Information Retrieval – HW1

B10615046 柯元豪

**實作環境**

Jupyter notebook (Python 3)

**使用的套件**

numpy, pandas, math, nltk

**資料前處理**

指定document, query的目錄，用nltk的PlaintextCorpusReader去目錄下存取所有檔案，之後將檔案分別存至對應的list中

**參數調整**

1. TF 做 sublinear (4 + log(tf))

因應詞頻的算法為(詞在文件中出現次數/文件總詞長)，原先sublinear的方法應該為 1+log(tf)，但除了總詞長平均log完都會 -3 (因文件的詞均長大概1000)，因此這裡的參數調整為 4 (1 + 3)

1. IDF 做 smooth (log(1 + N / df))

嘗試過分子分母同時加上0.5，分數反而下降，這是調整後最高分的smooth

**運作原理**

1. 依序讀取query
2. 取得此query與所有文件的 d向量與 q向量
3. 相似度計算 這裡採用 d dot q2 / |d| 因為做法是依序做每個query跟document的tfidf，因此query的長度並不影響計算，便只normalize d，而q的向量做了multiply分數卻變高的部分尚且不知道原理
4. 將document排序並匯出

**心得**

在前處理與函式實作的部分都是先以自己的理解去做，後來vector space model 的相似度計算那邊，document vector 跟query vector要做內積時發現得跟講義做不一樣，因為自己的做法不是將所有query跟document一起做出一個大矩陣，而是將每個query分開來跟document做，所以在寫主函式的流程有點卡。因此最大的收穫是應該先研究得更熟悉一點，才能寫得更輕鬆。