HW5 - Query Modeling

- **使用套件**

  使用 numpy 處理矩陣以及數學邏輯運算。

```
import numpy as np
import math
```

- **資料前處理**

  1. 計算 tf，計算每篇 document 中出現的 word，並將其儲存至 np.array 轉換成矩陣。
  2. 計算 df 和 idf，利用 df 過濾文字，並將 idf 存成 dictionary 增加效能，應用於 BM25 上。
  3. 計算 Background Word，將其除以全部文章字數的總合做正規化，並存成矩陣，類似於 idf 的概念，應用於 SMM 上。

  ☆ 策略：為了減少記憶體使用量，以及濾除不必要的字，我只保留數量大於 9 的字，最後將 154240 的字過濾成只剩 31389 個字。

- **找出 Relevant Document**

  先利用 BM25 計算出每個 query 文章的分數，並選前五篇分數高的作為每個 query 的相關文章。
  參數設定：K1 = 0.8, b = 0.7。

- **初始化參數 Psmm**

  利用 random 隨機產出 [query_len, word_len] 維度的數字，用來表示機率 $P_{smm}(w)$，並除以總合做 normalize 保證和為 1。

- **實作 Simple Mixture Model**

  SSM 產生出的相關性文章是混和 query 和 background word 兩個部分而成，過程利用 EM algorithm 更新參數 P_{smm}，Estep 中包含 background word 的資訊並產生出 $P(T_{smm}|w)$，Mstep 中包含 query 的相關文章，再利用 $P(T_{smm}|w)$ 更新 P_{smm}，重複此步驟直到迭代結束。

  ✓ Mstep 中的 document，是利用 BM25 算出來對每個 query 的相關文章。

  ✓ 參數設定：*iteration = 30*

- **實作 KL Divergence**

  利用 KL Divergence 來計算 query 中每個 document 的分數，此演算法可計算兩個 distribution 的相近程度，數值越大表示兩個分布越不相近。

  1. 參數設定：$\alpha = 0.15$，$\beta = 0.8$，$\gamma = 0.3$
  2. 利用前處理得到的 tf、BG_word，以及 SMM model 得出的 P_{smm} 進行計算。
  3. 再結合 query 以及 document 的 tf 來得到 score。

  ☆ 認為 P_{smm} 是比較重要的參數，因此將 β 調高至 0.8，分數從 0.56 上升至 0.57。

- **實作遇到的困難、心得**

  這次作業原本利用 Rocchio 加上 BM25 就可達到 0.54 的分數，不料越到 deadline 同學們的分數就越高，為了不讓分數太難看我選擇了用 SMM 再實作出一個 model，有了上次 PLSA 的經驗後，這次實作 SMM 變得相對簡單。唯一要注意的是，這次文章的分數不能只看 query 中出現的字，而是要將所有 relevent document 中字的分數也包含進去，這是我在實作中有不小心寫錯的一個錯誤。