|  |
| --- |
| 運用工具 & 資料前處理說明 |
| 1. 透過nltk的word.tokenize function，將Documents與Queries進行斷詞。 2. 運用snowball\_stemmer function將詞性還原。 3. 使用nlt.corpus中stopwords篩選掉不帶有資訊的字詞，ex: the, a, and…。 4. 使用numpy做陣列操作。而math、operator做資料運算ex:math.log。 |
| TF參數設定-Documents、Quries |
|  |
| 1. Documents TF計算：   計算該文章每個字出現的次數，並運用Log Normalization ( 1+log(tf) )概念平滑TF參數，以完成該doc各文字的TF計算。   1. Query TF 計算：   計算query各文字出現的次數則是運用最原始的TF計算方式。 |
| 相似度計算-BM25 |
|  |
| 1. 相似度計算：   本作業主要使用方法為BM25L結合BM11中Correction Factor的概念進行實作，經由實驗參數設定結果如下：  K1 = 3、b = 0.685、K2 = 0.01751、K3 = 0.2、delta = 0.5，  由於，各大論文得出BM11的K2值=0通常會得到較好的準確度，但經由實驗 K2>0在我的模型中的準確率會提升，此外，「平均長度-長度」有可能為負值導致所求分數減少，因此我試套上絕對值後準確度的確會提升。  再者，在實作 𝑡𝑓𝑖’j時前面有多X 0.765，該參數亦是經由實驗而進行設定，而我認為該參數設定是要將Doc Term Frequency的值更平滑。 |
| 心得 |
| 由於第一次的作業結束後，老師有請作業1表現前幾名的同學上台分享他們的做法，聽完他們的做法後，了解到在製作tf的參數設定時，盡量讓tf的向量平滑，因此，再做第二份作業的時候，就沒有那麼耗費心力就可超過baseline。  但想要達到前20名還是相當困難的，所以這次我還是很好奇前面的同學是怎麼實做的，希望這次亦可聽到前面的高手回答。  當然，在實作的過程中，促使我對整個model的運算過程更加了解，雖然排名沒有達到前20，其實還是蠻難過的….。但從無到有，並且嘗試了各式各樣的參數以及異想天開的想法後，當看到準確度有越來越提升，其實還是相當開心的。 |