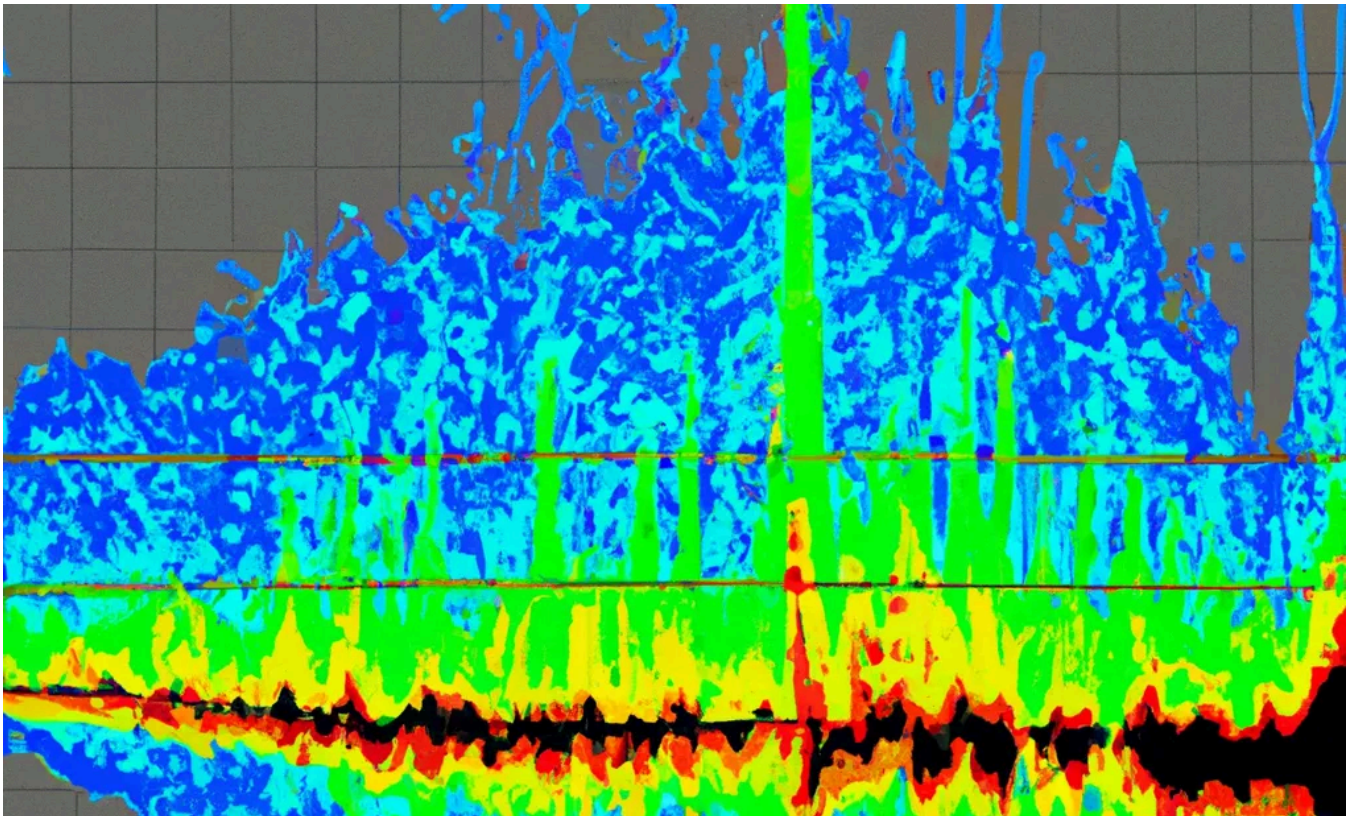

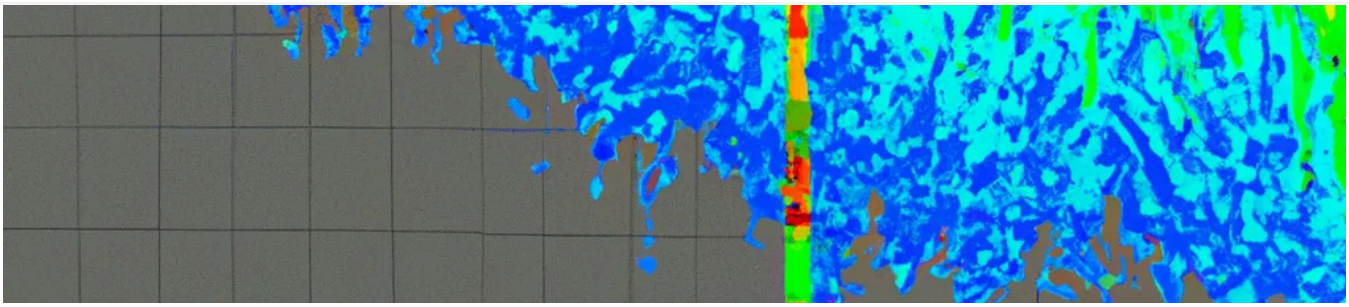

Image by author

Demystifying NDCG

How to best use this important metric for monitoring ranking models

Aparna Dhinakaran · Follow

Published in Towards Data Science

11 min read · Jan 25, 2023

This piece is co-authored by Jianshu Chi, Software Engineer at Arize AI, and Amber Roberts, ML engineer at Arize AI

Ranking models underpin many aspects of modern digital life, from search results to music recommendations. Anyone who has built a recommendation system understands the many challenges that come from developing and evaluating ranking models to serve their customers.

While these challenges start in data preparation and model training and continue through model development and model deployment, often what tends to give data scientists and machine learning engineers the most trouble is maintaining their ranking models in production. It is notoriously difficult to maintain models in production because of how these models are constantly changing as they adapt to dynamic environments.

In order to break down how to monitor **normalized discounted cumulative gain (NDCG)** for ranking models in production, this post covers:

- What is NDCG and where is it used?
- The intuition behind NDCG
- What is NDCG@K?
- How does NDCG compare to other metrics?
- How is NDCG used in model monitoring?

After tackling these main questions, your team will be able to achieve real time monitoring and root cause analysis using NDCG for ranking models in production.

What Is NDCG and Where Is It Used?

Normalized discounted cumulative gain is a measure of ranking quality. ML teams often use NDCG to evaluate the performance of a search engine, recommendation, or other information retrieval system. Search engines are popular for companies that have applications which directly interact with customers, like Alphabet, Amazon, Etsy, Netflix, and Spotify — just to name a few.

The value of NDCG is determined by comparing the relevance of the items returned by the search engine to the relevance of the item that a hypothetical "ideal" search engine would return. For example, if you search "Hero" on a popular music streaming app, you might get 10+ results with the word "Hero" in either the song, artist, or album.

The relevance of each song or artist is represented by a score (also known as a "grade") that is assigned to the search query. The scores of these recommendations are then discounted based on their position in the search results — did they get recommended first or last? The discounted scores are then cumulated and divided by the maximum possible discounted score, which is the discounted score that would be obtained if the search engine returned the documents in the order of their true relevance.

If a user wants the song "My Hero" by Foo Fighters, for example, the closer that song is to the top for the recommendation the better the search will be for that user. Ultimately, the relative order of returned results or recommendations is important for customer satisfaction.

Understanding the Intuition Behind Ranking Models

A ranking model predicts the ranks of a list of items based on the search queries made in the model. There are relevance scores assessed for each item based on relevancy of them within the search.

Here is a simple version dataset for a ranking model. There are two different search queries: **x** and **y**. Within each group, there are five different items shown as the result of search and each item has rank based on the position they are at the result list. Lastly, there are gains for each item representing the relevance of each item within the search.

search_group_id	item_id	Ranks	Gains (relevance score)
x	item_a	1	0
x	item_b	2	0
x	item_c	3	1
x	item_d	4	1
x	item_e	5	1
y	item_f	1	1
y	item_g	2	0
y	item_h	3	1
y	item_i	4	0
y	item_j	5	1

Image by author

It's hard to understand the intuition behind NDCG without diving into the meaning of each word in the name. So let's break it down…

1.) Cumulative Gain (CG)

Cumulative Gain is a sum of *gains* associated for items within a search query. Here is the formula for it:

$$CG = \sum_{i=1}^{ranks} Gains$$

Image by author

Using the dataset above, we can calculate CG for each group:

$$CG_x = 0 + 0 + 1 + 1 + 1 = 3$$

$$CG_y = 1 + 0 + 1 + 0 + 1 = 3$$

Image by author

In this example, both groups have the same CG — 3 — so we are still not able to tell which search groups are better. In order to do that, we need to take consideration of rank in the formula — which brings us into the next part: DCG.

2.) Discounted Cumulative Gain (DCG)

DCG is the same concept as CG but takes the additional step of discounting the gains by rank. Here is the formula for DCG:

$$DCG = \sum_{i=1}^{ranks} \frac{Gains}{log_2(i+1)}$$

Image by author

Using the dataset above, we can calculate DCG for each group:

$$DCG_x = \frac{0}{log_2(1+1)} + \frac{0}{log_2(2+1)} + \frac{1}{log_2(3+1)} + \frac{1}{log_2(4+1)} + \frac{1}{log_2(5+1)} \approx 1.31752$$

Image by author

$$DCG_y = \frac{1}{log_2(1+1)} + \frac{0}{log_2(2+1)} + \frac{1}{log_2(3+1)} + \frac{0}{log_2(4+1)} + \frac{1}{log_2(5+1)} \approx 1.88685$$

Image by author

Good news! Now we can see the DCG of y is better than the DCG of x. It also makes sense that group y has better DCG because the items in the higher rank are more relevant (higher gain) to the search group y. So why do we still need NDCG? To answer this question, let's introduce another search groups z to the count example:

search_group_id	item_id	Ranks	Gains *(relevance score)*
z	item_k	1	1
z	item_l	2	0
z	item_m	3	0
z	item_n	4	0
z	item_o	5	0

Image by author

Then, let's practice DCG calculation one more time:

$$DCG_z = \frac{1}{log_2(1+1)} + \frac{0}{log_2(2+1)} + \frac{0}{log_2(3+1)} + \frac{0}{log_2(4+1)} + \frac{0}{log_2(5+1)} = 1$$

Image by author

The DCG of z is 1, but it has the most relevant item at the first rank. If we compare the data, it should be at least better than group x. The problem is group x has three relevant items and group z only has one, and it's not fair to just compare the DCG since it's cumulative sum. This is how NDCG comes into play since it normalizes the DCG before comparing — but the problem is how to normalize it to make a fair comparison. For this task, we need IDCG.

3.) Ideal Discounted Cumulative Gain (IDCG)

IDCG stands for **ideal discounted cumulative gain**, which is calculating the DCG of the ideal order based on the gains. It answers the question: what is the best possible DCG for a group? Returning to a real-world example, when a user searches for something online they always want to have the most relevant item at the top and above any irrelevant items. That is, all the relevant information should always be at the top, and it should have the best DCG. Let's do more calculations, using IDCG for each search group from the dataset above:

$$IDCG_x = \frac{1}{log_2(1+1)} + \frac{1}{log_2(2+1)} + \frac{1}{log_2(3+1)} + \frac{0}{log_2(4+1)} + \frac{0}{log_2(5+1)} \approx 2.13093$$

$$IDCG_y = \frac{1}{log_2(1+1)} + \frac{1}{log_2(2+1)} + \frac{1}{log_2(3+1)} + \frac{0}{log_2(4+1)} + \frac{0}{log_2(5+1)} \approx 2.13093$$

$$IDCG_z = \frac{1}{log_2(1+1)} + \frac{0}{log_2(2+1)} + \frac{0}{log_2(3+1)} + \frac{0}{log_2(4+1)} + \frac{0}{log_2(5+1)} = 1$$

Image by author

4.) Normalized Discounted Cumulative Gain (NDCG)

Finally, we are done with heavy math and can finally use NDCG! NDCG normalizes the DCG by the IDCG of the group. It can be interpreted as the comparison of the actual relevance order and the ideal relevance order. NDCG is the quotient of DCG and IDCG; see the equations below.

$$NDCG = \frac{DCG}{IDCG}$$

Image by author

$$\mathrm{nDCG_k} = \frac{\mathrm{DCG_k}}{\mathrm{IDCG_k}}$$

$$\mathrm{DCG_k} = \sum_{i=1}^{k} \frac{\mathrm{rel}_i}{\log_2(i+1)} \qquad \mathrm{IDCG_k} = \sum_{i=1}^{k} \frac{\mathrm{rel}_i}{\log_2(i+1)}$$

Image by author

Returning to the dataset above, let's get the final NDCG for each group:

$$NDCG_x = \frac{DCG_x}{IDCG_x} = \frac{1.31752}{2.13093} = 0.61828$$

$$NDCG_y = \frac{DCG_y}{IDCG_y} = \frac{1.88685}{2.13093} = 0.88546$$

$$NDCG_z = \frac{DCG_z}{IDCG_z} = \frac{1}{1} = 1$$

Image by author

With this, we can confidently say group *z* has the best NDCG. It also makes sense that all its relevant items are at the top of the list. Finally, it is also worth noting that the NDCG range is between 0 and 1 and 1 is the max NDCG value.

What Is NDCG@K?

K means the top *K* ranked item of the list, and only top *K* relevance contributes to the final calculation. When we are calculating the NDCG@K, we first calculate the

DCG up to *K* items from the actual relevance order and ideal relevance order, then get the normalized DCG of that result.

Here is the formula for **NDCG@K**:

$$NDCG@K = \frac{DCG@K}{IDCG@K} = \frac{\sum_{i=1}^{k\ (actual\ order)} \frac{Gains}{log_2(i+1)}}{\sum_{i=1}^{k\ (ideal\ order)} \frac{Gains}{log_2(i+1)}}$$

Image by author

Now, let's calculate the NDCG@3 for the group **x**:

$$DCG@3_x = \frac{0}{log_2(1+1)} + \frac{0}{log_2(2+1)} + \frac{1}{log_2(3+1)} = 1.88685$$

$$IDCG@3_x = \frac{1}{log_2(1+1)} + \frac{1}{log_2(2+1)} + \frac{1}{log_2(3+1)} = 2.13093$$

$$NDCG@3_x = \frac{DCG_x}{IDCG_x} = \frac{0.5}{2.13093} \approx 0.23464$$

Image by author

How Does NDCG Compare To Other Ranking Metrics?

There are three prevailing metrics used by teams to evaluate the performance of search and rank engines, where the goal is to rank a list of items according to their relevance to a given query or user.

- **NDCG (normalized discounted cumulative gain)**: NDCG is a measure of the effectiveness of a ranking system, taking into account the position of relevant items in the ranked list. It is based on the idea that items that are higher in the ranking should be given more credit than items that are lower in the ranking. NDCG is calculated by dividing the discounted cumulative gain (DCG) of the ranked list by the DCG of the ideal ranked list, which is the list with the relevant items ranked in the most optimal order. *NDCG ranges from 0 to 1, with higher values indicating better performance.*

- **MAP (mean average precision)**: mean average precision is a measure of the precision of a ranking system, taking into account the number of relevant items in the ranked list. It is calculated by averaging the precision at each position in the ranked list, where precision is defined as the number of relevant items in the list up to that position divided by the total number of items in the list up to that position. *MAP ranges from 0 to 1, with higher values indicating better performance.*

- **MRR (mean reciprocal rank)**: MRR is a measure of the rank of the first relevant item in a ranked list. It is calculated by taking the reciprocal of the rank of the first relevant item, and averaging this value across all queries or users. For example, if the first relevant item for a given query has a rank of 3, the MRR for that query would be 1/3. *MRR ranges from 0 to 1, with higher values indicating better performance.*

NDCG is often used in information retrieval because it takes into account the relative order of the returned items in the search results. This is important because users often only look at the top few search results, so the relative order of the results can be more important than any absolute scores. That said, NDCG is similar to the ranking metric MAP but is more sensitive to rank order because it takes into account the position of relevant items in the ranked list. It is based on the idea that items that are higher in the ranking should be given more credit than items that are lower in the ranking.

NDCG provides the ability to fine-tune which ranks are more valuable than others, and account for a scale of relevancy scores (graded relevance). While NDCG overcomes the shortcomings of MAP, it is limited by actual data and partial feedback and thus requires a more manual data-cleaning process for an accurate calculation.

Each ranking metric measures different aspects of ranking performance and the choice of which metric to use will depend on the specific goals of the ranking system and the context in which it is being used.

How Is NDCG Used In Model Monitoring?

To recap, NDCG is a useful metric for evaluating how well ranking models perform and ensures the most relevant items are shown at the top of the search results in descending order.

If you are a machine learning engineer that builds a search engine to recommend relevant items, you want to be sure that the results you are achieving in the model development and experimental phase are similar to what you are seeing in production. However, it is often the case that ranking models as well as any information retrieval system will decay in performance over time. This is where model monitoring and ML observability become crucial to the ML lifecycle.

Returning to the example from the beginning of the post, imagine you enter the word “Hero” into the search bar of a music streaming app. If you use ML monitoring in the production workflow, the music streaming app can use NDCG to evaluate how well their search ranking model predicts the list of songs or artists when users make a search on their application with the relevance of “play.”

Other examples abound, from social media companies using NDCG to evaluate the relevancy of their recommended posts and videos to retailers using it to optimize product listings.

These companies can also use NDCG@1, NDCG@5 and NDCG@10 to evaluate the relevance of those recommendations and strength of their search engine. Companies using ML observability platforms, are even able to monitor their ranking models and search engines that have multiple relevance labels (full disclosure: I am co-fouunder of Arize AI). These can be used to generate the gain (relevance scores) based on whether the relevance target (positive class) matches one of the relevance labels. If teams are using NDCG@K for each search group, then you should average all of them to get a final NDCG. Averaging the NDCG of all of the relevant search queries that are predicted by a model gives teams a good understanding of how well the model performs.

The aggregate NDCG value for groups x, y, and z above is:

$$Average\ NDCG = (0.61828 + 0.88546 + 1) / 3 = 0.83458$$

Image by author

What Should You Do If Your Model Does Not Have Relevance Scores?

Relevance scores are required to compute NDCG. In the case that relevance scores are not available, you can generate a binary relevance score using an attribution model. This model could produce a *score = 1* if your prediction label, relevance label, and positive class match.

If a relevance score is not available for multi-label cases (such as ['click', 'favorite', 'buy']) and the positive class is 'click,' relevance will be attributed to sum([1,0,0]). Thus, it's important to attribute relevance scores when possible to compute a more precise NDCG for further troubleshooting.

What Does a Low NDCG Value In Production Mean?

The example below shows what happens when the performance of a recommendation system in production starts to decline. In the image inset, you may notice that the training and production datasets are nearly identical, with only the first and last recommendations switched in the production dataset. This results in a significant difference in the performance between the two datasets, dropping NDCG from 0.993 to 0.646. NDCG is the most sensitive rank-aware metric to overall graded order and is favorable for cases when you can receive full relevance feedback.

Image by author

Armed with this information, we can say our NDCG values have underperformed in production when needed to provide relevant search results to customers. Now that we know our model is starting to decay, we can start to uncover the where and why of our performance issues.

More Resources

To learn more about how to proactively catch performance degradation with an appropriate evaluation metric and then identify the worst performing features and slices and easily root cause model issues, check out our previous pieces on performance tracing and monitoring ranking models in production.