

Information Retrieval and Applications

HW01-Vector Space Model

四資工四甲B10515047陳牧凡

此次作業運用到的MODEL是vector space model，透過計算query以及document之間的cosine similarity來找出吻合度，並得到符合該query描述的document優先排序。而在計算相似度的過程中，將query和document轉換成向量，其由n個詞(term/word)組成，每個詞都有一個權重，不同的詞根據自己在文件中的權重來影響文件相關性的重要程度。

具體演算法過程如下：

1. 讀取document以及query的文字，將所用到的字做成一個dictionary。
其目的在於方便之後將文章轉換成N-dimension的向量(N代表的dictionary收錄的字數)。
2. 取得TF-IDF

$$TF - IDF_{i,j} = tf_{i,j} \times idf_i$$

計算出每個字在每篇文章中的權重，除了考慮該字在文章中的出現頻率(TF)外，某些文字可能是如語助詞之類的並沒有實質意義，同時該字可能出現於大多數的文章，那這種情況可能就代表這個字沒有識別度，那這時會透過得到其IDF來降低該字的權重，以增進精準度。

a. TF

該字在文章中出現的頻率。我有實作原始頻率以及log正規化的計算方式，其中經log正規化的效果略好一點。

Raw Frequency	$tf_{i,j}$
Log Normalization	$1 + \log_2(tf_{i,j})$

b. IDF

包含該字的文章的倒頻率。

Inverse Frequency	$\log \frac{N}{n_i}$
-------------------	----------------------

3. 計算query和document之間的cosine similarity。

計算了每個文章的TF-IDF後，應該分別都會得到一個向量，然後將query的向量和document的向量取cosine，得到兩個向量之間的"夾角"來判斷兩者之間的吻合度，數字越接近1代表越符合。

可以使用sklearn寫好的library來直接計算兩個向量的cosine。


```
8 from sklearn.metrics.pairwise import cosine_similarity
```

不過老師說不可以用工具包，要自己寫所以：

```
9 import numpy as np
10 |
11 #teacher said we can not use the "工具包", so...
12 def calculateCosineValue(q,dj):#parameters are vectors which present query's and doc's TFIDF
13     return np.dot(q, dj) / (np.linalg.norm(q)*np.linalg.norm(dj))
```

雖然我不知道這樣差在哪裡就是了...

最後得到的正確率是:56%

2	B10515047_陳牧凡		0.56656	14	now
Your Best Entry ↑					
Your submission scored 0.56656, which is not an improvement of your best score. Keep trying!					