TME 3 - Evaluation

Exercice 1 – Exercice de compréhension : calcul des mesures

On considère la collection de documents et la liste des stopwords du TME1 ainsi que les fonctions d'appariement codées dans l'exercice 1 du TME 2. L'objectif dans cet exercice est de mesurer la qualité de l'ordonnancement. Pour cela, on définit les jugements de pertinence suivant :

- Requête 1 "top sales" - Documents pertinents : 1
- Requête 2 "sales increase july" - Documents pertinents : 2 et 3 (avec 2 plus pertinent que 3)
- Requête 3 "new home"

Q 1.1 Calculer les mesures de précision, rappel et F-mesure au rang 2 ($P@2$, $R@2$ et $F@2$) pour chaque requête et ensuite leur moyenne sur l'ensemble des requêtes.

Q 1.2 Calculer la mesure de NDCG pour toutes les requêtes.

Exercice 2 – Projet "Moteur de Recherche" : Etape Evaluation

1 Chargement du jeu de données

Dans le TME1, nous avons chargé la collection de documents pour l'indexer. Nous avons évoqué deux fichiers liés à l'évaluation :

- .qry : Des jeux de tests de requêtes (cisi.qry contient 112 requetes, cacm.qry en contient 64). Les balises .I et .W seront considérés pour prendre en compte l'identifiant et le texte de la requête.
- .rel : Les jugements de pertinence pour les requêtes de test. Sur chaque ligne, le premier nombre correspond à l'identifiant de la requête concernée et le second correspond a l'identifiant d'un document pertinent pour cette requête.

Q 2.1 Construire une classe `Query` qui permet de stocker l'identifiant de la requête, son texte et la liste des identifiants des documents pertinents.

Q 2.2 Construire une classe `QueryParser` permettant de lire les fichiers de tests de requêtes et de jugements de pertinence qui retourne une collection de `Query`.

2 Métriques

On souhaite maintenant implémenter les mesures d'évaluation (à minima, P@k, R@k, AP, NDCG pour chaque requête).

Q 2.3 Créer une classe abstraite `EvalMesure` qui contient une méthode abstraite evalQuery(liste,query) permettant de calculer la mesure pour la liste des documents retournés par un modèle et un objet `Query`.

Q 2.4 Créer les classes associées à chaque mesure d'évaluation (précision/rappel/f-measure au rang k, précision moyenne, reciprocal rank, ndcg - au minimum) ; ces classes héritent de la classe abstraite `EvalMesure`.

3 Plateforme d'évaluation

Q 2.5 Définir une classe `EvalIRModel` permettant l'évaluation de différents modèles de recherche sur un ensemble de requêtes selon différentes mesures d'évaluation. Les résultats devront être résumés pour l'ensemble

des requêtes considérées en présentant la moyenne et l'écart-type pour chaque modèle.

4 Bonus - 2.6 et 2.7 très fortement conseillés

Q 2.6 Compléter la classe `EvalIRModel` pour permettre de calculer des tests de significativité entre deux modèles (t-test) pour tester si deux modèles sont significativement différents.

Q 2.7 Poursuivre l'optimisation des paramètres du TME 2 en considérant comme critère à optimiser la valeur de MAP.

Q 2.8 Comme dans le TME1, pour l'implémentation du QueryParser, on peut également se questionner sur l'efficacité du parsing en cas de fichier volumineux...