GBM Ranking (Light GBM)

Introduction

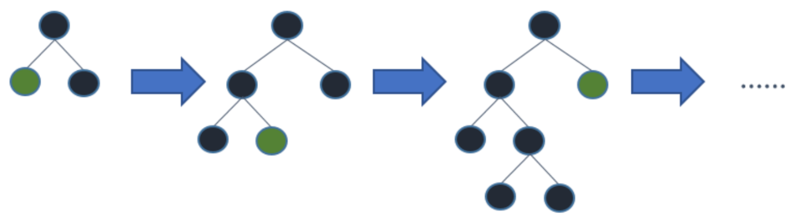
Over the past few years, ranking algorithms have been growing in popularity in many industries and companies are finding its merits in their application. The research focuses on lightGBM which is fast, distributed and high-performance gradient boosting tree-based learning model that ca be used in various applications including regression, classification, and ranking. As the name suggests, it is based on the gradient-based learning model that uses an ensemble model of decision trees.

Methodology

In each iteration, the algorithm learns from a decision tree based on residual error that brings such advantages as below:

- Faster training speed and higher efficiency
- Lower memory usage
- Better accuracy
- Support for parallel and GPU learning
- Capable of handling large-scale data (Big data)

This method splits the tree leaf wise whereas other boosting algorithms splits the tree depth or level wise. The leaf-wise algorithm can perform better than the level-wise algorithm that results in higher accuracy as a measurement.



Leaf-wise tree growth

Key parameters

- **num_leaves** *(default=31)* — Maximum tree leaves for base learners.
- **max_depth** *(default=-1)* — Maximum tree depth for base learners.
- **n_estimators** *(default=100)* — Number of boosted trees to fit.
- **objective** — 'regression' for LGBMRegressor, 'binary' or 'multiclass' for LGBMClassifier, 'lambdarank' for LGBM**Ranker**.
- **subsample** *(default=1.0)* — Subsample ratio of the training instance.
- **reg_alpha** *(default=0.0)* — L1 regularization term on weights.
- **reg_lambda** *(default=0.0)* — L2 regularization term on weights.

Key evaluation methods

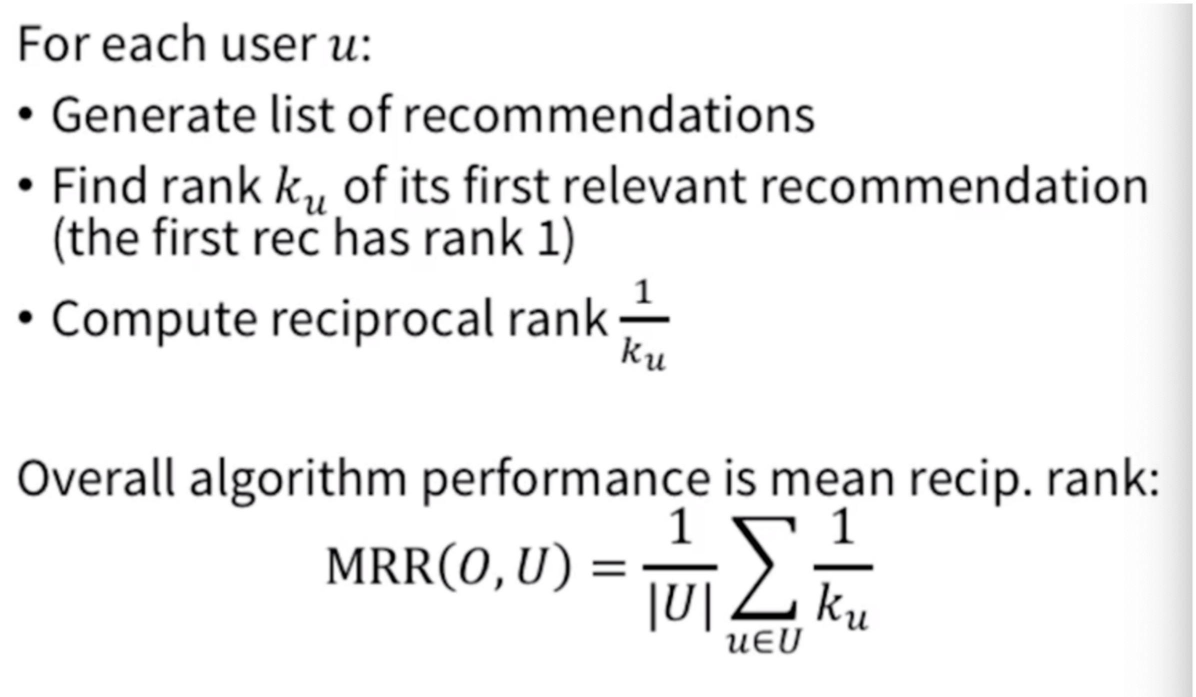
One of the most common ways of measuring accuracy is mean absolute error (MAE), and root mean square error (RMSE). To support such decision, decision support metrics are used which includes Precision, Recall and F1 score.

Rank-aware evaluation metrics consists of MRR (Mean Reciprocal Rank), MAP (Mean Average Precision), and NDCG (Normalized Discounted Cumulative Gain).

MRR: Mean Reciprocal Rank

A simplest metric of the tree that measures "where is the first relevant item?"
It is closely linked to the binary relevance family of metrics

Algorithm:



For each user u:

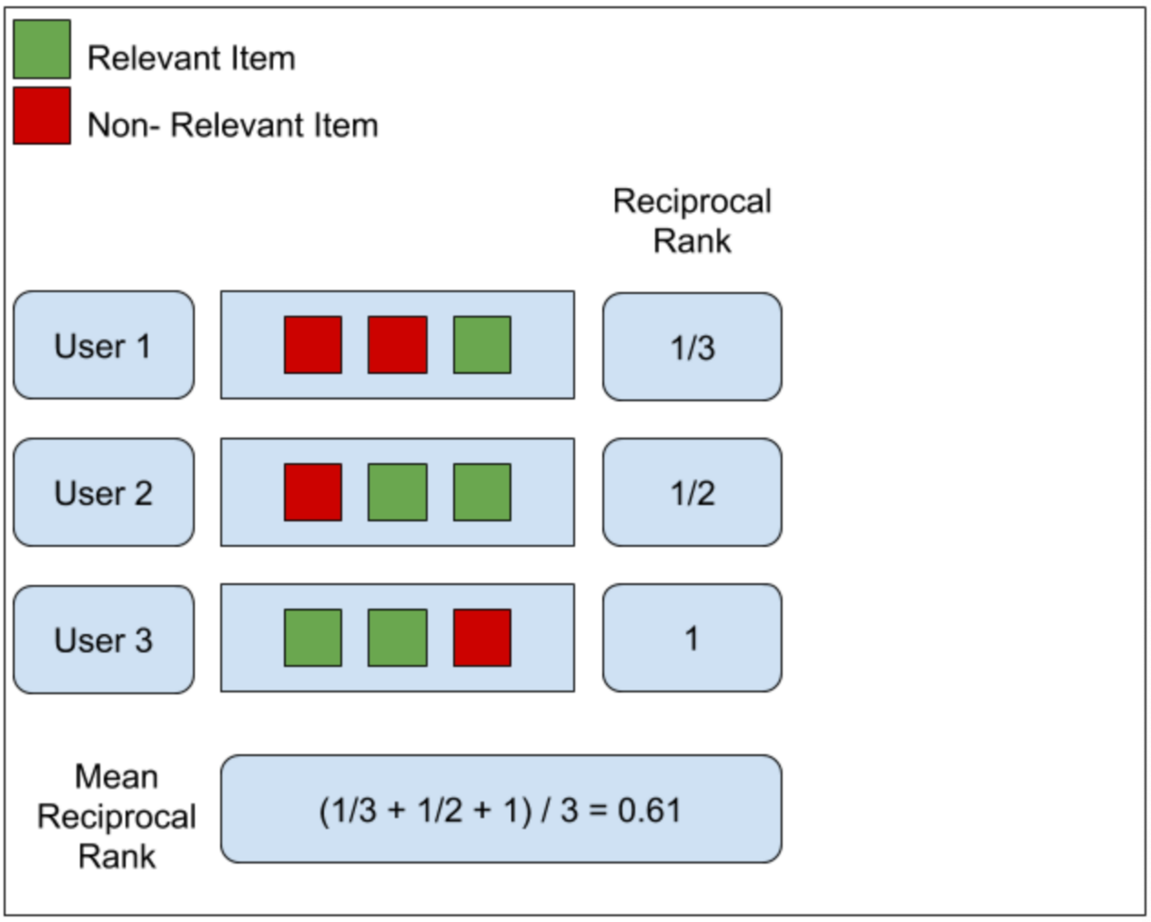
- Generate list of recommendations
- Find rank k_u of its first relevant recommendation (the first rec has rank 1)
- Compute reciprocal rank $\frac{1}{k_u}$

Overall algorithm performance is mean recip. rank:

$$\mathrm{MRR}(O, U) = \frac{1}{|U|} \sum_{u \in U} \frac{1}{k_u}$$

MRR metric calculation

Example



MRR Pros:

- Simple to compute and easy to interpret
- Puts a high focus on the first relevant element of the list. It is best suited for targeted searches such as users asking for the "best item for me"
- In Redback Operation, it can be used to determine the best exercise game or best curated cycling course of a map

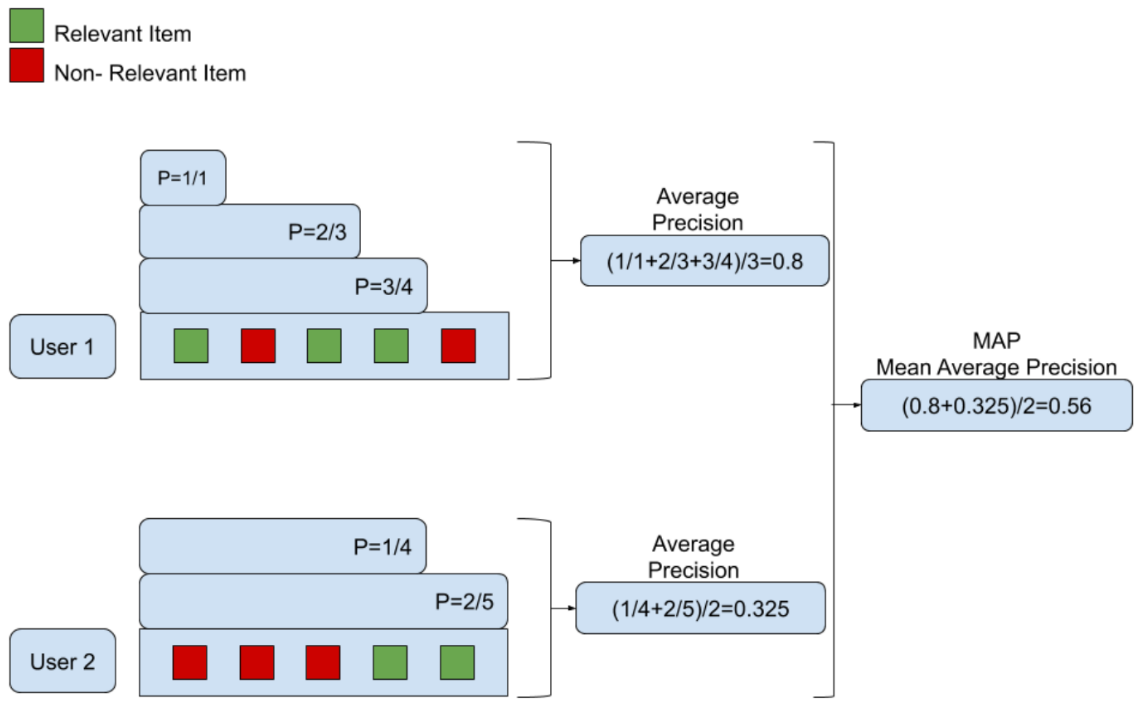
MRR Cons:

- Does not evaluate the rest of the list of recommended items. Focuses on a single item from the list
- Gives a list with a single relevant item just a much weight as a list with many relevant items.
- Might not be a good evaluation metric for users that want a list of related items to browse. The goal of the users might be to compare multiple related items.

MAP: Average Precision and Mean Average

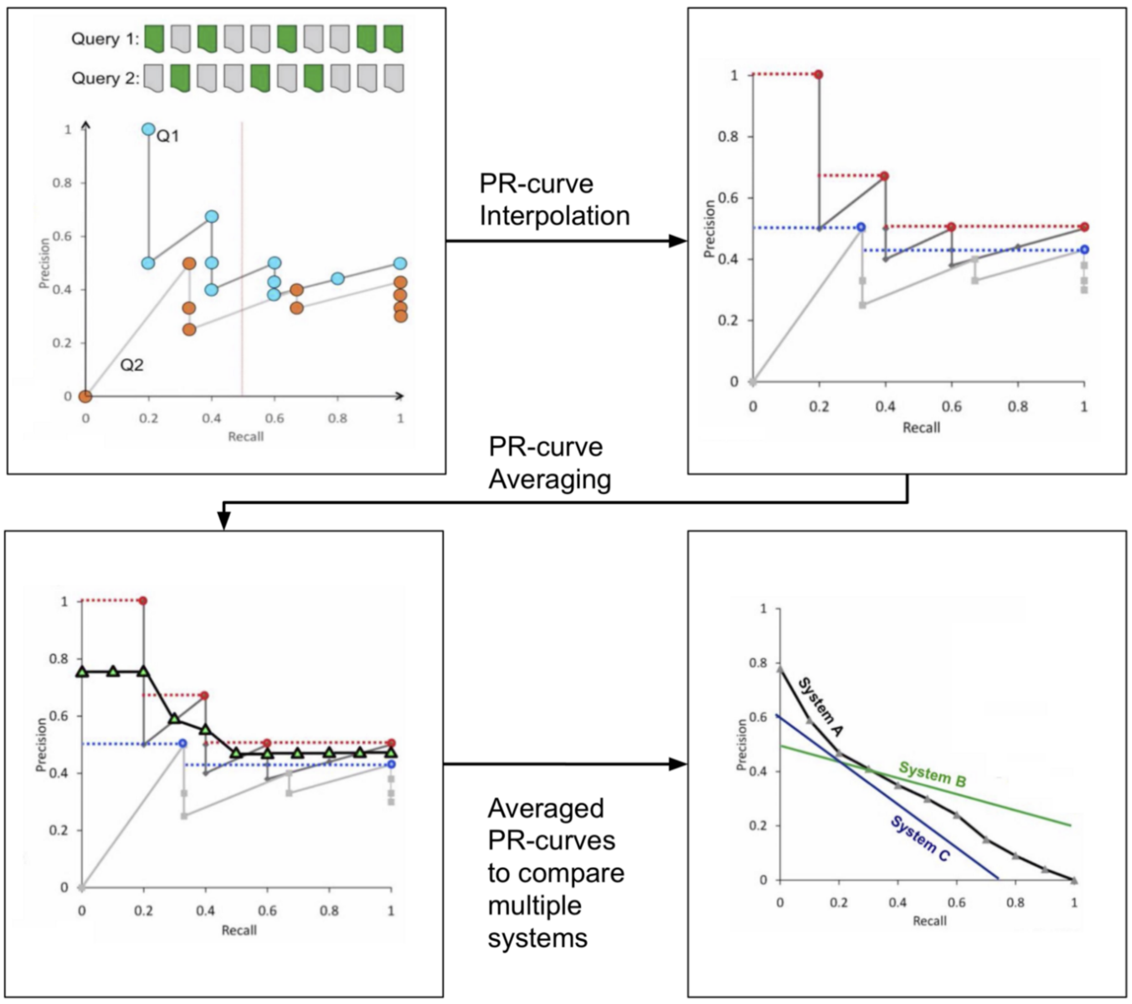
A good fit for a data set for evaluating the list of recommended items up to a specific cut-off N. One of the major drawbacks is this metric is not considering the recommended list as an ordered list. The goal of using this metric is to cut the erorr in the first few elements rather than much later in the list. Weight heavily the errors at the top of the list then gradually move down the significance of the errors as we go down the lower end of the item list. Hence, the average prediction metric tries to approximate this weighting scale by using a combination of precision at successive sub-lists, combined with the change in recall in these lists.

Algorithm:



Example MAP calculation

Average Precision metric is at the single recommendation list. Computing the precision through this item means sub-diving the recommendation list. This technique examines sub-list every time we get a relevant item. Then we calculate the precision on recommendations. Now that we have a set of precisions, we average them to get the average precision for a single user. Finally, we get the AP for all users and get the mean average precision.



Interpretation of the MAP measure through the area under the PR curve

First, it is crucial to look at the largest possible area under the PR curve. Looking at the systems A, B and C, it can be noticed that system A is better than C for all levels of recall. However, System A and C intersect where system B does better at higher levels of recall. In this scenario, it is hard to determine which system does better overall. Plots are harder to interpret than single metrics. This is the reason for using Average Precision.

MAP Pros

- Gives a single metric that represents the complex area under the precision-recall curve. This provides the average precision (AP) per list
- Handles ranking of lists recommended items naturally. This contrasts with metrics that considering the retrieved items as sets.
- This metric gives more weights to errors that happen higher up in the recommended lists. Conversely, it gives less weight to errors that happens deeper in the recommended lists.

MAP Cons

- This metrics shines for binary rating. It is not a good fit for a fine-grained numerical rating
- Unable to extract an error measure from this information

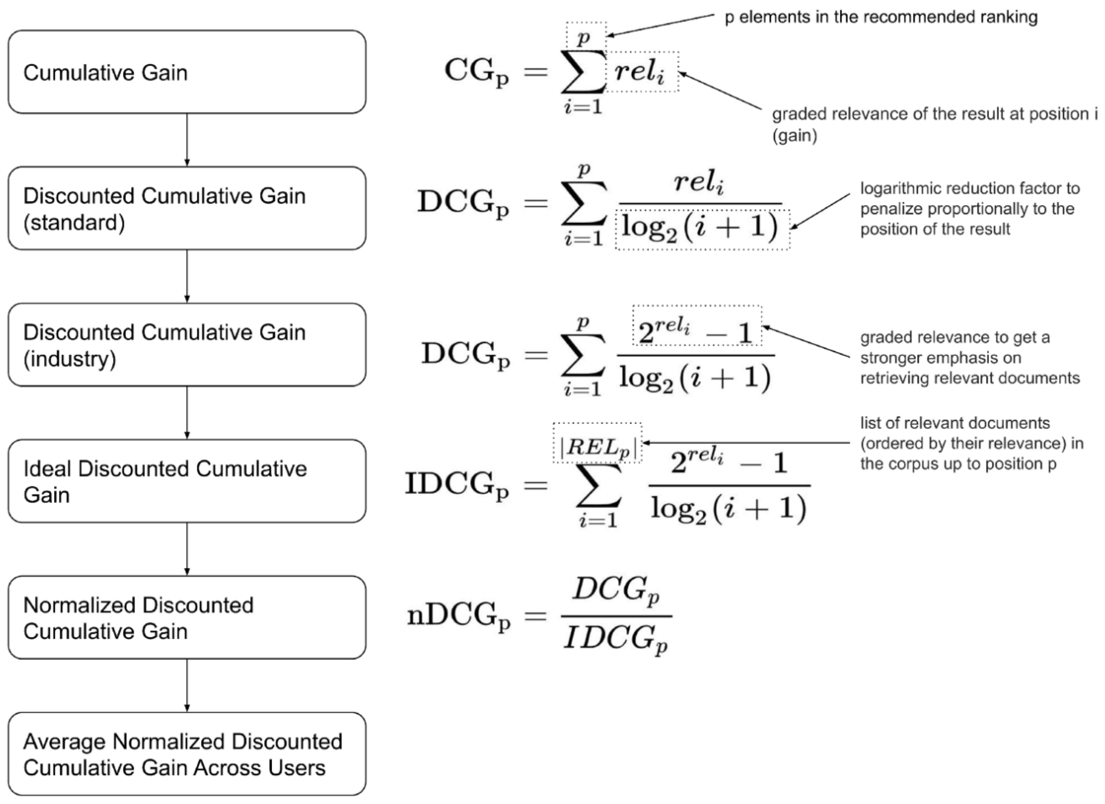
- For ratings to work, binary relevancies need to be defined

To overcome such issues, another more recent metric is introduced which is called Normalized Discounted Cumulative Gain (NDCG) metric

Normalized Discounted Cumulative Gain (NDCG)

The goal of the MAP measure is similar to the goal of the NDCG. They both value putting highly relevant documents high up the recommended lists. However, the NDCG further tunes the recommended lists evaluation. It enables to use the fact that some documents are more relevant than others.

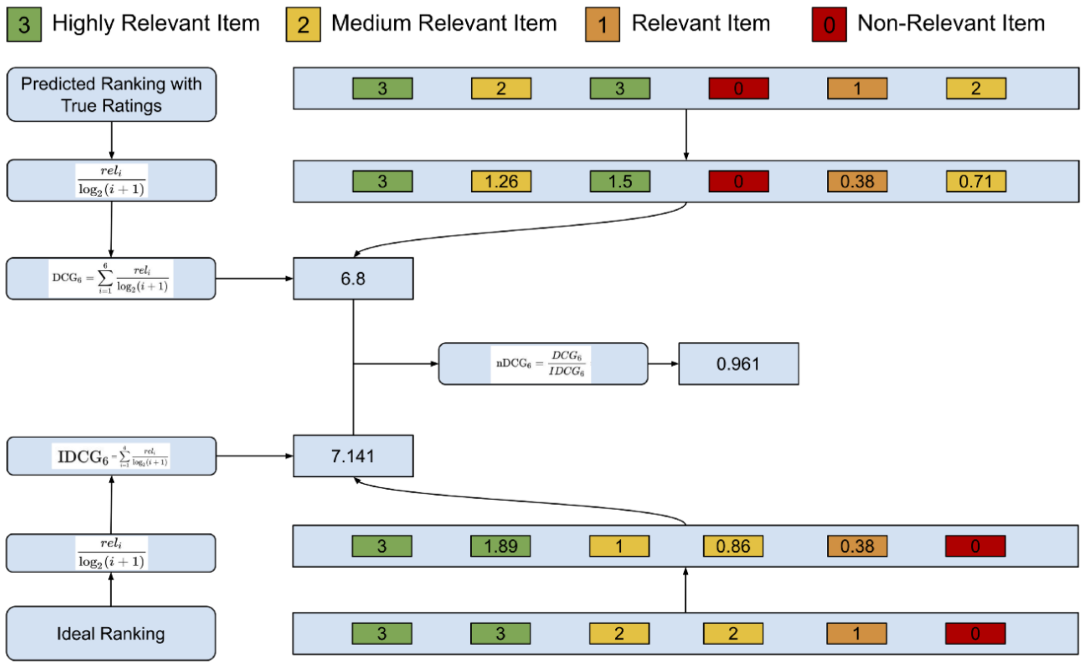
Algorithm:



Cumulative Gain
Discounted Cumulative Gain (standard)
Discounted Cumulative Gain (industry)
Ideal Discounted Cumulative Gain
Normalized Discounted Cumulative Gain
Average Normalized Discounted Cumulative Gain Across Users
$\mathrm{CG_p} = \sum_{i=1}^{p} rel_i$
p elements in the recommended ranking
graded relevance of the result at position i (gain)
$\mathrm{DCG_p} = \sum_{i=1}^{p} \frac{rel_i}{\log_2(i+1)}$
logarithmic reduction factor to penalize proportionally to the position of the result
$\mathrm{DCG_p} = \sum_{i=1}^{p} \frac{2^{rel_i}-1}{\log_2(i+1)}$
graded relevance to get a stronger emphasis on retrieving relevant documents
$\mathrm{IDCG_p} = \sum_{i=1}^{|REL_p|} \frac{2^{rel_i}-1}{\log_2(i+1)}$
list of relevant documents (ordered by their relevance) in the corpus up to position p
$\mathrm{nDCG_p} = \frac{DCG_p}{IDCG_p}$

Annotated derivation of the NDCG metric

For this method to work, first need to increase the relative impact of the position of elements in the ranked list. The NDCG adds a logarithmic reduction factor to penalise the relevance score proportionally to the position of the item.


3 Highly Relevant Item
2 Medium Relevant Item
1 Relevant Item
0 Non-Relevant Item
Predicted Ranking with True Ratings
3 2 3 0 1 2
$\frac{rel_i}{\log_2(i+1)}$
3 1.26 1.5 0 0.38 0.71
$DCG_6 = \sum_{i=1}^{6} \frac{rel_i}{\log_2(i+1)}$
6.8
$nDCG_6 = \frac{DCG_6}{IDCG_6}$
0.961
$IDCG_6 = \sum_{i=1}^{6} \frac{rel_i}{\log_2(i+1)}$
7.141
$\frac{rel_i}{\log_2(i+1)}$
3 1.89 1 0.86 0.38 0
Ideal Ranking
3 3 2 2 1 0

NDCG Pros

- Takes into account the graded relevance values. When they are available in the dataset, NDCG is a good fit.
- Compared to the MAP metric, it does a good job at evaluation the position of ranked items. It operates beyond the binary relevant/non-relevant scenario
- The smooth logarithmic discounting factor has a good theoretical basis.

NDCG Cons

- When the dataset has incomplete ratings, the NDCG has issues.
- The user needs to manually handle the case where the IDCG is equal to zero.
- This occurs when users have no relevant documents

Result/Implementation

Although it requires further research on how to implement on Redback Operation's coin system and gaming, a GBM ranking system is sound and effective as it provides deep intuition in deriving an efficient way of scoring a user/player's scores on the basis of which rank does he/she belong, hence entitlement of the coins.

Recommendation

All these ranking evaluation methods have their own usage depending on the scenario that it’s implemented on. For a simple ranking system, MRR does a fabulous job whereas MAP would be useful when evaluating cyclist’s average exercise pace. Lastly, NDCG can be useful for tailored/curated recommendation system where the algorithm recommends best cycling route, best game for cyclist’s specific and so on.