# HW2 Report

R07725021 資管碩1 洪靖雯

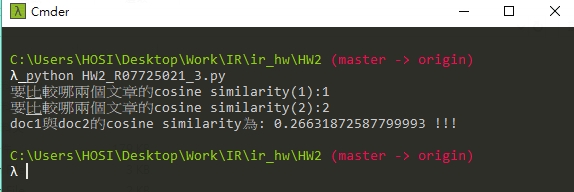
## 1.執行環境

* Sublime
* Cmder

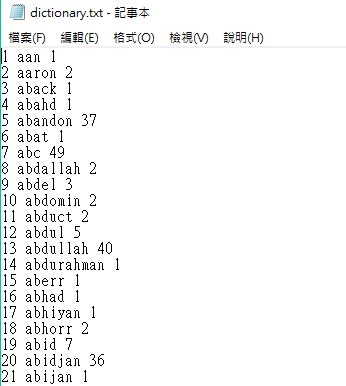
## 2.程式語言

* Python 3.6.6

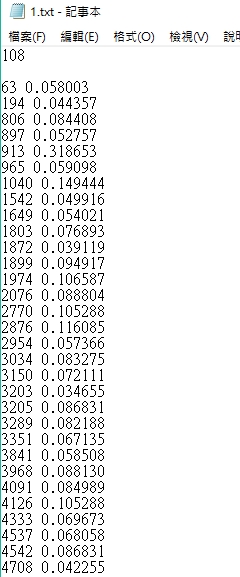
## 3.執行方式

* 套件: requests、bs4、nltk
  + 有另外用conda 4.5.8架虛擬環境安裝，直接用在system裡應該也可行
  + 使用pip install方式安裝
    - pip install requests
    - pip install bs4
    - pip install nltk
* 編譯方式: 這次共有三個檔案，HW2\_R07725021\_1.py是做將documents tokenize後計算每個term的df並儲存到dictionary.txt裡、HW2\_R07725021\_2.py是算出每個term的tf-idf unit vector並各別輸出到對應的檔案x.txt裡面、HW2\_R07725021\_3.py是根據前面tf-idf結果讓使用者輸入要比較的兩個document，算出這兩個document的cosine similarity並輸出到畫面上。所以，如只是要算出兩個檔案cosine similarity，只需在cmder(cmd)中輸入 **python HW2\_R07725021\_3.py**即可執行。  
  
* 輸出結果

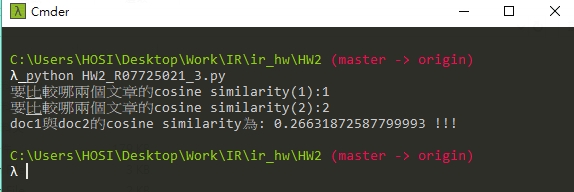
1. dictionary.txt (總共11678個term)



1. 1.txt (the vector file of document)



1. cosine similarity輸出在cmd上(document 1和document 2的cosine similarity)



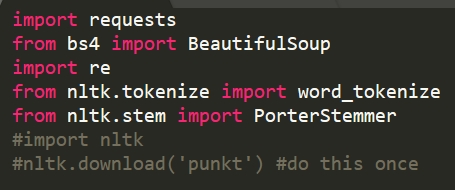
## 4.作業處理邏輯說明

因為有三個script，所以以下分別介紹

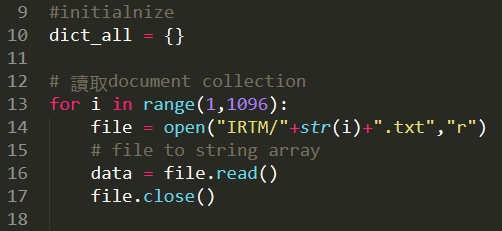
1. **HW2\_R07725021\_1.py**

運用HW1的tokenize方法，將document切割取得數個term，計算各個term的df值後輸出到dictionary.txt中。

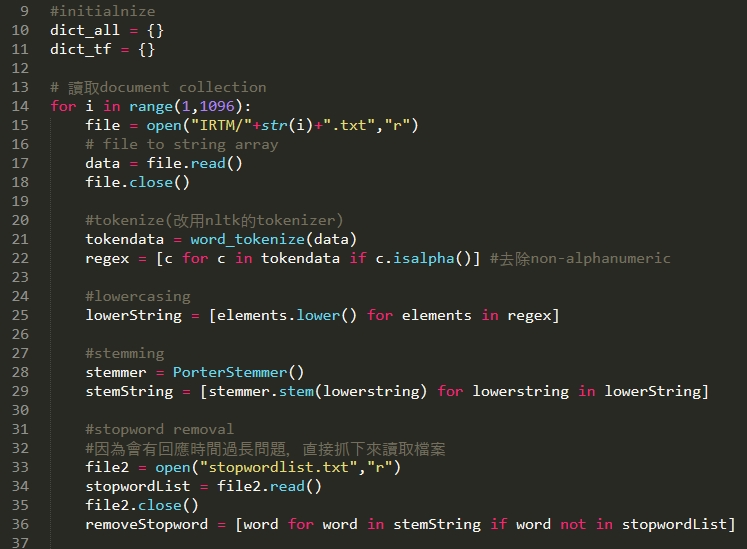
* Import必要的package



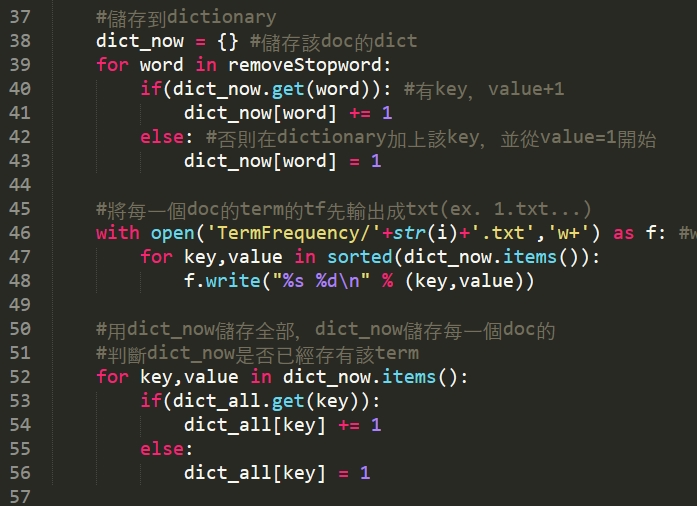
* 蒐集1095個document內容



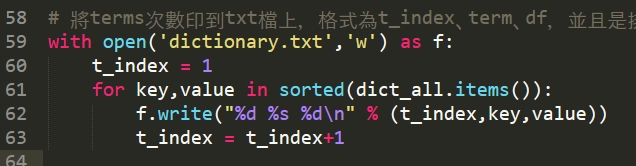
* 引用HW1 script來做tokenize，並改良了stopword removal方法(因為原先方法是去網路抓，這樣等網路回應太慢，所以改由抓下來儲存為stopwordlist.txt，讀取檔案來判斷)



* 用一個dictionary型態dict\_now儲存目前這個document的term和其tf(key為term，value為tf)，並將結果輸出在TermFrequency/x.txt中(為了第二個程式用的)。而後也用一個dictionary型態的dict\_all來儲存全部documents的term和df(key為term,value為df)，用來與dict\_now比較目前有沒有儲存dict\_now的term，如果有就在dict\_all的value加一，沒有就儲存該term。因此就能獲得1095 documents所有的term和總共出現次數df!

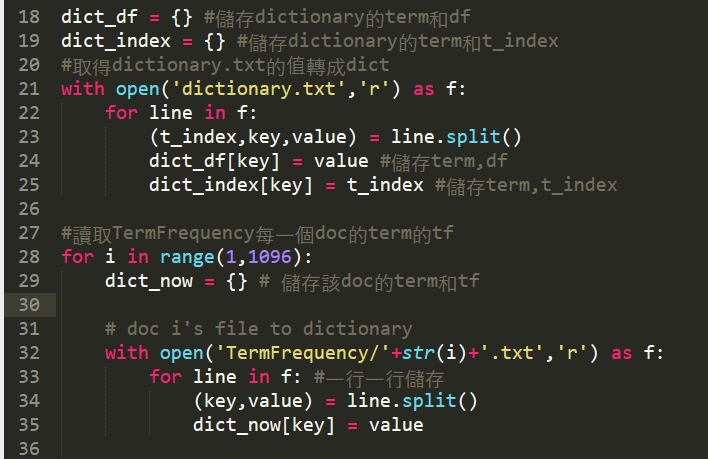
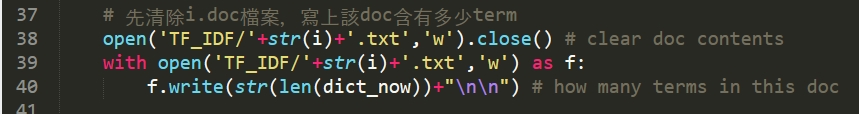


* 最終將dict\_now排序過，並加上t\_index(按照sort過的dictionary排順序，從1開始)以及term和df輸出到dictionary.txt檔案上，格式為t\_index、term、df。

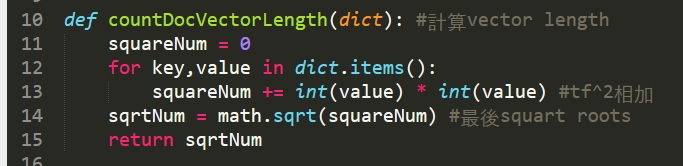


1. **HW2\_R07725021\_2.py**

這步驟主要做的是計算每個term的tf-idf值，並做normalizationize(除以vector length)，變成tf-idf unit vector.

* 讀取dictionary.txt裡面儲存的term和df值，用dict\_df儲存；另讀取TermFrequenct/各檔案裡面的term和tf值，用dict\_tf儲存。  
  
* 將準備輸出term和tf-idf結果的檔案TF\_IDF做清理(因為後面輸出方式是用append，所以需要在這裡清理)。先輸出該document有多少的term在第一行。  
  
* 定義一個function countDocVectorLength，輸入一個dictionary，計算該dictionary的vector length.

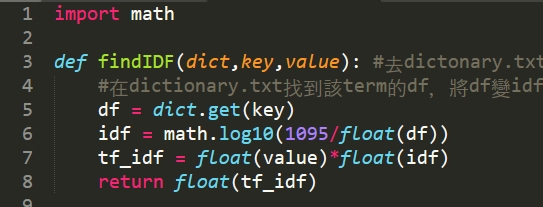
( |V(d1)| = sqrt(d1[0]^2+d1[1]^2+d1[2]^2+…d1[n]^2 )

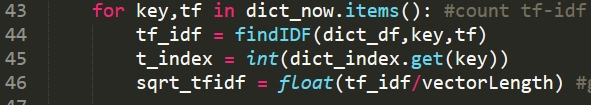




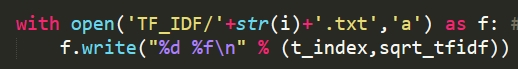
* 另外，import math，定義一個function findIDF，輸入dict\_df、term、tf，取得dict\_df的df值換成idf值(idf =log10(N/df))，與輸入的tf值後相乘得tf-idf值(tf\_idf=tf\*idf)。

得到tf-idf值後，做normalize，將tf-idf值除以前面得到的vector length。最終得到tf-idf unit vector.





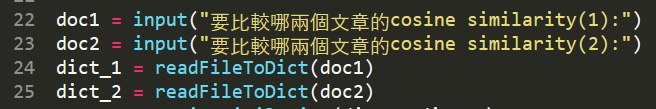
* 輸出結果到TF\_IDF/x.txt(各document檔案)，輸出t\_index、tf-idf unit vector

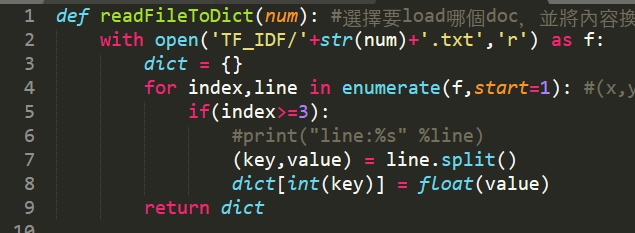


1. **HW2\_R07725021\_3.py**

算出每個document tf-idf值後，給user輸入要比較兩個document，得到這兩個document的cosine similarity，最後輸出在cmd上。

* 詢問要比較的2個document ID，定義一個function readFileToDict，輸入document的代號，從TF\_IDF/x.txt中取得該document的term、tf-idf值，並轉成dictionary格式回傳。





* 另外定義一個function cosineSimilarity，輸入兩個document的dictionary，比較兩個document是否含有相同的term，有則相乘tf-idf的值，最後把所有各term相乘的tf-idf值相加，得到這兩個document的cosine similarity值，輸出到cmd上。

