methods & NDCG** • Videos: "Result Summaries" • Slides: PPT | PDF/6 | PDF/1	• IIR chapter 8 • MG section 4.5 • MIR chapter 3
	Tues. 4/30	Ranking quiz release	**Ranking quiz released**	
	Thurs. 5/2	Lecture (Pandu)	**Systems issues in efficient retrieval and scoring**	• IIR chapter 6 • IIR chapter 7 • Efficient Query Evaluation using a

Introduction to

Information Retrieval

Evaluation

Chris Manning and Pandu Nayak

CS276 – Information Retrieval and Web Search

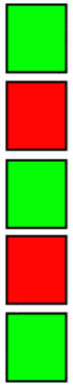
Rank-Based Measures

- Binary relevance
  - Precision@K (P@K)
  - Mean Average Precision (MAP)
  - Mean Reciprocal Rank (MRR)

- Multiple levels of relevance
  - Normalized Discounted Cumulative Gain (NDCG)

Precision@K

- Set a rank threshold K
- Compute % relevant in top K
- Ignores documents ranked lower than K
- Ex:
  - Prec@3 of 2/3
  - Prec@4 of 2/4
  - Prec@5 of 3/5



- In similar fashion we have Recall@K

```python
import numpy as np
from sklearn.metrics import precision_score
from sklearn.metrics import recall_score

y_true = [1, 0, 1, 1, 1, 1, 0, 0, 0, 1]
y_pred = np.zeros(len(y_true), dtype=int)
for i in range(len(y_true)):
    y_pred[i] = 1
    print(precision_score(y_true, y_pred), recall_score(y_true, y_pred))
```

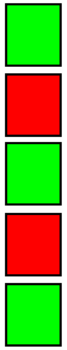
```
1.0 0.16666666666666666
0.5 0.16666666666666666
0.6666666666666666 0.3333333333333333
0.75 0.5
0.8 0.6666666666666666
0.8333333333333334 0.8333333333333334
0.7142857142857143 0.8333333333333334
0.625 0.8333333333333334
0.5555555555555556 0.8333333333333334
0.6 1.0
```

Mean Average Precision

- Consider rank position of each ***relevant*** doc
  - K_1, K_2, … K_R

- Compute Precision@K for each K_1, K_2, … K_R

- Average precision = average of P@K



- Ex: has AvgPrec of $$\frac{1}{3}\cdot\left(\frac{1}{1}+\frac{2}{3}+\frac{3}{5}\right)\approx 0.76$$

- MAP is Average Precision across multiple queries/rankings

File Edit View History Bookmarks Tools Help

sklearn.metrics.average_precisio

https://scikit-learn.org/stable/modules/generated/sklearn.metrics.average_precision 150%

scikit learn

sklearn.metrics.average_precision_score

`sklearn.metrics.`**`average_precision_score`**(*y_true, y_score, *, average='macro', pos_label=1, sample_weight=None*) [source]

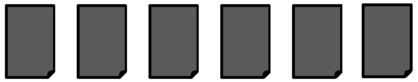
Compute average precision (AP) from prediction scores.

AP summarizes a precision-recall curve as the weighted mean of precisions achieved at each threshold, with the increase in recall from the previous threshold used as the weight:

$$\text{AP} = \sum_n (R_n - R_{n-1}) P_n$$

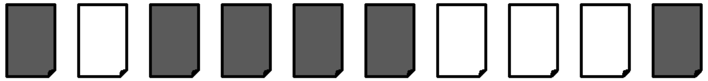
Toggle Menu

Average Precision



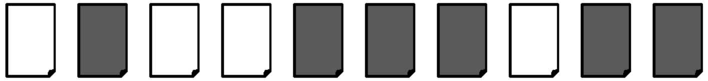
= the relevant documents

Ranking #1



Recall	0.17	0.17	0.33	0.5	0.67	0.83	0.83	0.83	0.83	1.0
Precision	1.0	0.5	0.67	0.75	0.8	0.83	0.71	0.63	0.56	0.6

Ranking #2

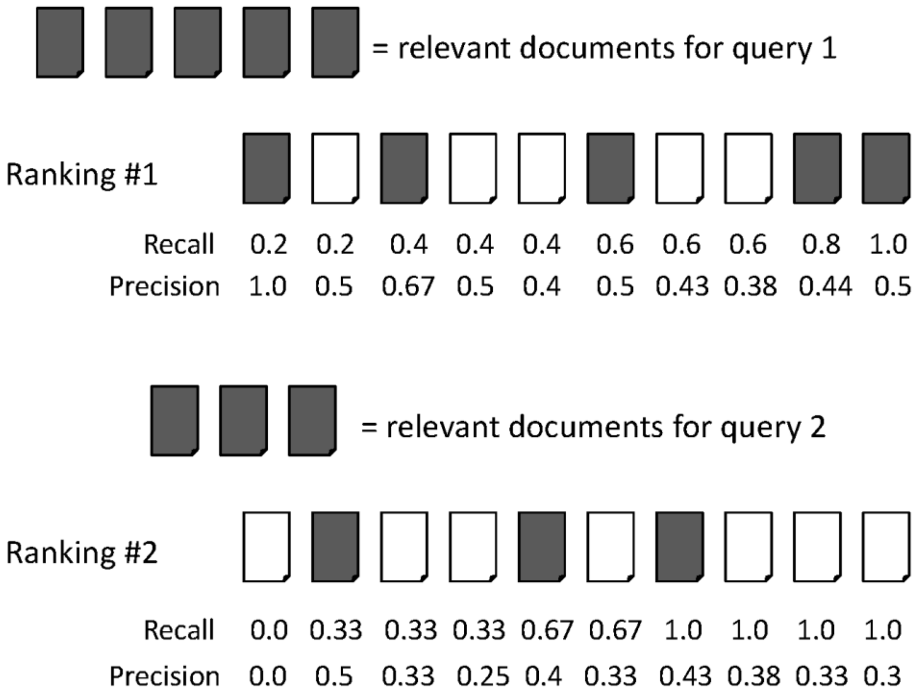


Recall	0.0	0.17	0.17	0.17	0.33	0.5	0.67	0.67	0.83	1.0
Precision	0.0	0.5	0.33	0.25	0.4	0.5	0.57	0.5	0.56	0.6

$$\text{Ranking } \#1\text{:}\ \ (1.0 + 0.67 + 0.75 + 0.8 + 0.83 + 0.6)/6 = 0.78$$

$$\text{Ranking } \#2\text{:}\ \ (0.5 + 0.4 + 0.5 + 0.57 + 0.56 + 0.6)/6 = 0.52$$

MAP



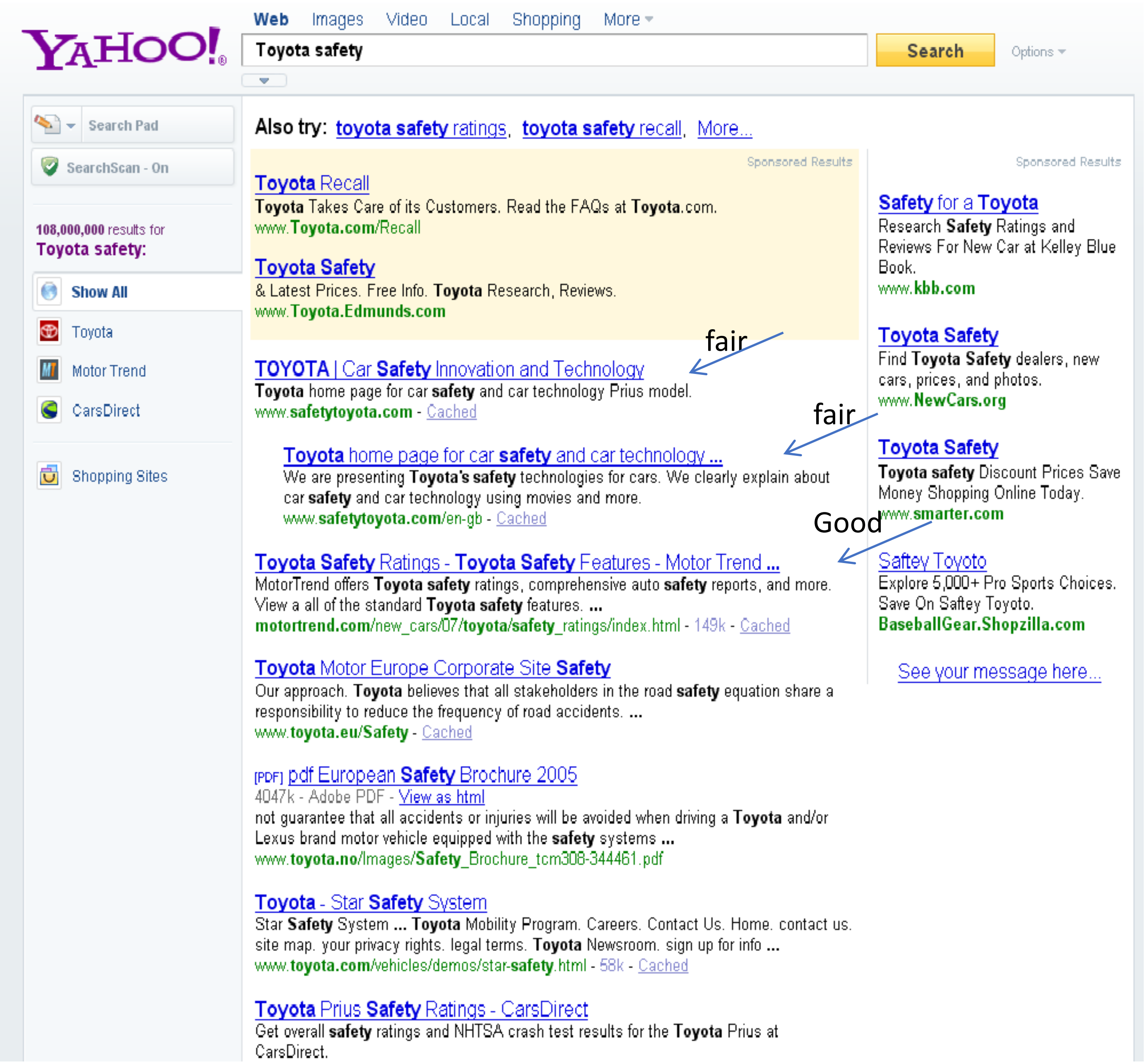
$$average\ precision\ query\ 1 = (1.0 + 0.67 + 0.5 + 0.44 + 0.5)/5 = 0.62$$
$$average\ precision\ query\ 2 = (0.5 + 0.4 + 0.43)/3 = 0.44$$

$$mean\ average\ precision = (0.62 + 0.44)/2 = 0.53$$

Mean average precision

- If a relevant document never gets retrieved, we assume the precision corresponding to that relevant doc to be zero
- MAP is macro-averaging: each query counts equally
- Now perhaps most commonly used measure in research papers
- Good for web search?
- MAP assumes user is interested in finding many relevant documents for each query
- MAP requires many relevance judgments in text collection

Beyond binary relevance



YAHOO!
Web Images Video Local Shopping More
Toyota safety
Search
Options
Search Pad
SearchScan - On
108,000,000 results for
Toyota safety:
Show All
Toyota
Motor Trend
CarsDirect
Shopping Sites
Also try: toyota safety ratings, toyota safety recall, More...
Sponsored Results
Toyota Recall
Toyota Takes Care of its Customers. Read the FAQs at Toyota.com.
www.Toyota.com/Recall
Toyota Safety
& Latest Prices. Free Info. Toyota Research, Reviews.
www.Toyota.Edmunds.com
fair
TOYOTA | Car Safety Innovation and Technology
Toyota home page for car safety and car technology Prius model.
www.safetytoyota.com - Cached
fair
Toyota home page for car safety and car technology ...
We are presenting Toyota's safety technologies for cars. We clearly explain about car safety and car technology using movies and more.
www.safetytoyota.com/en-gb - Cached
Good
Toyota Safety Ratings - Toyota Safety Features - Motor Trend ...
MotorTrend offers Toyota safety ratings, comprehensive auto safety reports, and more. View a all of the standard Toyota safety features. ...
motortrend.com/new_cars/07/toyota/safety_ratings/index.html - 149k - Cached
Toyota Motor Europe Corporate Site Safety
Our approach. Toyota believes that all stakeholders in the road safety equation share a responsibility to reduce the frequency of road accidents. ...
www.toyota.eu/Safety - Cached
[PDF] pdf European Safety Brochure 2005
4047k - Adobe PDF - View as html
not guarantee that all accidents or injuries will be avoided when driving a Toyota and/or Lexus brand motor vehicle equipped with the safety systems ...
www.toyota.no/Images/Safety_Brochure_tcm308-344461.pdf
Toyota - Star Safety System
Star Safety System ... Toyota Mobility Program. Careers. Contact Us. Home. contact us. site map. your privacy rights. legal terms. Toyota Newsroom. sign up for info ...
www.toyota.com/vehicles/demos/star-safety.html - 58k - Cached
Toyota Prius Safety Ratings - CarsDirect
Get overall safety ratings and NHTSA crash test results for the Toyota Prius at CarsDirect.
Sponsored Results
Safety for a Toyota
Research Safety Ratings and Reviews For New Car at Kelley Blue Book.
www.kbb.com
Toyota Safety
Find Toyota Safety dealers, new cars, prices, and photos.
www.NewCars.org
Toyota Safety
Toyota safety Discount Prices Save Money Shopping Online Today.
www.smarter.com
Saftey Toyoto
Explore 5,000+ Pro Sports Choices. Save On Saftey Toyoto.
BaseballGear.Shopzilla.com
See your message here...

Discounted Cumulative Gain

- Popular measure for evaluating web search and related tasks

- Two assumptions:
    - Highly relevant documents are more useful than marginally relevant documents
    - the lower the ranked position of a relevant document, the less useful it is for the user, since it is less likely to be examined

Discounted Cumulative Gain

- Uses *graded relevance* as a measure of usefulness, or *gain,* from examining a document
- Gain is accumulated starting at the top of the ranking and may be reduced, or *discounted*, at lower ranks
- Typical discount is 1/log *(rank)*
  - With base 2, the discount at rank 4 is 1/2, and at rank 8 it is 1/3

Summarize a Ranking: DCG

- What if relevance judgments are in a scale of [0,r]? r>2
- Cumulative Gain (CG) at rank n
    - Let the ratings of the n documents be r_1, r_2, …r_n (in ranked order)
    - CG = $r_1+r_2+\ldots r_n$
    - Normalized CG for binary relevance scale is precision.
- Discounted Cumulative Gain (DCG) at rank n
    - DCG = $r_1 + r_2/\log_2 2 + r_3/\log_2 3 + \ldots\ r_n/\log_2 n$
        - We may use any base for the logarithm

Discounted Cumulative Gain

- *DCG* is the total gain accumulated at a particular rank *p*:

$$DCG_p = rel_1 + \sum_{i=2}^{p} \frac{rel_i}{\log_2 i}$$

- Alternative formulation (used in iir):

$$DCG_p = \sum_{i=1}^{p} \frac{2^{rel_i} - 1}{log(1+i)}$$

  - used by some web search companies
  - emphasis on retrieving highly relevant documents
  - When relevance is binary, the two formulations merge.
    - rel_i is either 0 or 1 and 2^rel_i – 1 is also 0 or 1

DCG Example

- 10 ranked documents judged on 0–3 relevance scale:

  3, 2, 3, 0, 0, 1, 2, 2, 3, 0

  Or binary 1, 1, 1, 0, 0, 1, 1, 1, 1, 0

  CG: 1, 2, 3, 3, 3, 4, 5, 6, 7, 7 (tp?)

- discounted gain:

  3, 2/1, 3/1.59, 0, 0, 1/2.59, 2/2.81, 2/3, 3/3.17, 0

  = 3, 2, 1.89, 0, 0, 0.39, 0.71, 0.67, 0.95, 0

  Or 1, 1, 1/1.59, 0, 0, 1/2.59, 1/2.81, 1/3, 1/3.17, 0

- DCG:

  3, 5, 6.89, 6.89, 6.89, 7.28, 7.99, 8.66, 9.61, 9.61

scikit learn

sklearn.metrics.dcg_score

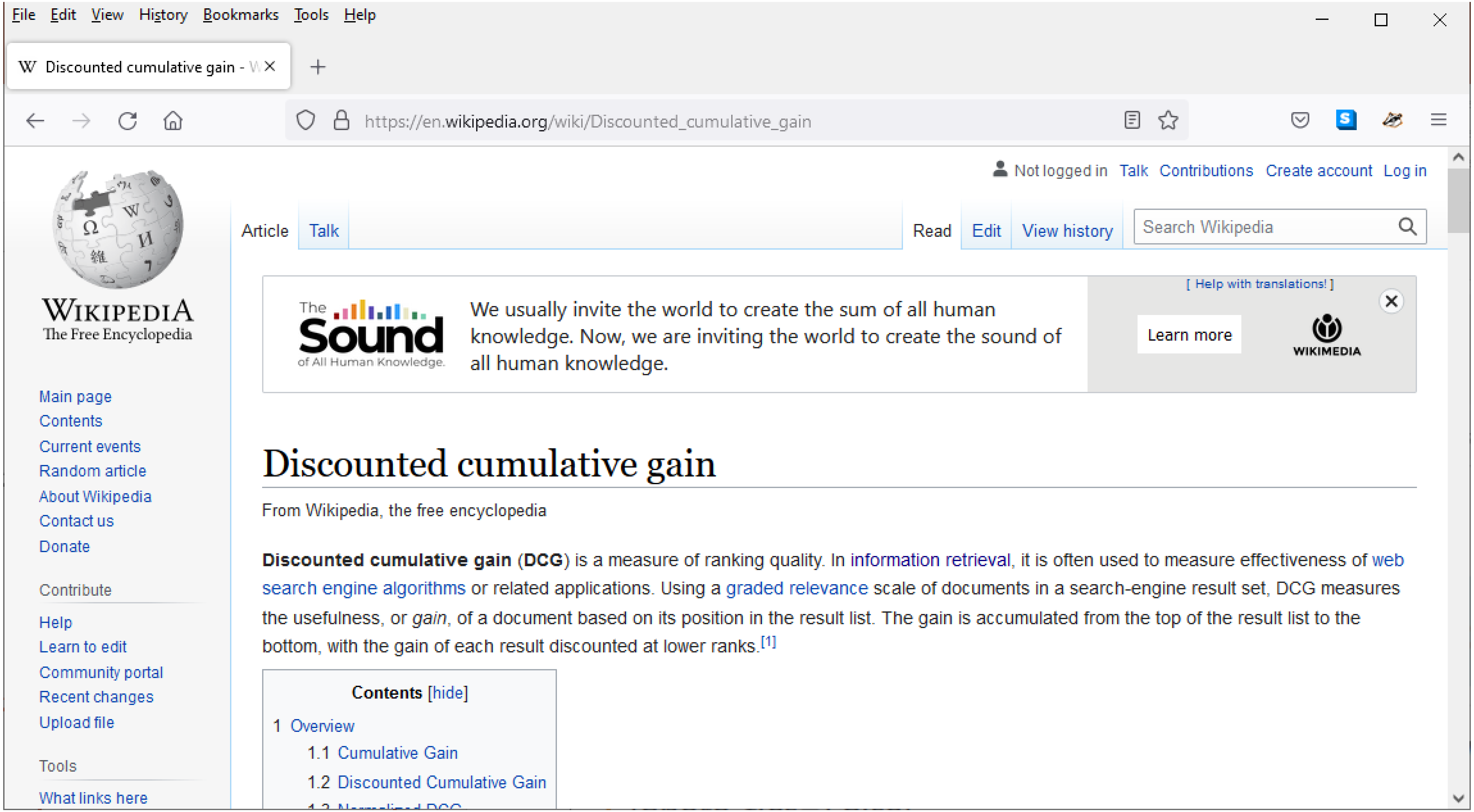
`sklearn.metrics.dcg_score`(*y_true, y_score, *, k=None, log_base=2, sample_weight=None, ignore_ties=False*) [source]

Compute Discounted Cumulative Gain.

Sum the true scores ranked in the order induced by the predicted scores, after applying a logarithmic discount.

This ranking metric yields a high value if true labels are ranked high by `y_score`.

Usually the Normalized Discounted Cumulative Gain (NDCG, computed by ndcg_score) is preferred.


File Edit View History Bookmarks Tools Help
Discounted cumulative gain -
https://en.wikipedia.org/wiki/Discounted_cumulative_gain
Not logged in Talk Contributions Create account Log in
Article Talk
Read Edit View history
Search Wikipedia
WIKIPEDIA
The Free Encyclopedia
The Sound of All Human Knowledge.
We usually invite the world to create the sum of all human knowledge. Now, we are inviting the world to create the sound of all human knowledge.
[Help with translations!]
Learn more
WIKIMEDIA
Main page
Contents
Current events
Random article
About Wikipedia
Contact us
Donate
Contribute
Help
Learn to edit
Community portal
Recent changes
Upload file
Tools
What links here
Discounted cumulative gain
From Wikipedia, the free encyclopedia
Discounted cumulative gain (DCG) is a measure of ranking quality. In information retrieval, it is often used to measure effectiveness of web search engine algorithms or related applications. Using a graded relevance scale of documents in a search-engine result set, DCG measures the usefulness, or gain, of a document based on its position in the result list. The gain is accumulated from the top of the result list to the bottom, with the gain of each result discounted at lower ranks.[1]
Contents [hide]
1 Overview
1.1 Cumulative Gain
1.2 Discounted Cumulative Gain

Print/export

Download as PDF

Printable version

Languages

Add links

Overview [edit]

Two assumptions are made in using DCG and its related measures.

1. Highly relevant documents are more useful when appearing earlier in a search engine result list (have higher ranks)
2. Highly relevant documents are more useful than marginally relevant documents, which are in turn more useful than non-relevant documents.

DCG originates from an earlier, more primitive, measure called Cumulative Gain.

Cumulative Gain [edit]

Cumulative Gain (CG) is the sum of the graded relevance values of all results in a search result list. This predecessor of DCG does not include the rank (position) of a result in the result list into the consideration of the usefulness of a result set. The CG at a particular rank position p is defined as:

$$\mathrm{CG_p} = \sum_{i=1}^{p} rel_i$$

Where rel_i is the graded relevance of the result at position i.

The value computed with the CG function is unaffected by changes in the ordering of search results. That is, moving a highly relevant document d_i above a higher ranked, less relevant, document d_j does not change the computed value for CG (assuming $i, j \leq p$). Based on the two assumptions made above about the usefulness of search results, (N)DCG is usually preferred over CG.

Cumulative Gain is sometimes called Graded Precision as it is identical to the Precision metric if the rating scale is binary.

File Edit View History Bookmarks Tools Help

W Discounted cumulative gain - W ×

https://en.wikipedia.org/wiki/Discounted_cumulative_gain

Discounted Cumulative Gain [edit]

The premise of DCG is that highly relevant documents appearing lower in a search result list should be penalized as the graded relevance value is reduced logarithmically proportional to the position of the result.

The traditional formula of DCG accumulated at a particular rank position p is defined as:[1]

$$\mathrm{DCG_p} = \sum_{i=1}^{p} \frac{rel_i}{\log_2(i+1)} = rel_1 + \sum_{i=2}^{p} \frac{rel_i}{\log_2(i+1)}$$

Previously there was no theoretically sound justification for using a logarithmic reduction factor[2] other than the fact that it produces a smooth reduction. But Wang et al. (2013)[3] gave theoretical guarantee for using the logarithmic reduction factor in Normalized DCG (NDCG). The authors show that for every pair of substantially different ranking functions, the NDCG can decide which one is better in a consistent manner.

An alternative formulation of DCG[4] places stronger emphasis on retrieving relevant documents:

$$\mathrm{DCG_p} = \sum_{i=1}^{p} \frac{2^{rel_i} - 1}{\log_2(i+1)}$$

The latter formula is commonly used in industry including major web search companies[5] and data science competition platforms such as Kaggle.[6]

These two formulations of DCG are the same when the relevance values of documents are binary;[2]:320 $rel_i \in \{0,1\}$.

Note that Croft et al. (2010) and Burges et al. (2005) present the second DCG with a log of base e, while both versions of DCG above use a log of base 2. When computing NDCG with the first formulation of DCG, the base of the log does not matter, but the base of the log does affect the value of NDCG for the second formulation. Clearly, the base of the log affects the value of DCG in both formulations.

NDCG for summarizing rankings

- Normalized Discounted Cumulative Gain (NDCG) at rank *n*
  - Normalize DCG at rank *n* by the DCG value at rank *n* of the ideal ranking
  - The ideal ranking would first return the documents with the highest relevance level, then the next highest relevance level, etc
- Normalization useful for contrasting queries with varying numbers of relevant results

- NDCG is now quite popular in evaluating Web search

NDCG - Example

4 documents: d_1, d_2, d_3, d_4

i	Ground Truth		Ranking Function$_1$		Ranking Function$_2$	
	Document Order	r_i	Document Order	r_i	Document Order	r_i
1	d4	2	d3	2	d3	2
2	d3	2	d4	2	d2	1
3	d2	1	d2	1	d4	2
4	d1	0	d1	0	d1	0
	$NDCG_{GT}$=1.00		$NDCG_{RF1}$=1.00		$NDCG_{RF2}$=0.9203	

$$DCG_{GT} = 2 + \left(\frac{2}{\log_2 2} + \frac{1}{\log_2 3} + \frac{0}{\log_2 4} \right) = 4.6309$$

$$DCG_{RF1} = 2 + \left(\frac{2}{\log_2 2} + \frac{1}{\log_2 3} + \frac{0}{\log_2 4} \right) = 4.6309$$

$$DCG_{RF2} = 2 + \left(\frac{1}{\log_2 2} + \frac{2}{\log_2 3} + \frac{0}{\log_2 4} \right) = 4.2619$$

$$MaxDCG = DCG_{GT} = 4.6309$$

File Edit View History Bookmarks Tools Help

sklearn.metrics.ndcg_score — s

https://scikit-learn.org/stable/modules/generated/sklearn.metrics.ndcg_score.html# 150%

scikit learn

sklearn.metrics.ndcg_score

sklearn.metrics.ndcg_score(*y_true, y_score, *, k=None, sample_weight=None, ignore_ties=False*) [source]

Compute Normalized Discounted Cumulative Gain.

Sum the true scores ranked in the order induced by the predicted scores, after applying a logarithmic discount. Then divide by the best possible score (Ideal DCG, obtained for a perfect ranking) to obtain a score between 0 and 1.

Toggle Menu metric returns a high value if true labels are ranked high by `y_score`.

Normalized DCG [edit]

This section **needs additional citations for verification**. Please help improve this article by adding citations to reliable sources. Unsourced material may be challenged and removed. *(February 2020) (Learn how and when to remove this template message)*

Search result lists vary in length depending on the query. Comparing a search engine's performance from one query to the next cannot be consistently achieved using DCG alone, so the cumulative gain at each position for a chosen value of p should be normalized across queries. This is done by sorting all **relevant** documents in the corpus by their relative relevance, producing the maximum possible DCG through position p, also called Ideal DCG (IDCG) through that position. For a query, the *normalized discounted cumulative gain*, or nDCG, is computed as:

$$\mathrm{nDCG_p} = \frac{DCG_p}{IDCG_p},$$

where IDCG is ideal discounted cumulative gain,

$$\mathrm{IDCG_p} = \sum_{i=1}^{|REL_p|} \frac{rel_i}{\log_2(i+1)}$$

and REL_p represents the list of relevant documents (ordered by their relevance) in the corpus up to position p.

The nDCG values for all queries can be averaged to obtain a measure of the average performance of a search engine's ranking algorithm. Note that in a perfect ranking algorithm, the DCG_p will be the same as the $IDCG_p$ producing an nDCG of 1.0. All nDCG calculations are then relative values on the interval 0.0 to 1.0 and so are cross-query comparable.

The main difficulty encountered in using nDCG is the unavailability of an ideal ordering of results when only partial relevance feedback is available.

Example

Presented with a list of documents in response to a search query, an experiment participant is asked to judge the relevance of each document to the query. Each document is to be judged on a scale of 0-3 with 0 meaning not relevant, 3 meaning highly relevant, and 1 and 2 meaning "somewhere in between". For the documents ordered by the ranking algorithm as

$$D_1, D_2, D_3, D_4, D_5, D_6$$

the user provides the following relevance scores:

$$3, 2, 3, 0, 1, 2$$

That is: document 1 has a relevance of 3, document 2 has a relevance of 2, etc. The Cumulative Gain of this search result listing is:

$$\mathrm{CG_6} = \sum_{i=1}^{6} rel_i = 3 + 2 + 3 + 0 + 1 + 2 = 11$$

Changing the order of any two documents does not affect the CG measure. If D_3 and D_4 are switched, the CG remains the same, 11. DCG is used to emphasize highly relevant documents appearing early in the result list. Using the logarithmic scale for reduction, the DCG for each result in order is:

i	rel_i	$\log_2(i+1)$	$\frac{rel_i}{\log_2(i+1)}$
1	3	1	3
2	2	1.585	1.262
3	3	2	1.5
4	0	2.322	0
5	1	2.585	0.387
6	2	2.807	0.712

So the DCG_6 of this ranking is:

$$\mathrm{DCG_6} = \sum_{i=1}^{6} \frac{rel_i}{\log_2(i+1)} = 3 + 1.262 + 1.5 + 0 + 0.387 + 0.712 = 6.861$$

Now a switch of D_3 and D_4 results in a reduced DCG because a less relevant document is placed higher in the ranking; that is, a more relevant document is discounted more by being placed in a lower rank.

```
# IR12A.py CS5154/6054 cheng 2022
# This is the example from sklearn.metrics.dcg_score
# Usage: IR12A.py

import numpy as np
from sklearn.metrics import dcg_score
# we have groud-truth relevance of some answers to a query:
true_relevance = np.asarray([[10, 0, 0, 1, 5]])
# we predict scores for the answers
scores = np.asarray([[.1, .2, .3, 4, 70]])
print(dcg_score(true_relevance, scores))
# we can set k to truncate the sum; only top k answers contribute
print(dcg_score(true_relevance, scores, k=2))
# now we have some ties in our prediction
scores = np.asarray([[1, 0, 0, 0, 1]])
# by default ties are averaged, so here we get the average true
# relevance of our top predictions: (10 + 5) / 2 = 7.5
print(dcg_score(true_relevance, scores, k=1))
# we can choose to ignore ties for faster results, but only
# if we know there aren't ties in our scores, otherwise we get
# wrong results:
print(dcg_score(true_relevance, scores, k=1, ignore_ties=True))
```

```
9.499457825916874
5.630929753571458
7.5
5.0
```

The performance of this query to another is incomparable in this form since the other query may have more results, resulting in a larger overall DCG which may not necessarily be better. In order to compare, the DCG values must be normalized.

To normalize DCG values, an ideal ordering for the given query is needed. For this example, that ordering would be the monotonically decreasing sort of all known relevance judgments. In addition to the six from this experiment, suppose we also know there is a document D_7 with relevance grade 3 to the same query and a document D_8 with relevance grade 2 to that query. Then the ideal ordering is:

$$3, 3, 3, 2, 2, 2, 1, 0$$

The ideal ranking is cut again to length 6 to match the depth of analysis of the ranking:

$$3, 3, 3, 2, 2, 2$$

The DCG of this ideal ordering, or *IDCG (Ideal DCG)* , is computed to rank 6:

$$\mathrm{IDCG_6} = 8.740$$

And so the nDCG for this query is given as:

$$\mathrm{nDCG_6} = \frac{DCG_6}{IDCG_6} = \frac{6.861}{8.740} = 0.785$$

What if the results are not in a list?

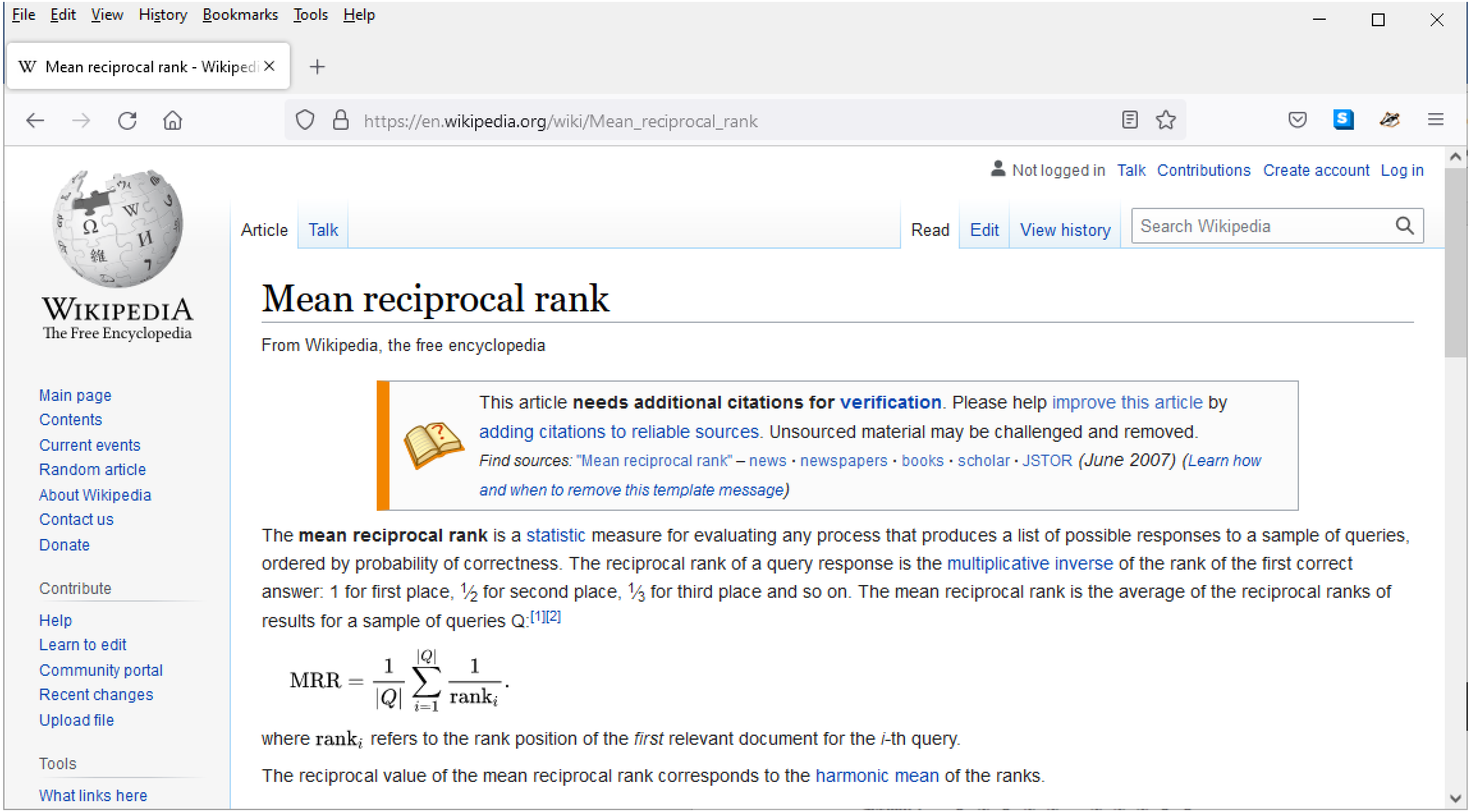
- Suppose there's only one Relevant Document
- Scenarios:
  - known-item search
  - navigational queries
  - looking for a fact
- Search duration ~ Rank of the answer
  - measures a user's effort

Mean Reciprocal Rank

- Consider rank position, K, of first relevant doc
  - Could be – only clicked doc

- Reciprocal Rank score = $\frac{1}{K}$

- MRR is the mean RR across multiple queries


File Edit View History Bookmarks Tools Help

Mean reciprocal rank - Wikipedi

https://en.wikipedia.org/wiki/Mean_reciprocal_rank

Not logged in Talk Contributions Create account Log in

Article Talk Read Edit View history Search Wikipedia

WIKIPEDIA
The Free Encyclopedia

Main page
Contents
Current events
Random article
About Wikipedia
Contact us
Donate

Contribute

Help
Learn to edit
Community portal
Recent changes
Upload file

Tools

What links here

Mean reciprocal rank

From Wikipedia, the free encyclopedia

This article **needs additional citations for verification**. Please help improve this article by adding citations to reliable sources. Unsourced material may be challenged and removed.
Find sources: "Mean reciprocal rank" – news · newspapers · books · scholar · JSTOR *(June 2007) (Learn how and when to remove this template message)*

The **mean reciprocal rank** is a statistic measure for evaluating any process that produces a list of possible responses to a sample of queries, ordered by probability of correctness. The reciprocal rank of a query response is the multiplicative inverse of the rank of the first correct answer: 1 for first place, ½ for second place, ⅓ for third place and so on. The mean reciprocal rank is the average of the reciprocal ranks of results for a sample of queries Q:[1][2]

$$\mathrm{MRR} = \frac{1}{|Q|}\sum_{i=1}^{|Q|}\frac{1}{\mathrm{rank}_i}.$$

where rank_i refers to the rank position of the *first* relevant document for the *i*-th query.

The reciprocal value of the mean reciprocal rank corresponds to the harmonic mean of the ranks.

File Edit View History Bookmarks Tools Help

W Mean reciprocal rank - Wikipedi ×

https://en.wikipedia.org/wiki/Mean_reciprocal_rank

What links here
Related changes
Special pages
Permanent link
Page information
Cite this page
Wikidata item

Print/export
Download as PDF
Printable version

Languages
Italiano
Русский
中文
Edit links

Example [edit]

For example, suppose we have the following three sample queries for a system that tries to translate English words to their plurals. In each case, the system makes three guesses, with the first one being the one it thinks is most likely correct:

Query	Proposed Results	Correct response	Rank	Reciprocal rank
cat	catten, cati, **cats**	cats	3	1/3
torus	torii, **tori**, toruses	tori	2	1/2
virus	**viruses**, virii, viri	viruses	1	1

Given those three samples, we could calculate the mean reciprocal rank as (1/3 + 1/2 + 1)/3 = 11/18 or about 0.61.

If none of the proposed results are correct, reciprocal rank is 0.[1] Note that only the rank of the first relevant answer is considered, possible further relevant answers are ignored. If users are interested also in further relevant items, mean average precision is a potential alternative metric.

sklearn.metrics.label_ranking_average_precision_score

`sklearn.metrics.label_ranking_average_precision_score`(*y_true*, *y_score*, *, *sample_weight=None*)

[source]

Compute ranking-based average precision.

Label ranking average precision (LRAP) is the average over each ground truth label assigned to each sample, of the ratio of true vs. total labels with lower score.

This metric is used in multilabel ranking problem, where the goal is to give better rank to the labels associated to each sample.

The obtained score is always strictly greater than 0 and the best value is 1.

n the User Guide.

3.3.3.2. Label ranking average precision

The `label_ranking_average_precision_score` function implements label ranking average precision (LRAP). This metric is linked to the `average_precision_score` function, but is based on the notion of label ranking instead of precision and recall.

Label ranking average precision (LRAP) averages over the samples the answer to the following question: for each ground truth label, what fraction of higher-ranked labels were true labels? This performance measure will be higher if you are able to give better rank to the labels associated with each sample. The obtained score is always strictly greater than 0, and the best value is 1. If there is exactly one relevant label per sample, label ranking average precision is equivalent to the mean reciprocal rank.

Formally, given a binary indicator matrix of the ground truth labels $y \in \{0, 1\}^{n_\text{samples} \times n_\text{labels}}$ and the score associated with each label $\hat{f} \in \mathbb{R}^{n_\text{samples} \times n_\text{labels}}$, the average precision is defined as

$$LRAP(y, \hat{f}) = \frac{1}{n_{\text{samples}}} \sum_{i=0}^{n_{\text{samples}}-1} \frac{1}{||y_i||_0} \sum_{j:y_{ij}=1} \frac{|\mathcal{L}_{ij}|}{\text{rank}_{ij}}$$

where $\mathcal{L}_{ij} = \left\{k: y_{ik} = 1, \hat{f}_{ik} \geq \hat{f}_{ij} \right\}$, $\text{rank}_{ij} = \left|\left\{k: \hat{f}_{ik} \geq \hat{f}_{ij} \right\}\right|$, $|\cdot|$ computes the cardinality of the set (i.e., elements in the set), and $||\cdot||_0$ is the ℓ_0 “norm” (which computes the number of nonzero elements

Toggle Menu

```python
# IR12C.py CS5154/6054 cheng 2022
# An example from sklearn User Guide 3.3.3.2 Label ranking average precision
# Usage: python IR12C.py

import numpy as np
from sklearn.metrics import label_ranking_average_precision_score
y_true = np.array([[1, 0, 0], [0, 0, 1]])
y_score = np.array([[0.75, 0.5, 1], [1, 0.2, 0.1]])
print(label_ranking_average_precision_score(y_true, y_score))
```

```
0.41666666666666663
```

Relevance Feedback: The Whole Is Inferior to the Sum of Its Parts

FIANA RAIBER, Yahoo Research
OREN KURLAND, Technion — Israel Institute of Technology

ACM Transactions on Information Systems, Vol. 37, No. 4, Article 44. Publication date: October 2019.

Table 4. Comparison with Additional Baselines
Using RM3 as the Query Model

	ClueWeb		GOV2		ROBUST	
	MAP	NDCG	MAP	NDCG	MAP	NDCG
Passage	26.0_s	35.4_s^m	35.5_s^m	53.0_s^m	**28.9^m**	50.4^m
PredictM	26.7_s	37.3	36.3	54.7_s	27.9	48.3_s
ClustLTR	25.6_s^m	34.6_s^m	32.4_s^m	47.8_s^m	24.7_s^m	41.5_s^m
ClustMRF	27.2	37.3	36.3_s	55.4_s	27.7_s	48.3_s
ClustDrift	27.0^m	37.2_s	36.5_s	55.7	28.1_s	49.0_s^m
RM3	26.7_s	36.8_s	36.3_s	55.4_s	27.8_s	47.9_s
Select(RM3)	26.8_s	37.0	36.4	55.3	28.2_s	49.6_s^m
Subsets(RM3)	**27.3^m**	**38.0^m**	**36.7^m**	**56.2^m**	28.7^m	**50.8^m**

"*m*" and "*s*" mark statistically significant differences with RM3 (induced from all given relevant documents) and Subsets(RM3), respectively. Bold: best result in a column.

IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received April 13, 2022, accepted April 30, 2022, date of publication May 23, 2022, date of current version May 26, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3176894

Multitask Fine-Tuning for Passage Re-Ranking Using BM25 and Pseudo Relevance Feedback

MEOUNGJUN KIM[1] **AND YOUNGJOONG KO**[2]
[1]Department of Artificial Intelligence, Sungkyunkwan University, Suwon 16419, South Korea
[2]Department of Computer Science and Engineering, Sungkyunkwan University, Suwon 16419, South Korea

$$
\begin{aligned}
& BM25\,(Q, P) \\
& \quad = \sum_{i=1}^{n} IDF\,(t_i)\, \frac{TF\,(t_i, P) \cdot (k+1)}{TF\,(t_i, P) + k\{1 - b + b \cdot L(P)\}}
\end{aligned} \tag{1}
$$

$$
w_t = log \frac{p(1-q)}{q(1-p)} = log \frac{(r+0.5)(S-s+0.5)}{(R-r+0.5)(s+0.5)} \tag{2}
$$

TABLE 7. Comparisons between the proposed and other models in MS MARCO leaderboard using the total training set.

Model Base: $RoBERTa_{base}$ Large: $RoBERTa_{large}$	Dev. set MRR@10	Test set MRR@10
DUET V2 [27]	0.252	0.253
OpenMatch - ELECTRA Base [28]	0.352	0.344
OpenMatch - ELECTRA Large [28]	0.388	<u>0.376</u>
TF-Ranking Ensemble [19]	0.405	0.391
$LR_{Baseline}$ (Base)	0.345	-
$LR + MLM$ (Base)	0.382	-
$LR + WMLM + SE$ (Base)	0.383	**0.372**
$LR + WMLM + SE$ (Large)	0.389	**0.378**