1. **Impact des procédures sur le rappel et la précision :**

- **Rappel** ($Recall$) : Proportion des documents pertinents retrouvés parmi tous les documents pertinents disponibles.

$$Recall = \frac{\text{Documents pertinents retrouvés}}{\text{Documents pertinents totaux}}$$

- **Précision** ($Precision$) : Proportion des documents pertinents retrouvés parmi tous les documents retrouvés.

$$Precision = \frac{\text{Documents pertinents retrouvés}}{\text{Documents retrouvés totaux}}$$

a. **Lemmatisation des mots :**
- **Impact sur le rappel :** Augmente le rappel en regroupant les variantes d'un mot (par exemple, "courir", "couru", "court" deviennent "courir"). Cela permet de récupérer plus de documents pertinents.
- **Impact sur la précision :** Peut diminuer légèrement la précision si des formes ambiguës sont mal résolues (par exemple, "banc" pour "banque" et "banc de poissons").

b. **Utilisation des synonymes :**
- **Impact sur le rappel :** Augmente le rappel en élargissant la recherche pour inclure des termes équivalents (par exemple, "automobile" pour "voiture").
- **Impact sur la précision :** Peut réduire la précision si des synonymes non pertinents sont ajoutés au contexte de la recherche.

Suppression des mots vides
- **Description** : Élimination des termes fréquents et peu informatifs (ex. : "le", "et", "de").
- **Impact** :
  - **Rappel** : Inchangé ou légèrement diminué. Certains documents pertinents pourraient être ignorés si des mots vides portaient un sens dans le contexte.
  - **Précision** : Améliorée. Les documents contenant principalement des termes peu informatifs sont filtrés.

Utilisation des bigrammes de mots dans les modèles de RI basés sur des modèles de langue
- **Description** : Représentation des séquences de deux mots successifs (bigrammes) pour capturer des dépendances locales.
- **Impact** :
  - **Rappel** : Inchangé ou légèrement amélioré. Les bigrammes ajoutent un niveau de détail mais risquent de manquer certains documents qui utilisent des mots isolés.
  - **Précision** : Améliorée. **Les bigrammes aident à mieux comprendre le contexte immédiat**, réduisant ainsi les correspondances hors sujet.

Représentation des documents et des requêtes par des vecteurs [CLS] issus de BERT
- **Description** : Utilisation des représentations contextuelles générées par le token spécial [CLS] dans BERT.
- **Impact** :
  - **Rappel** : Amélioré. Les embeddings contextuels **capturent les relations sémantiques** complexes, augmentant les chances de trouver des documents pertinents.
  - **Précision** : Améliorée. La compréhension contextuelle aide à différencier les documents réellement pertinents de ceux qui contiennent des termes superficiellement similaires.

c. **Utilisation de la position des termes dans les documents :**
- **Impact sur le rappel :** N'a pas d'impact direct sur le rappel, mais permet de mieux classer les documents pertinents.
- **Impact sur la précision :** Augmente la précision en favorisant les documents où les termes apparaissent dans des positions pertinentes (par exemple, proches les uns des autres).

d. **Utilisation des expressions :**
- **Impact sur le rappel :** Réduit le rappel, car seules les occurrences exactes de l'expression sont prises en compte (par exemple, "machine learning" ne récupère pas "learning machine").
- **Impact sur la précision :** Améliore la précision en ciblant des documents contenant des expressions exactes, souvent plus pertinentes.

e. **Utilisation des représentations distribuées de mots (ex. word embeddings) :**
- **Impact sur le rappel :** Augmente le rappel en capturant des relations sémantiques entre les mots, même si des termes différents sont utilisés (par exemple, "chat" et "félin").
- **Impact sur la précision :** Peut améliorer la précision si le contexte est bien pris en compte, mais risque de l'abaisser en cas de bruit sémantique.

f. **Utilisation de l'opérateur AND dans le modèle booléen :**
- **Impact sur le rappel :** Réduit le rappel car seuls les documents contenant tous les termes sont récupérés.
- **Impact sur la précision :** Améliore la précision en ne récupérant que les documents qui répondent strictement à tous les critères de la requête.

---

2. Réponses Vrai/Faux avec justifications :

a. **Dans un système de recherche d'information qui utilise des n-grammes de caractères plutôt que des mots, la racinisation n'est pas nécessaire.**
- **Réponse : Vrai.**
  Les n-grammes de caractères (par exemple, "cour", "ouri", "rir") capturent automatiquement les variations morphologiques des mots, rendant la racinisation superflue.

b. **L'analyse sémantique latente permet de récupérer des documents pertinents même si ces documents n'ont aucun terme en commun avec la requête.**
- **Réponse : Vrai.**
  L'analyse sémantique latente (LSA) utilise des relations latentes entre termes et documents en réduisant les dimensions. Cela permet de trouver des correspondances sémantiques, même sans termes communs exacts.

c. **La normalisation par la longueur des documents vise à éviter que les documents courts ne soient classés trop haut.**
- **Réponse : Faux.**
  La normalisation par la longueur vise à éviter que les documents **longs** soient avantagés, car ils contiennent plus de termes et ont donc un score plus élevé dans des modèles comme TF-IDF.

d. **En général, comme les documents d'un ensemble de résultats sont listés par ordre décroissant de pertinence estimée, la précision diminue à mesure que le rappel augmente.**
- **Réponse : Vrai.**
  À mesure que le rappel augmente (en incluant plus de documents), des documents moins pertinents sont ajoutés, ce qui tend à réduire la précision.

Exercice 2 (6 pts)

On considère un système de recherche d'information basé le modèle de langue mixte (JM) combinant le modèle de document et le modèle de collection. Les probabilités sont estimées en se basant sur la fréquence des termes dans le document (pour le modèle de document) et la fréquence des termes dans la collection (pour le modèle de collection).

Considérons la requête suivante : q (XML, document), dont les ICF (Inverse Collection Frequency, fréquence relative des termes dans la collection) sont : 10/10 000 000 pour le terme "XML" et 10 000/10 000 000 pour le terme "document".

Soit une collection de 4 documents : d1, d2, d3 et d4. On suppose que le document d1 contient 1000 termes dont le terme « XML » qui apparaît une seule fois, et le document d2 contient également 1000 termes dont le terme « document » qui apparaît 1 fois. Les documents d3 et d4 ne contiennent pas ces deux termes.

Questions :

Donner l'ordre dans lequel les 4 documents seront renvoyés en réponse à la requête q

1. dans le cas d'un modèle de langue mixte JM avec lambda=0.5
2. dans le cas du modèle probabiliste BIR
3. dans le cas du modèle vectoriel utilisant la pondération de type *qqq.ddd=nnn.ltn*
4. On suppose que la requête q est pondérée q (XML 3, document 1), donner l'ordre dans lequel les documents seront renvoyés dans le cas du modèle (JM).

1. Modèle de langue mixte JM avec $\lambda = 0.5$

Le modèle mixte JM combine la probabilité basée sur le document ($P(t|d)$) et la probabilité basée sur la collection ($P(t|C)$) :

$$P(q|d) = \prod_{t \in q} \left[\lambda \cdot P(t|d) + (1 - \lambda) \cdot P(t|C) \right]$$

Données :

- $P(t|d)$: Fréquence du terme dans le document divisé par la longueur du document.
- $P(t|C)$: Fréquence du terme dans la collection (ICF).
- $\lambda = 0.5$

Calcul pour chaque document :

1. **Document d_1 (contient "XML" mais pas "document") :**
   - $P(XML|d_1) = \frac{1}{1000}$
   - $P(document|d_1) = 0$
   - $P(XML|C) = \frac{10}{10^7} = 10^{-6}$
   - $P(document|C) = \frac{10^4}{10^7} = 10^{-3}$
   - $P(q|d_1) = \left[0.5 \cdot \frac{1}{1000} + 0.5 \cdot 10^{-6}\right] \cdot \left[0.5 \cdot 0 + 0.5 \cdot 10^{-3}\right]$

$$P(q|d_1) \approx 0.5 \cdot 10^{-3} \cdot 0.5 \cdot 10^{-3} = 2.5 \cdot 10^{-7}$$

2. **Document d_2 (contient "document" mais pas "XML") :**
   - $P(XML|d_2) = 0$
   - $P(document|d_2) = \frac{1}{1000}$
   - $P(q|d_2) = \left[0.5 \cdot 0 + 0.5 \cdot 10^{-6}\right] \cdot \left[0.5 \cdot \frac{1}{1000} + 0.5 \cdot 10^{-3}\right]$

$$P(q|d_2) \approx 0.5 \cdot 10^{-6} \cdot 0.5 \cdot 10^{-3} = 2.5 \cdot 10^{-10}$$

3. **Documents d_3 et d_4 (ne contiennent ni "XML" ni "document") :**
   - $P(q|d_3) = P(q|d_4) = \left[0.5 \cdot 10^{-6}\right] \cdot \left[0.5 \cdot 10^{-3}\right]$

$$P(q|d_3) = P(q|d_4) = 2.5 \cdot 10^{-10}$$

Ordre des documents :

- $d_1 > d_2 = d_3 = d_4$

2. Modèle probabiliste BIR

Le modèle probabiliste classe les documents en fonction de leur probabilité d'être pertinents. Formule approximative :

$$P(R|d) \propto \prod_{t \in q} \left[\frac{P(t|d)}{1 - P(t|d)} \right]$$

Les calculs montrent que d_1 et d_2 auront des scores significatifs tandis que d_3 et d_4 auront des scores nuls, donnant :

- **Ordre des documents :** $d_1 = d_2 > d_3 = d_4$.

3. Modèle vectoriel (qqq.ddd = nnn.ltn)

Dans le modèle vectoriel :

$$\text{Score}(q, d) = \sum_{t \in q} w_{tq} \cdot w_{td}$$

- Pondération w_{td} dépend de la fréquence relative du terme dans le document et du poids w_{tq} qui dépend de l'importance du terme dans la requête.

Documents ayant des termes en commun avec la requête obtiennent des scores proportionnels aux fréquences. Par conséquent :

- **Ordre des documents :** $d_1 = d_2 > d_3 = d_4$.

Signification des notations (qqq.ddd = nnn.ltn)

1. **qqq (poids pour la requête) :**
   - **n** : Fréquence brute (tf) du terme dans la requête.
   - **n** : Pas de normalisation du poids pour la requête.
   - **n** : Pas d'utilisation de la fréquence inverse du document (idf).
2. **ddd (poids pour les documents) :**
   - **l** : Pondération logarithmique du tf dans le document ($1 + \log(tf)$).
   - **t** : Utilisation de l'idf pour les documents ($idf = \log(\frac{N}{df})$, où N est le nombre total de documents et df est le nombre de documents contenant le terme).
   - **n** : Pas de normalisation des longueurs des documents.

Formule générale pour le modèle vectoriel

La similarité entre un document d et une requête q est donnée par :

$$\text{Sim}(q, d) = \sum_{t \in q \cap d} w_{tq} \cdot w_{td}$$

- w_{tq} : Poids du terme t dans la requête.
- w_{td} : Poids du terme t dans le document.

Poids w_{tq} pour la requête (qqq = nnn) :

$$w_{tq} = tf_{tq}$$

tf_{tq} est la fréquence brute du terme t dans la requête.

Poids w_{td} pour le document (ddd = ltn) :

$$w_{td} = (1 + \log(tf_{td})) \cdot idf_t$$

- tf_{td} : Fréquence brute du terme t dans le document d.
- $idf_t = \log\left(\frac{N}{df_t}\right)$, où :
  - N : Nombre total de documents.
  - df_t : Nombre de documents contenant le terme t.

4. Modèle mixte JM avec pondération de la requête $q = (XML : 3, document : 1)$:

Ici, chaque terme a un poids dans la requête. Le score est pondéré par le poids du terme :

$$P(q|d) = \prod_{t \in q} \left[\lambda \cdot P(t|d) + (1-\lambda) \cdot P(t|C)\right]^{w_t}$$

Calculs :

1. **Document d_1 :**
   - $P(q|d_1) = \left[0.5 \cdot \frac{1}{1000} + 0.5 \cdot 10^{-6}\right]^3 \cdot \left[0.5 \cdot 0 + 0.5 \cdot 10^{-3}\right]^1$
2. **Document d_2 :**
   - $P(q|d_2) = \left[0.5 \cdot 0 + 0.5 \cdot 10^{-6}\right]^3 \cdot \left[0.5 \cdot \frac{1}{1000} + 0.5 \cdot 10^{-3}\right]^1$

Comparaison :

Le terme **"XML"** a un poids 3, donc d_1 sera favorisé.

Ordre des documents :

- $d_1 > d_2 > d_3 = d_4$.

---

a) Un document ne comportant aucun terme de la requête ne peut pas être pertinent pour cette requête ?
Réponse : Faux
Justification :
Dans des modèles comme l'**analyse sémantique latente** (LSA) ou les **représentations distribuées (word embeddings)**, un document peut être jugé pertinent même s'il ne contient pas les termes exacts de la requête. Ces modèles capturent des relations sémantiques entre les termes, permettant de récupérer des documents contenant des synonymes ou des termes sémantiquement proches des termes de la requête.

---

b) La représentation en sac de mots permet de capturer (représenter) le sens des mots.
Réponse : Faux
Justification :
La représentation en **sac de mots (Bag of Words)** ne tient pas compte de l'ordre des mots ni des relations syntaxiques ou sémantiques entre eux. Chaque mot est traité indépendamment, ce qui signifie que le "sens" contextuel des mots n'est pas représenté.

---

c) La représentation en trigrammes permet de capturer la syntaxe.
Réponse : Vrai
Justification :
La représentation en **trigrammes** (séquences de trois mots ou caractères) permet de capturer des informations contextuelles et syntaxiques locales. Par exemple, la phrase "je mange du pain" peut être représentée par les trigrammes "jemange","mangedu","dupain""je mange", "mange du", "du pain""jemange","mangedu","dupain", ce qui préserve une partie de l'ordre des mots et des relations grammaticales.

---

d) La racinisation permet-elle d'améliorer le rappel ou la précision ou les deux ?
Réponse : Elle améliore le **rappel, mais peut réduire la **précision**.
Justification :

- La **racinisation** (réduction des mots à leur racine commune) augmente le rappel en regroupant les termes similaires (e.g., "manger", "mange", "mangé" → "mang"), ce qui permet de retrouver des documents qui utilisent des formes différentes d'un mot.
- En revanche, cela peut réduire la précision en introduisant des ambiguïtés (e.g., "port" peut correspondre à "porter" ou "port maritime").

---

e) À votre avis, quel modèle de RI serait le plus approprié pour rechercher des Tweets ? Justifiez.
Réponse : Le modèle basé sur les représentations distribuées (e.g., Word2Vec, BERT) est le plus approprié.
Justification :

1. Les **tweets** sont courts, et les mots se répètent rarement. Les modèles comme Word2Vec ou BERT capturent les relations sémantiques entre les termes, même si ceux-ci ne se répètent pas. Cela permet de retrouver des tweets pertinents même avec des synonymes ou des termes contextuellement proches.
2. Les **modèles vectoriels pondérés** (e.g., TF-IDF) peuvent également être utilisés, mais ils dépendent fortement de la fréquence des mots, ce qui peut être moins efficace pour des messages courts.
3. Les modèles comme le **modèle booléen** ou le **modèle probabiliste** sont moins adaptés, car ils exigent des termes exacts ou des hypothèses sur la distribution des mots, ce qui est difficile à appliquer aux tweets, qui sont très variés et souvent informels.

Question 1 : Quelle est la principale différence entre BERT et GPT ?
Réponse :
a. BERT est bidirectionnel tandis que GPT est un modèle autoregressif.
Question 2 : Quel composant des Transformers permet de conserver l'ordre des séquences ?
Réponse :
b. Les embeddings positionnels.
Question 3 : Quel est le rôle principal des modèles encodeur/décodeur dans les Transformers ?
Réponse :
c. Traduire ou transformer des séquences d'entrée en séquences de sortie.
Question 4 : Quel modèle serait le plus adapté pour une tâche de génération de texte ?
Réponse :
c. GPT
Question 5 : Dans le cadre des Transformers, que signifie "fine-tuning" ?
Réponse :
b. Adapter un modèle pré-entraîné à une tâche spécifique.
Question 6 : Explorez le Hub (de Hugging Face) et cherchez le modèle roberta-large-mnli. Quelle tâche accomplit-il ?
Réponse :
c. Classification
Question 7 : Vrai ou faux ? Un modèle de langage n'a généralement pas besoin d'étiquettes pour son pré-entraînement
Réponse :
a. Vrai
Question 8 : Combien de dimensions le tenseur produit par le Transformer de base possède-t-il et quelles sont-elles ?
Réponse :
c. La longueur de la séquence, la taille du batch et la taille cachée.
Question 9 : Qu'est-ce qu'un AutoModel ?
Réponse :
b. Un objet qui renvoie la bonne architecture basée sur le checkpoint.
Question 10 : Comment le modèle BERT attend-il qu'une paire de phrases soit traitée ?
Réponse :
a. [CLS] Tokens_de_la_phrase_1 [SEP] Tokens_de_la_phrase_2 [SEP]
Question 11 : Quel est le but de TrainingArguments ?
Réponse :
c. Contenir tous les hyperparamètres utilisés pour l'entraînement et l'évaluation avec le Trainer.

Question 1 : Qu'est-ce qu'une représentation en sac de mots (Bag of Words) ?
Réponse : c. Une représentation des textes où chaque mot est indépendant et compte uniquement son occurrence.
Explication :
La représentation en sac de mots ignore l'ordre des mots et se concentre uniquement sur la fréquence ou la présence des mots dans un texte, sans prendre en compte leur contexte.

Question 2 : Qu'est-ce que le lissage dans les modèles de langage ?
Réponse : c. Une méthode pour donner des probabilités non nulles aux mots absents.
Explication :
Le lissage (e.g., Laplace, Lidstone, ou Jelinek-Mercer) est utilisé pour attribuer une probabilité non nulle aux mots ou combinaisons de mots qui n'apparaissent pas dans les données d'entraînement, évitant des erreurs dues à des probabilités nulles.

Question 3 : Quels modèles sont cités comme des exemples de "word embeddings" ?
Réponse : c. Glove, Word2Vec, et FastText.
Explication :
Ces modèles représentent les mots en tant que vecteurs dans un espace continu et capturent leurs relations sémantiques en fonction de leur contexte.

Question 4 : Quelle phrase décrit le mieux la représentation distribuée (continue) des mots ?
Réponse : b. Les mots sont représentés en fonction de leurs voisins dans le texte.
Explication :
La représentation distribuée utilise le contexte local (comme les mots voisins) pour encoder les relations sémantiques des mots dans un espace vectoriel.

Question 5 : Quel est le rôle principal d'un modèle de langage ?
Réponse : d. Prédire le mot suivant dans une séquence de mots.
Explication :
Le rôle fondamental d'un modèle de langage est de calculer la probabilité d'une séquence de mots, souvent en prédisant le mot suivant. C'est une base pour des tâches comme la génération de texte ou la complétion automatique.

Question 1 : Quel est le rôle principal d'un modèle de langage ?
Réponse :
c. Prédire le mot suivant dans une séquence de mots
Explication : Le rôle principal d'un modèle de langage est de modéliser la probabilité d'apparition d'une séquence de mots, souvent en prédisant le mot suivant à partir du contexte donné.

Question 2 : Qu'est-ce que le lissage dans les modèles de langage ?
Réponse :
b. Une méthode pour donner des probabilités non nulles aux mots absents
Explication : Le lissage est une technique utilisée pour gérer le problème des mots ou séquences de mots absents dans les données d'apprentissage. Cela garantit que même les mots rares ou inconnus reçoivent une probabilité non nulle.

Question 3 : Quelle phrase décrit le mieux la représentation distribuée (continue) des mots ?
Réponse :
c. Les mots sont représentés en fonction de leurs voisins dans le texte
Explication : Les représentations distribuées (par exemple, les vecteurs Word2Vec ou embeddings) capturent les relations sémantiques entre les mots en apprenant à les représenter dans un espace vectoriel en fonction des contextes dans lesquels ils apparaissent.

---

a) Un document ne comportant aucun terme de la requête ne peut pas être pertinent pour cette requête.

- **Réponse** : **Faux**
  - **Justification** : Les représentations modernes, comme les embeddings contextuels (BERT, GPT), permettent d'identifier des documents pertinents grâce à des relations sémantiques. Par exemple, un document contenant "IA" peut être pertinent pour une requête "intelligence artificielle" même sans partage de termes exacts.

---

b) La représentation en sac de mots permet de capturer (représenter) le sens des mots.

- **Réponse** : **Faux**
  - **Justification** : Le modèle BoW (Bag of Words) ignore l'ordre des mots et ne tient pas compte des relations sémantiques ou contextuelles. Par exemple, "chat noir" et "noir chat" sont représentés de manière identique, ce qui empêche de capturer leur sens.

---

c) La représentation en trigrammes permet de capturer la syntaxe.

- **Réponse** : **Vrai**
  - **Justification** : Les trigrammes capturent des séquences de trois mots consécutifs, ce qui aide à représenter des structures syntaxiques locales. Par exemple, dans "le chat mange", la dépendance entre "chat" et "mange" est conservée.

---

d) L'utilisation des représentations distribuées de mots (Word Embedding) permet de sélectionner des documents pertinents même si ces derniers n'ont aucun terme en commun avec la requête.

- **Réponse** : **Vrai**
  - **Justification** : Les embeddings distribués (Word2Vec, GloVe) capturent les similarités sémantiques entre les mots. Un document contenant "canin" pourrait être pertinent pour une requête "chien" grâce à la proximité de leurs vecteurs dans l'espace sémantique.

---

e) Un système de recherche d'information basé sur un modèle de langue de type bigrammes renvoie deux listes différentes pour les requêtes suivantes : "Information retrieval" et "Retrieval information".

- **Réponse** : **Vrai**
  - **Justification** : Les bigrammes capturent des relations entre mots consécutifs. "Information retrieval" génère les bigrammes ["information retrieval"], tandis que "Retrieval information" génère ["retrieval information"]. Les différences dans les représentations entraînent des listes de documents distinctes.

---

3. Modèle de RI pour la recherche de tweets

- **Modèle recommandé** : **Modèle probabiliste avec Word Embeddings (Dense Retrieval)**.
- **Justification** :
  - Les tweets sont courts et les répétitions de mots rares. Les modèles traditionnels comme BM25, qui se basent sur la fréquence des termes, risquent de mal performer.
  - Les embeddings (BERT, GPT) capturent le contexte et les relations sémantiques, ce qui est utile pour identifier des tweets pertinents même en présence de termes différents.

---

4. Représentation dans des espaces réduits (BERT et GPT)

- **Réponse** : Les modèles BERT et GPT **ne suppriment pas 150 mille mots**, mais utilisent des mécanismes avancés pour représenter un grand vocabulaire dans un espace réduit :
  - **Tokenisation sous-mot** : Ces modèles segmentent les mots rares en sous-unités. Par exemple :
    - "Antidisestablishmentarianism" → ["anti", "##dis", "##establishment", "##arianism"].
    - Cela réduit la taille du vocabulaire tout en permettant de représenter des mots rares ou nouveaux.
  - **Spécialisation contextuelle** : Les mots partagent des représentations communes si leurs parties sont similaires. Ainsi, les sous-mots s'assemblent dynamiquement selon le contexte pour former des significations spécifiques.

- **Avantages** :
  - Flexibilité pour traiter des mots inconnus.
  - Réduction de la mémoire requise pour stocker les paramètres du modèle.

Exercice 3 (4 pts)

Soit $q = q_1, \ldots, q_m$ une requête, d un document et $P(q_i|d)$ la probabilité du mot q_i dans le modèle de langue de d. On suppose que nous disposons d'une collection de documents comportant au total 8 mots $w1, \ldots, w8$.

La Table ci-dessous liste pour chaque mot sa probabilité dans le modèle de langue de référence, $P_{ml}(w|REF)$, estimé sur la collection (2ème colonne), la fréquence du terme c(w; d) dans un document (3ème colonne). Les colonnes 4 et 5 représentent les probabilités du terme dans le modèle langue du document d, estimé respectivement selon le maximum de vraisemblance et Dirichlet avec le paramètre μ.

Mots	$P_{ml}(w\|REF)$	c(w, d)	$P_{ml}(w\|d)$	$P_{dir}(w\| d)$
w1	0.3	2		
w2	0.15	1		
w3	0.1	2		0.125
w4	0.1	4		
w5	0.05	1		
w6	0.1	0		
w7	0.1	0		
w8	0.1	0		

1- Remplir la colonne 4, ($P_{ml}(w|d)$), le modèle de langue du document.
2- La colonne 5 représente la probabilité du terme calculée après un lissage de Dirichlet effectuée sur la collection. Seule la probabilité de $w3$ est donnée dans le tableau, déduire la valeur de μ? (posez l'équation puis déduire cette valeur)

Étape 1 : Calcul de $P_{ml}(w|d)$ (Colonne 4)

La formule pour le modèle de maximum de vraisemblance est :

$$P_{ml}(w|d) = \frac{c(w,d)}{\text{Total des occurrences dans } d}$$

Le total des occurrences dans d est la somme des $c(w,d)$:

$$\text{Total} = 2+1+2+4+1+0+0+0 = 10$$

On calcule donc $P_{ml}(w|d)$ pour chaque mot :

- Pour w_1 : $P_{ml}(w_1|d) = \frac{2}{10} = 0.2$
- Pour w_2 : $P_{ml}(w_2|d) = \frac{1}{10} = 0.1$
- Pour w_3 : $P_{ml}(w_3|d) = \frac{2}{10} = 0.2$
- Pour w_4 : $P_{ml}(w_4|d) = \frac{4}{10} = 0.4$
- Pour w_5 : $P_{ml}(w_5|d) = \frac{1}{10} = 0.1$
- Pour w_6, w_7, w_8 : $P_{ml}(w|d) = \frac{0}{10} = 0.0$

La colonne 4 devient donc :

Mots	$P_{ml}(w\|REF)$	$c(w,d)$	$P_{ml}(w\|d)$
w_1	0.3	2	0.2
w_2	0.15	1	0.1
w_3	0.1	2	0.2
w_4	0.1	4	0.4
w_5	0.05	1	0.1
w_6	0.1	0	0.0
w_7	0.1	0	0.0
w_8	0.1	0	0.0

Étape 2 : Calcul de $P_{dir}(w|d)$ pour w_3 et déduction de μ

La formule pour le lissage de Dirichlet est :

$$P_{dir}(w|d) = \frac{c(w,d) + \mu \cdot P_{ml}(w|\text{REF})}{\text{Total} + \mu}$$

On sait que pour w_3, $P_{dir}(w_3|d) = 0.125$.
En remplaçant dans la formule :

$$0.125 = \frac{2 + \mu \cdot 0.1}{10 + \mu}$$

Isolons μ :

$$0.125(10+\mu) = 2 + 0.1\mu$$

$$1.25 + 0.125\mu = 2 + 0.1\mu$$

$$0.125\mu - 0.1\mu = 2 - 1.25$$

$$0.025\mu = 0.75 \implies \mu = \frac{0.75}{0.025} = 30$$

Donc, $\mu = 30$.

Étape 3 : Calcul de $P_{dir}(w|d)$ (Colonne 5)

On utilise maintenant $\mu = 30$ pour calculer $P_{dir}(w|d)$ pour les autres mots :

- Pour w_1 :

$$P_{dir}(w_1|d) = \frac{2 + 30 \cdot 0.3}{10 + 30} = \frac{2+9}{40} = \frac{11}{40} = 0.275$$

- Pour w_2 :

$$P_{dir}(w_2|d) = \frac{1 + 30 \cdot 0.15}{10 + 30} = \frac{1+4.5}{40} = \frac{5.5}{40} = 0.1375$$

- Pour w_4 :

$$P_{dir}(w_4|d) = \frac{4 + 30 \cdot 0.1}{10 + 30} = \frac{4+3}{40} = \frac{7}{40} = 0.175$$

- Pour w_5 :

$$P_{dir}(w_5|d) = \frac{1 + 30 \cdot 0.05}{10 + 30} = \frac{1+1.5}{40} = \frac{2.5}{40} = 0.0625$$

- Pour w_6, w_7, w_8 (tous identiques car $c(w,d) = 0$) :

$$P_{dir}(w|d) = \frac{0 + 30 \cdot 0.1}{10 + 30} = \frac{3}{40} = 0.075$$

Résultat final

Le tableau complété est :

Mots	$P_{ml}(w\|\text{REF})$	$c(w,d)$	$P_{ml}(w\|d)$	$P_{dir}(w\|d)$
w_1	0.3	2	0.2	0.275
w_2	0.15	1	0.1	0.1375
w_3	0.1	2	0.2	0.125
w_4	0.1	4	0.4	0.175
w_5	0.05	1	0.1	0.0625
w_6	0.1	0	0.0	0.075
w_7	0.1	0	0.0	0.075
w_8	0.1	0	0.0	0.075

Exercice 4 (4pts) :

La table ci-dessous montre les documents trouvés par un système de recherche d'information, S, en réponse à une requête, parmi les 10 documents d'une collection. Les documents retrouvés sont listés par ordre décroissant de leur pertinence (calculée par un modèle (BM25). La valeur **1** de la table indique que le système a effectivement sélectionné le document spécifié dans la colonne correspondante et la valeur **0** indique que le document n'a pas été pas retrouvé. La dernière ligne « Pert » indique si le document est pertinent (noté 1) ou non pertinent (noté 0) pour la requête.

docs	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10
S	1	0	0	1	1	1	0	0	1	1
Pert	1	0	1	0	1	0	1	0	1	0

Questions

1- Représenter graphiquement la courbe de rappel et de précision interpolée de S.
2- Calculer la précision moyenne de S
3- Calculer également sa R-Précision.

Étape 1 : Calcul de la précision et du rappel pour chaque point

La précision (P) et le rappel (R) sont définis comme suit :

- **Précision** : $P = \frac{\text{Nombre de documents pertinents retrouvés}}{\text{Nombre total de documents retrouvés}}$
- **Rappel** : $R = \frac{\text{Nombre de documents pertinents retrouvés}}{\text{Nombre total de documents pertinents}}$

Données initiales :

- Les documents pertinents (Pert = 1) sont : $D1, D3, D5, D7$ → 4 documents pertinents au total.
- Les documents retrouvés sont ceux marqués 1 dans la ligne S.

Étapes de calcul :

Rang	Docs retrouvés	Pertinence (Pert)	Nb pertinents retrouvés	Précision (P)	Rappel (R)
1	D1	1	1	$1/1 = 1.0$	$1/4 = 0.25$
2	D2	0	1	$1/2 = 0.5$	$1/4 = 0.25$
3	D3	1	2	$2/3 \approx 0.666$	$2/4 = 0.5$
4	D4	0	2	$2/4 = 0.5$	$2/4 = 0.5$
5	D5	1	3	$3/5 = 0.6$	$3/4 = 0.75$
6	D6	0	3	$3/6 = 0.5$	$3/4 = 0.75$
7	D7	1	4	$4/7 \approx 0.571$	$4/4 = 1.0$
8	D8	0	4	$4/8 = 0.5$	$4/4 = 1.0$
9	D9	0	4	$4/9 \approx 0.444$	$4/4 = 1.0$
10	D10	0	4	$4/10 = 0.4$	$4/4 = 1.0$

Étape 2 : Courbe de rappel et de précision interpolée

Pour tracer la courbe interpolée, on considère la précision maximale atteinte pour chaque niveau de rappel R.

Rappel (R)	Précision maximale interpolée
0.25	1.0
0.5	0.666
0.75	0.75
1.0	1.0

Je peux générer la courbe si besoin.

Étape 3 : Calcul de la précision moyenne

La précision moyenne est la moyenne des précisions obtenues aux rangs où un document pertinent est retrouvé.
Les rangs pertinents sont 1, 3, 5, et 7. Les précisions associées sont $1.0, 0.666, 0.6, 0.571$.

$$\text{Précision moyenne} = \frac{1.0 + 0.666 + 0.6 + 0.571}{4} \approx 0.709$$

Étape 4 : Calcul de la R-Précision

La R-Précision correspond à la précision calculée après avoir retrouvé R documents, où R est le nombre total de documents pertinents.

Ici, $R = 4$.

Parmi les 4 premiers documents retrouvés ($D1, D2, D3, D4$), 2 sont pertinents ($D1, D3$).

$$\text{R-Précision} = \frac{2}{4} = 0.5$$

Exercice 2 (6)

Nous disposons d'une collection comportant les 4 documents suivants:

D1. jean donne un livre à marie
D2. jean qui lit le livre travaille avec marie
D3. qui pense que jean travaille avec marie ?
D4. jean pense qu'un livre est un bon cadeau

Questions

1- On suppose que ces documents sont indexés avec suppression des mots vides (à, un avec, qui, que, qu', est, le) et racinisation à 7 caractères. Proposez une structure de fichier inversé (dictionnaire+*posting*) permettant de représenter ces documents indexés

2- Considérons la requête suivante : **q** (marie marie travail)
Donner les scores de pertinence des documents D1 et D2 vis-à-vis de la requête **q** pour les 4 modèles suivants :

- Le modèle vectoriel utilisant la pondération de type qqq.ddd=nnn.ltn,
- Le modèle probabiliste BIR,
- Le modèle de langue basé sur l'interpolation de Jelinek Mercer (JM) avec λ=0.5

1. Fichier inversé (Dictionnaire + Posting List)

Étapes :

1. **Suppression des mots vides** : On supprime les termes comme "à", "un", "avec", "qui", "que", etc.
2. **Racinisation** : On réduit les mots à leurs racines de 7 caractères.
3. **Indexation** : On crée un dictionnaire associant chaque terme indexé à sa liste de documents contenant le terme et leur fréquence.

Documents indexés :

- **D1** : jean, donne, livre, marie
- **D2** : jean, lit, livre, travail, marie
- **D3** : pense, jean, travail, marie
- **D4** : jean, pense, livre, cadeau

Fichier inversé :

Terme	Posting List (Document, Fréquence)
cadeau	D4 (1)
donne	D1 (1)
jean	D1 (1), D2 (1), D3 (1), D4 (1)
lit	D2 (1)
livre	D1 (1), D2 (1), D4 (1)
marie	D1 (1), D2 (1), D3 (1)
pense	D3 (1), D4 (1)
travail	D2 (1), D3 (1)

2. Calcul des Scores de Pertinence pour les Documents D1 et D2

2.1. Modèle vectoriel avec pondération $qqq.ddd = nnn.ltn$

- **Formule** : Le score est calculé par la similarité cosinus.

$$\text{Score}(q, d) = \frac{\sum_{t \in q \cap d} w_{t,q} \cdot w_{t,d}}{\sqrt{\sum_{t \in d} w_{t,d}^2} \cdot \sqrt{\sum_{t \in q} w_{t,q}^2}}$$

- **Pondération** :
  - $w_{t,q} = 1 + \log(tf_{t,q})$ (l pour tf log dans la requête).
  - $w_{t,d} = \text{tf}_{t,d} \cdot \log(N/df_t)$ (t pour tf et idf dans le document).

Calcul pour D1 :

- Intersection ($q \cap D1$) : {marie}.
- $w_{\text{marie},q} = 1 + \log(2) = 1.301$.
- $w_{\text{marie},D1} = 1 \cdot \log(4/3) = 0.125$.
- Score :

$$\text{Score}(q, D1) = \frac{1.301 \cdot 0.125}{\sqrt{(0.125)^2} \cdot \sqrt{(1.301)^2}} = 0.125$$

Calcul pour D2 :

- Intersection ($q \cap D2$) : {marie, travail}.
- $w_{\text{marie},q} = 1.301,\ w_{\text{travail},q} = 1.301$.
- $w_{\text{marie},D2} = 0.125,\ w_{\text{travail},D2} = 0.125$.
- Score :

$$\text{Score}(q, D2) = \frac{(1.301 \cdot 0.125) + (1.301 \cdot 0.125)}{\sqrt{(0.125)^2 + (0.125)^2} \cdot \sqrt{(1.301)^2 + (1.301)^2}} \approx 0.354$$

2.2. Modèle probabiliste (BIR)

- **Formule** :

$$P(\text{pertinent}|q, d) = \prod_{t \in q} \frac{P(t|\text{pertinent})}{P(t|\text{non pertinent})}$$

- **Hypothèses** :
  - $P(t|\text{pertinent}) = \frac{\text{tf}_{t,d}}{\sum \text{tf}_{t,d} \text{dans les documents pertinents}}$.
  - $P(t|\text{non pertinent}) = 1 - P(t|\text{pertinent})$.

- **Calcul pour D1 :**
  - $P(\text{marie}|D1) = \frac{1}{2},\ P(\text{travail}|D1) = 0$.
  - Score :

$$P(\text{pertinent}|q, D1) = \frac{1/2}{1 - 1/2} = 1$$

- **Calcul pour D2 :**
  - $P(\text{marie}|D2) = \frac{1}{2},\ P(\text{travail}|D2) = \frac{1}{2}$.
  - Score :

$$P(\text{pertinent}|q, D2) = \frac{1/2}{1 - 1/2} \cdot \frac{1/2}{1 - 1/2} = 1$$

2.3. Modèle de Langue (Interpolation de Jelinek-Mercer)

- **Formule** :

$$P(t|d) = \lambda \cdot P(t|d) + (1 - \lambda) \cdot P(t|C)$$

- $P(t|C) = \frac{\text{tf}_t \text{dans } C}{\text{total des termes dans } C}$.

Calcul pour D1 :

- $P(\text{marie}|D1) = 0.5 \cdot (1/4) + 0.5 \cdot (3/12) = 0.25$.
- Score :

$$P(q|D1) = P(\text{marie}|D1)^2 = 0.0625$$

Calcul pour D2 :

- $P(\text{marie}|D2) = 0.5 \cdot (1/5) + 0.5 \cdot (3/12) = 0.25$.
- $P(\text{travail}|D2) = 0.5 \cdot (1/5) + 0.5 \cdot (1/12) = 0.15$.
- Score :

$$P(q|D2) = P(\text{marie}|D2)^2 \cdot P(\text{travail}|D2) = 0.09375$$