文本分析與數位人文 書面作業一

寶玉、黛玉在《紅樓夢》內出現過幾次、笑過幾次？

資科碩專班 105971001 杜若宇

大綱

- 問題
- 腦力激盪
- 實際作法
- 執行結果
- 討論
- 心得
- 本次作業google drive連結：https://goo.gl/rwvNHm

問題

1. 賈寶玉在這一份《紅樓夢》中出現幾次？林黛玉幾次？
2. 賈寶玉在這一份《紅樓夢》中笑過幾次？林黛玉幾次？

本次答案：

	出現次數	笑過次數
賈寶玉	3965	271
林黛玉	1369	111

腦力激盪

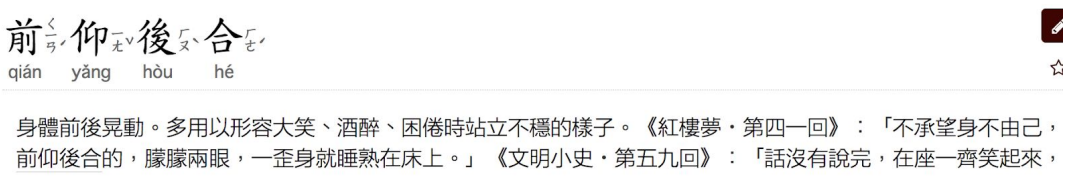
這是我第一次處理文本，首先拿到需要被分析的素材為 120份txt檔案，檔案名稱是有序的漢字，設計思路如下：

1. 首先嘗試擷取一份**非漢字**檔名的txt檔的資料，計算「出現次數」。
2. 擴增到多份**非漢字**檔名。
3. 能偵測並擷取漢字檔名。
4. 計算「笑的次數」。可不可以不用正則表達法。
5. 重新檢查文本，找出提升精確程度的方法。

實際作法

實際操作時，發現這不只是單純寫程式的工作。必須要程式、文字觀察雙邊交互進行、交互檢查、重新設計，方能得到較準確的數據。實際設計步驟如下：

1. 首先嘗試擷取一份**非漢字**檔名的txt檔的資料。
2. 計算「出現次數」。
3. 回頭觀察文章，避免誤差擴大。
    a. 必須去除標題。
    b. 僅用「黛玉」做偵測，不考慮姓。
4. 擴增到多份**非漢字**檔名。
    a. 首先嘗試：擷取整個資料夾的文檔，但如此會減少彈性。
    b. 後來採用for 迴圈，較冗長的方法。
5. 能偵測並擷取漢字檔名。
    a. 處理漢字和阿拉伯數字的互換。
    b. 既然能互換，同時嘗試增加可指定回數。
6. 增加計算「笑的次數」。
    a. 觀察辭典：大部分的「笑」文字表達方式都會內含「笑」字，可以直接粗略使用。
    b. 開始使用Regex，初步想法是篩選出「黛玉+笑」。
7. 重新檢查文本，找出提升精確程度的方法。
    a. 為求「笑」的精準，找了同義詞詞典：東東同義詞典 http://www.hkdictionary.net/synonym/result2.asp?Sense=%AF%BA
    b. 沒有笑字但也是笑：「莞爾」、「嫣然」「忍俊不禁」「春風滿面 」「喜逐顏開 」「哂」「 捧腹 」「絕倒」「喜形於色」「開顏」。
    c. 有爭議：「噱 」、「樂」、「前仰後合」、「粲」。參考：萌典 https://www.moedict.tw/%E7%B5%95%E5%80%92



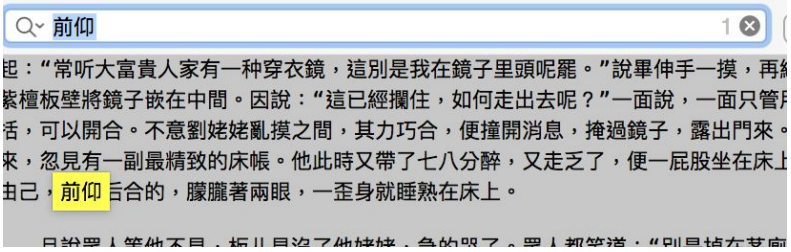
8. 乾脆設計三種系統：
    a. 簡易偵測：僅包含「笑」字。
    b. 狹義偵測：包含 7b 種類的笑。

c. 廣義偵測：包含 7b、7c 種類的笑。

d. 為了方便測試每一種笑，首先增加可以自由找字的功能。

9. 檢查文本，發現陷阱，這文本不單純是全繁體字的。

10. 但經過測試，這些「笑」和黛玉、寶玉都沒有連結。乾脆不做第8項次。



執行結果

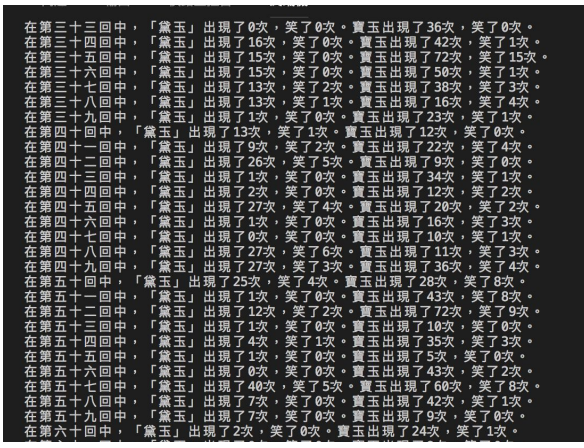
增加功能及解說：

1. 可以選擇查詢的起始回數、終止回數。如圖一。
2. 可以顯示出每一回的出現次數、笑過次數。如圖一到四。
3. 計算總結果如圖四。
4. 線上程式碼：https://github.com/RenoDououi/DH

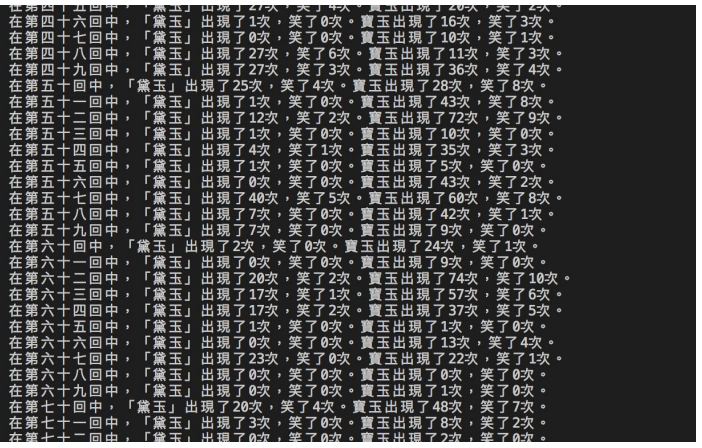
```
utput,RedirectOutput /Users/kakitsubatasakai/Desktop/DH/DH-hw1-105971001.py
歡迎來到：「黛玉寶玉呵呵笑計數器：」請問您想要從哪一回開始查呢？：
1
想要查到哪一回？：
120
在第一回中，「黛玉」出現了0次，笑了0次。寶玉出現了1次，笑了0次。
在第二回中，「黛玉」出現了1次，笑了0次。寶玉出現了2次，笑了0次。
在第三回中，「黛玉」出現了76次，笑了1次。寶玉出現了32次，笑了3次。
在第四回中，「黛玉」出現了3次，笑了0次。寶玉出現了0次，笑了0次。
在第五回中，「黛玉」出現了10次，笑了0次。寶玉出現了61次，笑了0次。
在第六回中，「黛玉」出現了0次，笑了0次。寶玉出現了13次，笑了0次。
在第七回中，「黛玉」出現了5次，笑了0次。寶玉出現了28次，笑了1次。
在第八回中，「黛玉」出現了24次，笑了4次。寶玉出現了84次，笑了8次。
在第九回中，「黛玉」出現了4次，笑了0次。寶玉出現了35次，笑了2次。
在第十回中，「黛玉」出現了0次，笑了0次。寶玉出現了5次，笑了0次。
在第十一回中，「黛玉」出現了0次，笑了0次。寶玉出現了10次，笑了0次。
在第十二回中，「黛玉」出現了3次，笑了0次。寶玉出現了1次，笑了0次。
在第十三回中，「黛玉」出現了2次，笑了0次。寶玉出現了11次，笑了1次。
在第十四回中，「黛玉」出現了0次，笑了0次。寶玉出現了15次，笑了1次。
在第十五回中，「黛玉」出現了0次，笑了0次。寶玉出現了57次，笑了6次。
在第十六回中，「黛玉」出現了7次，笑了0次。寶玉出現了25次，笑了0次。
在第十七回中，「黛玉」出現了0次，笑了0次。寶玉出現了47次，笑了0次。
在第十八回中，「黛玉」出現了16次，笑了0次。寶玉出現了41次，笑了2次。
在第十九回中，「黛玉」出現了33次，笑了4次。寶玉出現了116次，笑了16次。
在第二十回中，「黛玉」出現了23次，笑了1次。寶玉出現了55次，笑了8次。
在第二十一回中，「黛玉」出現了14次，笑了0次。寶玉出現了48次，笑了3次。
在第二十二回中，「黛玉」出現了29次，笑了2次。寶玉出現了42次，笑了2次。
在第二十三回中，「黛玉」出現了18次，笑了3次。寶玉出現了54次，笑了3次。
在第二十四回中，「黛玉」出現了6次，笑了0次。寶玉出現了40次，笑了5次。
在第二十五回中，「黛玉」出現了27次，笑了3次。寶玉出現了51次，笑了0次。
在第二十六回中，「黛玉」出現了17次，笑了0次。寶玉出現了70次，笑了6次。
在第二十七回中，「黛玉」出現了13次，笑了0次。寶玉出現了27次，笑了5次。
在第二十八回中，「黛玉」出現了46次，笑了2次。寶玉出現了97次，笑了10次。
在第二十九回中，「黛玉」出現了35次，笑了1次。寶玉出現了43次，笑了2次。
在第三十回中，「黛玉」出現了27次，笑了1次。寶玉出現了70次，笑了5次。
在第三十一回中，「黛玉」出現了17次，笑了4次。寶玉出現了60次，笑了11次。
在第三十二回中，「黛玉」出現了16次，笑了0次。寶玉出現了31次，笑了4次。
```

圖一：選擇功能、第一回至第三十二回次數統計。

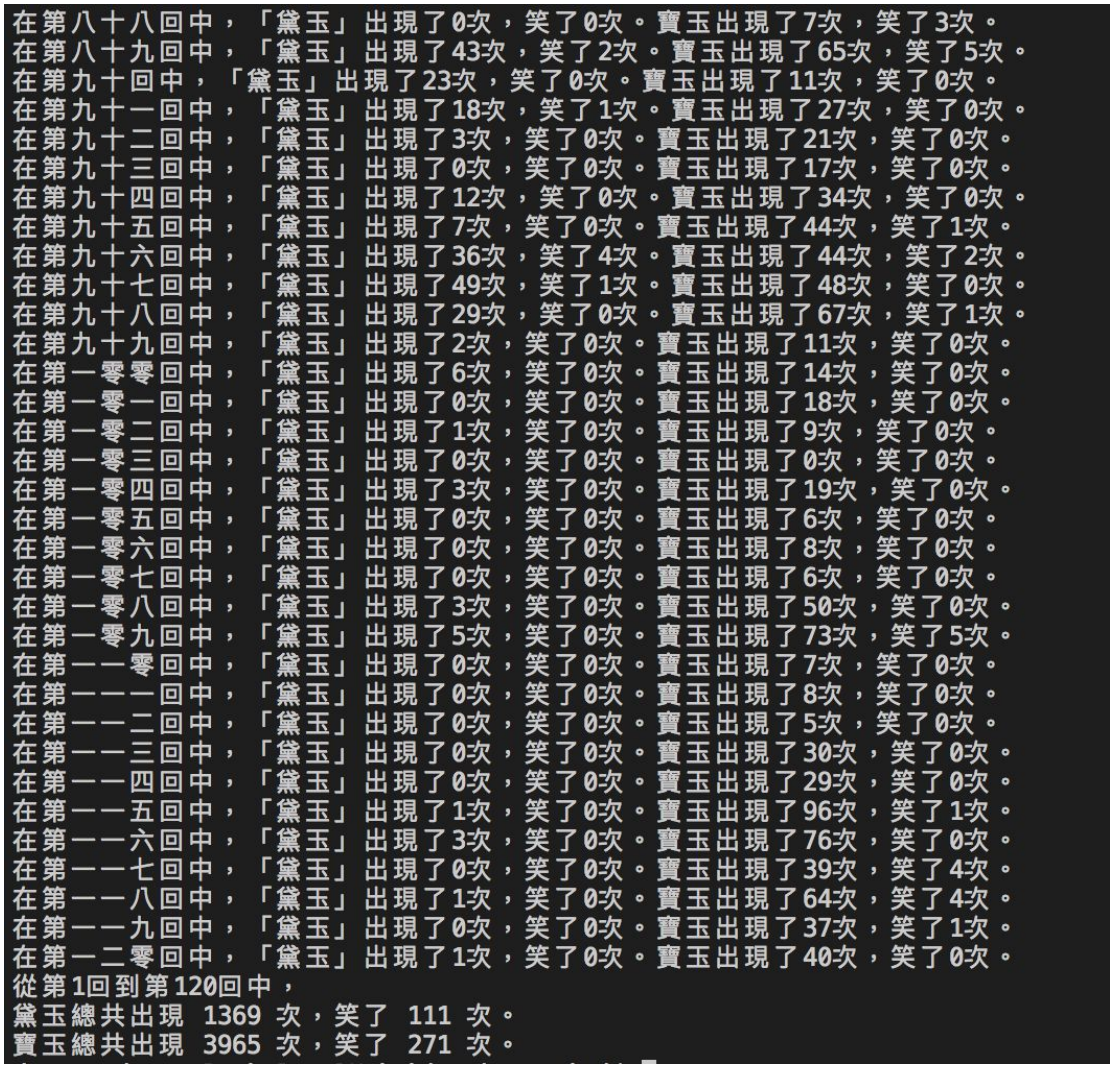
接下一頁



圖二：第三十三回到第六十回。



圖三：第六十一回至第八十八回。



圖四：計算總結果、第八十八回至第一二零回。

接下一頁

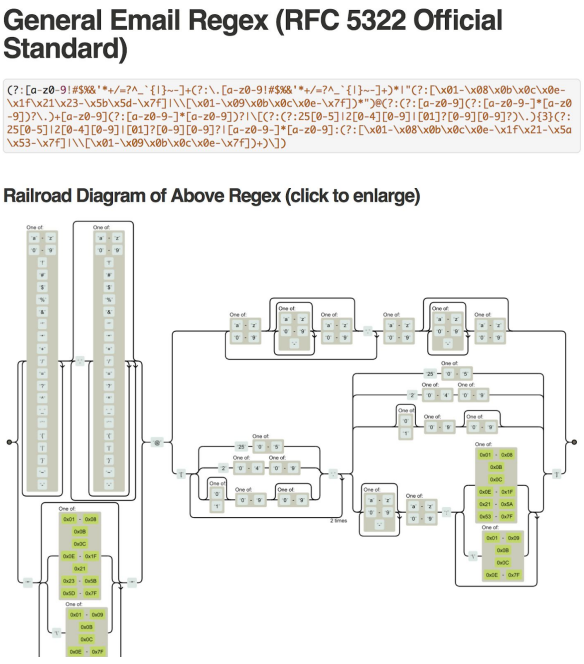
討論

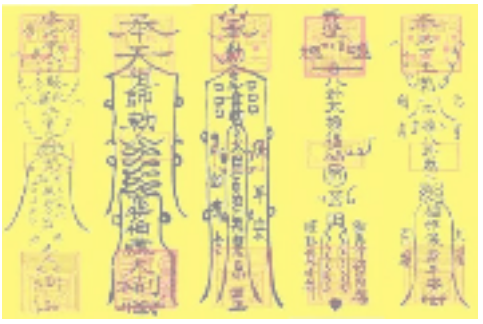
文本處理最難的地方，如同老師所說，人腦和機器記憶方式是不同的，擅長處理的問題也不盡相同。要拿到精確的答案，並不是一件容易的事情。如何找到一個通用的辦法去擷取想要的資料，可以是一門單獨得學問。必須要同時對文學、正則表達、資訊科學有基礎認識。

幾經掙扎，考慮不使用Regex 應該也是能達到目標。於是首先嘗試不使用Regex 做做看實驗。

同樣的，要考慮的點也是不少。在Mac 作業系統中，編碼為人詬病，正不巧學生不曉得，經過一段時間，同學指出問題之後才發現問題。同樣的，一開始偵測檔案名稱就遇到一些小問題：這份檔案的漢字檔名排列順序，跟法語實在有點像，1至10一組，11-20一組，21-99一組，100以上一組。之後的程式邏輯反而不是比較困難的地方。

這次最經典的地方在於在思考「如何使用良好的Regular expression」的同時，我溜覽一下業界如何處理實務，發現了幾個像是天書一樣的Regex，最有名的似乎就屬「判斷是否為email格式」了（如：http://emailregex.com/ ），擷取一部分，見下圖。





我的天，這根本就是道符。字字看得懂，組合起來還真是頭大。這下不得了，不畫 Railroad 還真的很不容易寫出這種Regex。

由此可見，要如何定義「笑」這個動作？並非只有「笑」字，「呵」、「呵呵」要怎麼去區別。再者，林黛玉在笑的時候，如果緊接著還有一個人笑，兩個人都是笑「呵」，我們必須處理的事情就更多了。

心得

這是我第一次使用Python處理文本，在此之前也沒有學過 regular expression。因為工作關係，將近四年沒有使用過Python，重回碼農，看到成品，有種說不出的成就感。

學生的領域比較特殊，電機系畢業，後簽約為軍人，因為軍中電機類武器多從法國製造，奉派學習法文，接觸翻譯，重新認識了中文。

部分法語詩句近乎無法翻譯成中文，在翻譯的過程中，必定會損失原著的特別用意，需要註解。最近翻譯界非常熱門的金庸，就算譯者通曉雙邊文化，因為金庸的時有半文言，時有方言，難以翻譯。楊鐵心翻譯為Ironheart Yang 隱姓埋名後，化名穆易，譯為Mu Yi，英文讀者沒辦法知道Yang 和 Mu Yi 的關係。

儘管許多狀況難以翻譯，但我們還是能盡力精進一般生活中的翻譯，甚至到商務翻譯，消彌世界的商務障礙，至少旅遊翻譯不要再翻錯了，我覺得這是大有可能的。

在學生去年至大陸背包旅遊時，才知道買火車票輸入「黃」山是找不到的，必須打「黄」山。 實用上，大陸不太需要去確認其他漢字的Regex。如同「絲」一字，在大陸、台灣、日本的寫法不同，但差異甚小(見 :https://www.jcinfo.net/tw/tools/kanji)，打字的時候挑錯，當下也沒發現，可能就會造成鬼打牆的結果。彥彦兩字也是個好例子，在輸入名字時偶爾會打錯，造成找不到病例之類的狀況。我相信這是很容易解決的問題，但需要時間去推廣。

經過實作，加上在網路的反覆搜尋，特別是查到「判斷email 格式」的Regex 、幾件漢字（繁體、簡體、和製）的例子後，我認為還會有非常多的挑戰。尤其是中文對英文的翻譯。更精確的說是漢語對日耳曼語系/拉丁語系等西方語系的翻譯，會遇到的窒礙肯定更多。希望在未來可以發現更多的問題以及解決方案。