資訊檢索與文字探勘導論

hw2 Report

B03705012 張晉華

1. 程式語言

Python 3.5.2

2. 執行環境

Linux OS (Ubuntu 16.04 LTS)

Python3 Packages Requirements:

- Python Natural Language Toolkit(nltk 3.2.2)
- Python Numpy(1.13.3)

3. 執行方式

- **Package Installation**
  - Python Natural Language Toolkit(NLTK)

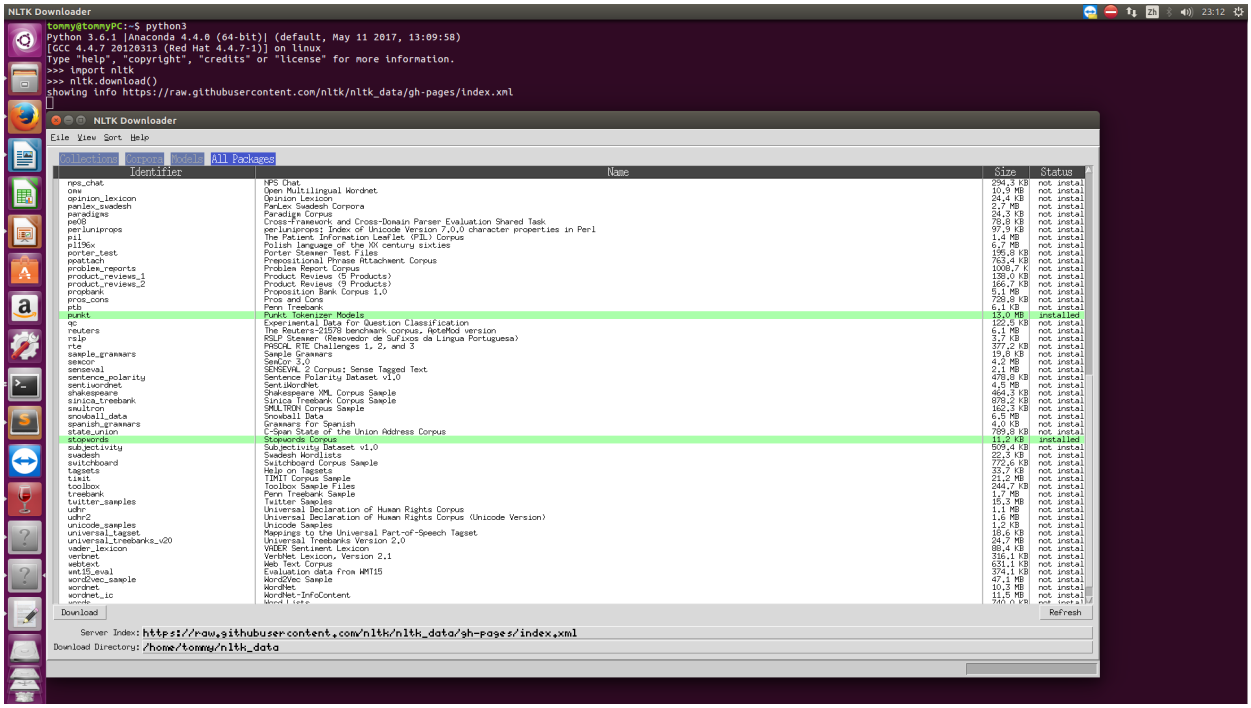
    Step1：Install Packages

    (可以透過 pip3 install nltk 安裝)

    Step2：Corpus Download

    有兩個 Corpus 需要下載以供程式使用：punkt(tokenize 需要)和 stopwords

    可以在 python3 環境下執行 nltk.download()下載



- Python Numpy：可透過 pip3 install numpy 安裝

- **執行指令**
  - cd 進程式所在路徑，在 termial 執行 python3 my_dictionary.py，檔案結果產出在同一路徑的 dictionary.txt 和資料夾 vectors/，內有所有文件的 tf-idf unit vector 的.txt 檔，然後再執行 python3 my_cosine.py $1 $2($1,$2 為要計算的文件編號)，產出結果為兩篇文章的 cosine similarity 印在 terminal 上(執行截圖見下面)
  - 可以用 chmod +x hw2.sh 指令讓 hw2.sh 改為可執行權限後執行./hw2.sh $1 $2($1,$2 為要計算的文件編號)，會一並執行上述動作，並將兩篇文章的 cosine similarity 印在 terminal 上。

4. 作業處理邏輯說明

- **my_dictionary.py**

  Step1：利用 hw1 的程式（此處改名為 my_tokenizer.py）將每一篇文章做 tokenize 並丟進 set 裡，計算 df 並合併到 dictionary 中

  Step2：將字典 sorted，並輸出到 dictionary.txt

  Step3：重新讀取每一篇文章做 tokenize，計算每個 term 在每一篇的出現次數（即為 tf）

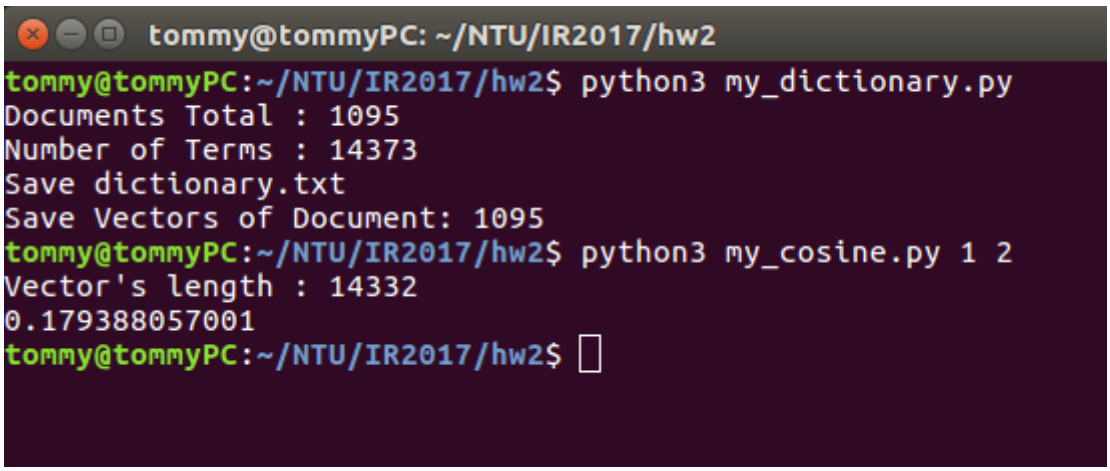
  Step4：計算 tf-idf，轉成單位向量後儲存進 vectors/內。

- **my_cosine.py**

  Step1：將兩篇文件編號所對應到 vectors/內的檔案讀進來，取兩篇最後一個 t_index 較大者為向量長度，再來將檔案內有紀錄的 tf-idf 值依照 t_index 一一對照進來，其餘補 0，如此將兩篇文件各自做成一個 numpy array。

  Step2：利用 numpy.dot()計算兩個文件向量的內積並輸出。

  執行截圖如下圖：



5. 心得

這次的作業我覺得最有趣的地方是在於非字母的處理部份，其實自己在處理時這類的 term 時其實掙扎了許久，一直不知道該不該保留，後來跟同學討論後覺得，數字如果沒有跟著單位在一起其實沒有辦法有明確的意義，所以決定在做 tokenize 的時候多加上一個 regular expression : "[0-9]*[a-zA-Z]+"來做篩選，篩掉單獨的數字只留下有跟著單位的數字（例如 16yearold），而在這不斷的嘗試過程中也學到了不少相關的知識。