**106-1 資訊檢索與文字探勘導論 Programming Assignment 2 Report**

**R06725032 資管碩一 吳定寰**

1. 執行環境

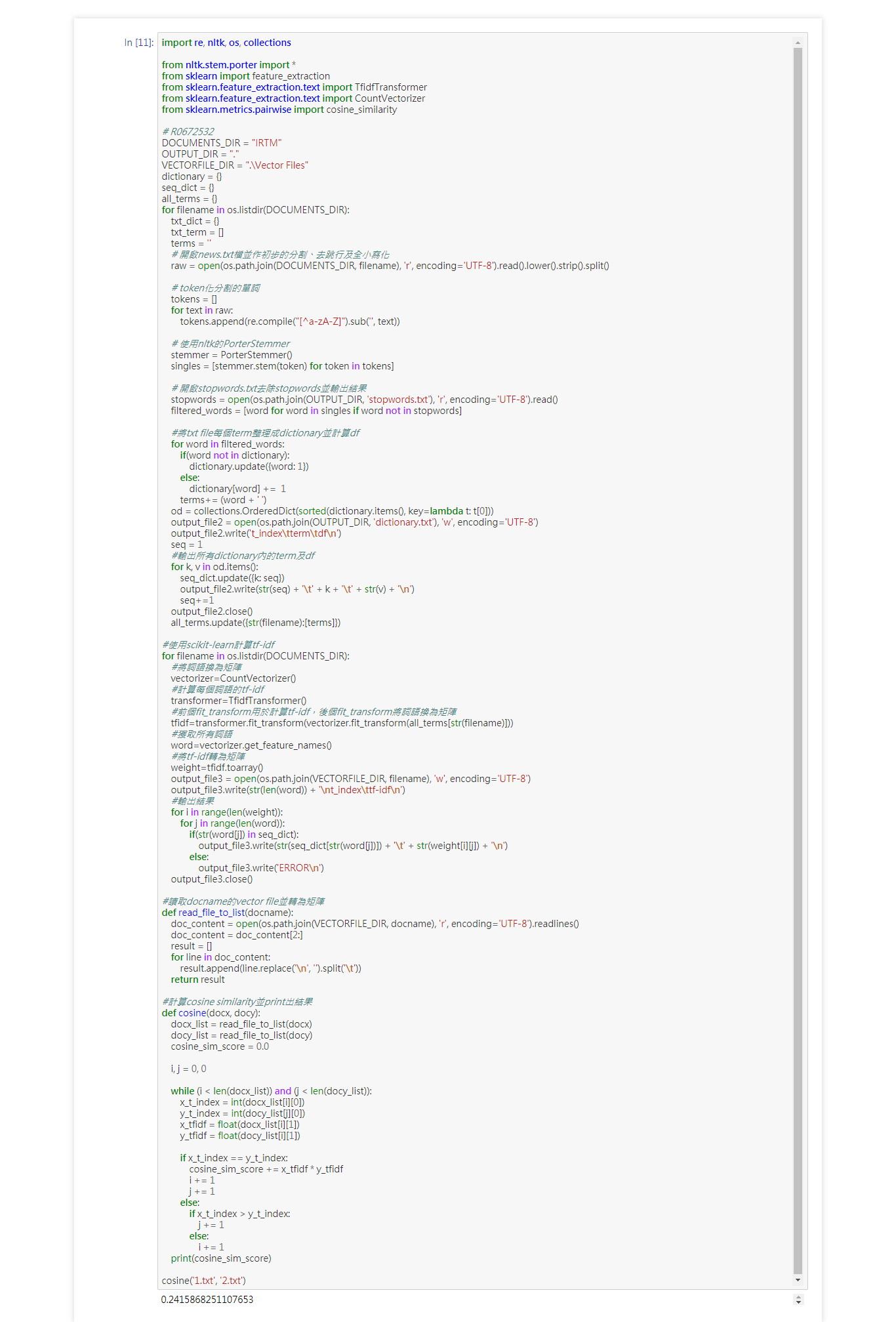
(.ipynb) Jupyter notebook

1. 程式語言與版本

Python 3.6

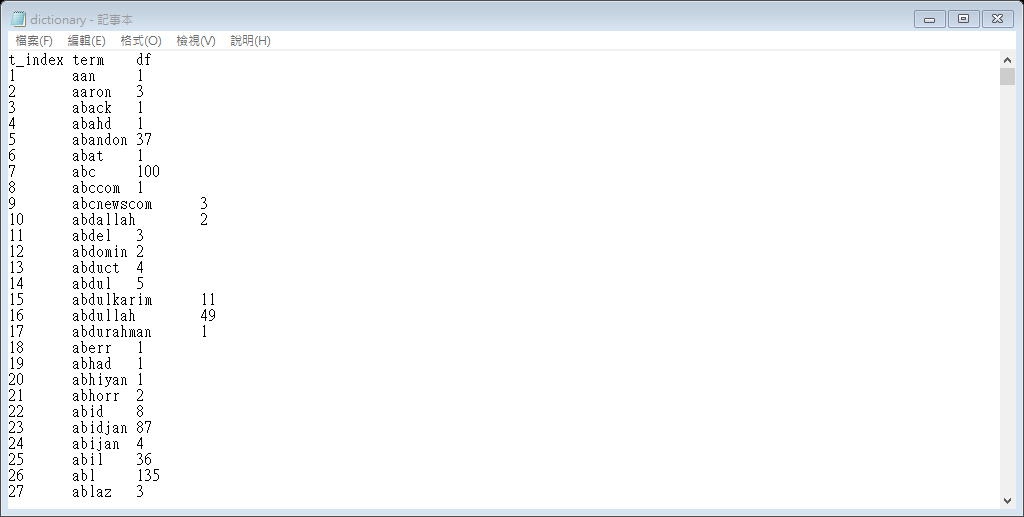
1. 執行方式及安裝套件
   1. pip install nltk, scikit-learn(需先安裝numpy及scipy)
   2. 執行方式：

直接在Jupyter內run cell即可，會在同資料夾內輸出dictonary.txt檔，在Vector Files資料夾內會輸出每個txt file的vector file，並會在cell的下方print出1.txt與2.txt的cosine similarity，約為0.2415868251107653，執行方式如下圖所示。



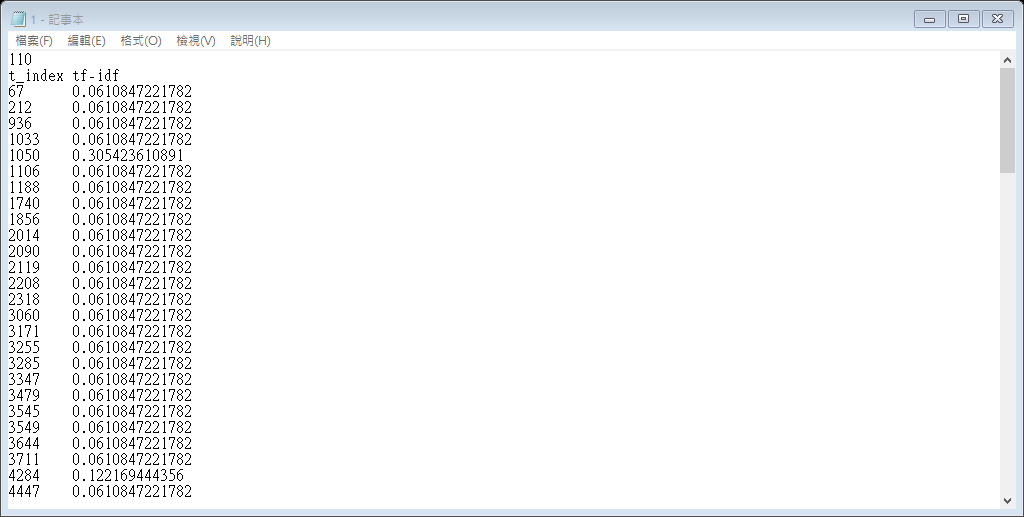
在Jupyter notebook開啟textProcessing.ipynb檔並執行

同資料夾內會輸出dictionary.txt檔



輸出之dictionary.txt

在Vector Files資料夾內會輸出各txt檔的vector file，以1.txt檔為例：



1. 作業處理邏輯說明

依序開啟IRTM資料夾內的\*.txt檔進行全小寫化、去跳行符號並依空白字元進行初步分割後，使用正則表示式留下符合規則的token，接著使用nltk套件的PorterStemmer，最後再藉由讀入stopword.txt比對來去除stopwords，將每個term整理為dictionary並計算df，待全部處理完成後，依term排序輸出成dictionary.txt檔。接著，使用scikit-learn套件計算tf-idf，並將其依照\*.txt命名規則在Vector files資料夾內輸出結果。最後，呼叫function cosine來計算1.txt與2.txt間的cosine similarity。Function cosine則是依照上課講義的邏輯，先使用function read\_file\_to\_list分別將1.txt及2.txt轉為矩陣後進行計算，當x\_t\_index==y\_t\_index時進行計算，不相等時則依情況將i+1或j+1，最後再print出。

1. 心得

此次作業延續PA-1，整理成dictionary.txt還算容易，但後面計算tf-idf的部分則不是很簡單，要找到快速的方法還要理解過程是需要一些時間的，function cosine則是依照講義和上課提到的邏輯進行即可。