國立高雄師範大學

軟體工程系

專題製作報告

以 TF-IDF 演算法實作熱門歌手之歌詞分析視覺化應用

指導教授：李文廷 博士

學　　生：鍾弘浩　　撰

中 華 民 國 111 年 9 月

目　　錄

壹、前言

一、研究動機

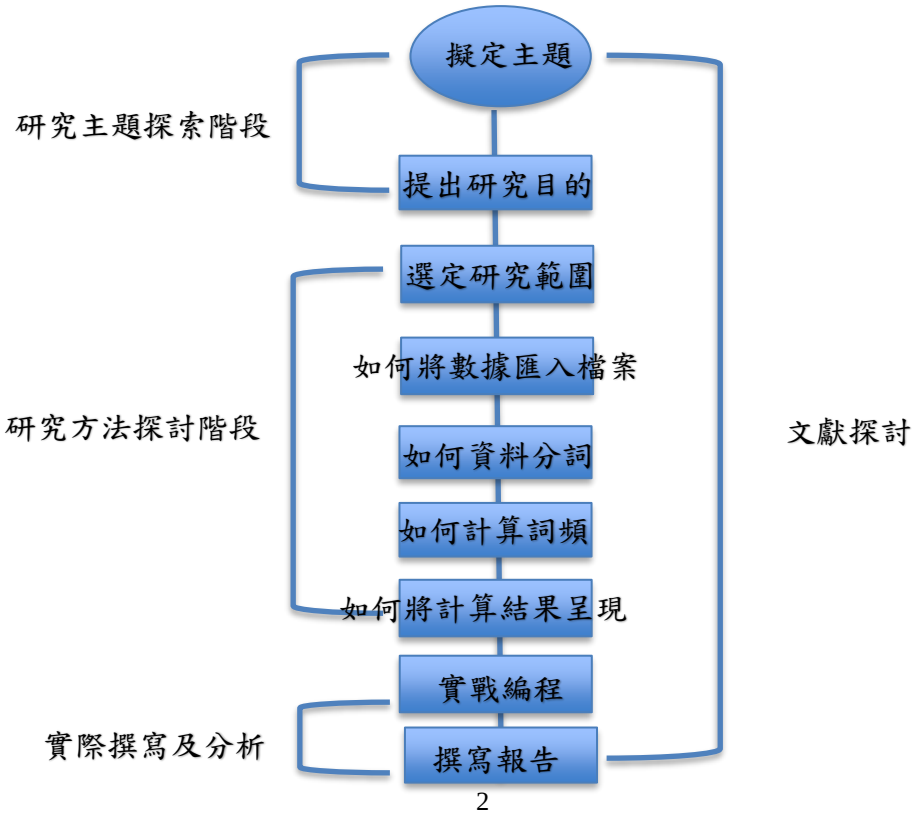
在這個瞬息萬變的時代，每天都有源源不絕的資訊產生，為了能夠善用這些資訊來協助我們管理以及優化工作及生活，可以透過量化一切來設定標準。因此，大數據分析不再單單只是一項技術，而是在這個資訊爆炸的時代裡，讓我們能掌握商機的好工具。阿里巴巴創辦人馬雲曾大膽指出，未來大數據將會比石油還貴，成為最有價值的事物，然而我自己很喜歡聽歌，我就想到大數據資料分析應用，將使用 Python 實作 tf-idf 演算法，利用網路爬蟲取得資料來源，再利用數據分析來完成乾淨的資料，接著利用演算法量化歌詞中出現的次數，再使用視覺化讓人一看就明白甚麼詞彙出現的頻率最高。

二、研究目的

（一）利用 tf-idf 來量化語詞次數來衡量關鍵字。我們以周杰倫、五月天、林俊傑 三位歌手的歌詞為例。

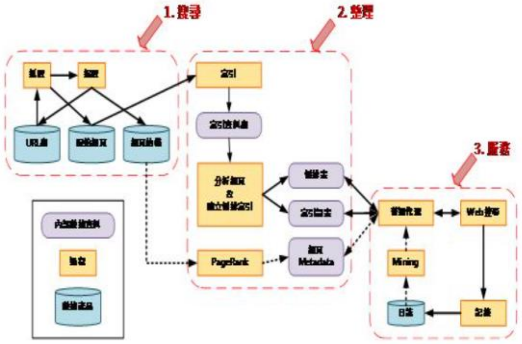
（二）利用 tf-idf 技術，以視覺化的方式淺顯易懂，一看就知道這些歌手的特色。

三、研究流程



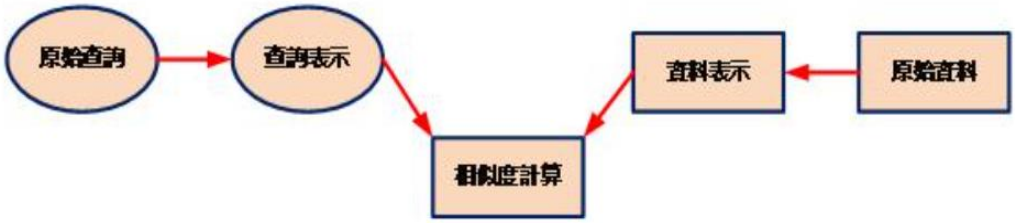
貳、文獻探討

一、搜尋引擎與文字探勘兩者之間的關係(來源:台大:計算機與資訊網路中心)

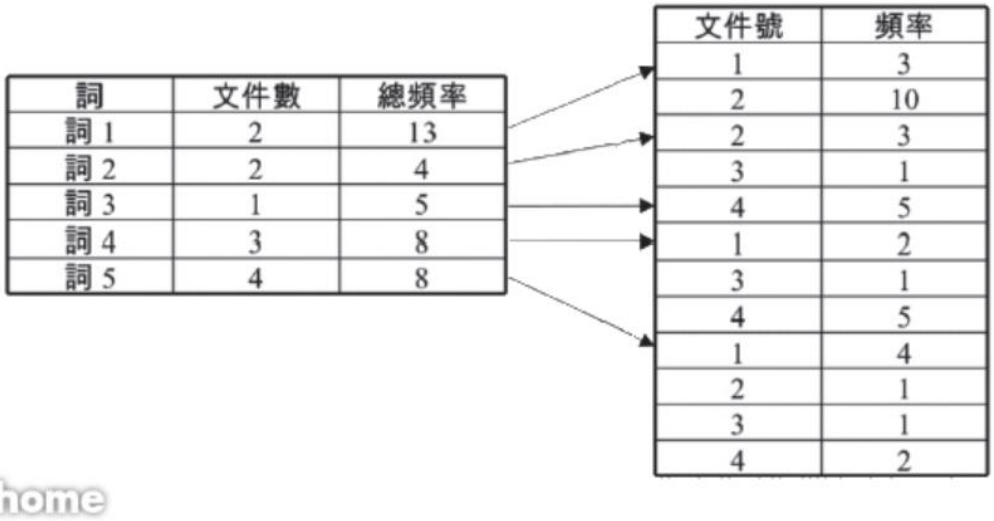


圖一

二、使用者將文件或文字透過搜尋引擎進行索引之後，存放於索引資料庫中，累積成文字探勘的基礎。搜尋引擎不僅提供全文搜尋、模糊搜尋、關鍵字搜尋等功能，在我們需要做文字探勘時，也可以以搜尋引擎為基礎發展出多種文字探勘的功能(來源: 台大:計算機與資訊網路中心)



三、將 5 個詞在不同文件出現頻率的權重關係結構呈現上，改以反向索引的方式呈現。(來源:iThome)



詞	文件數	總頻率
詞 1	2	13
詞 2	2	4
詞 3	1	5
詞 4	3	8
詞 5	4	8

文件號	頻率
1	3
2	10
2	3
3	1
4	5
1	2
3	1
4	5
1	4
2	1
3	1
4	2

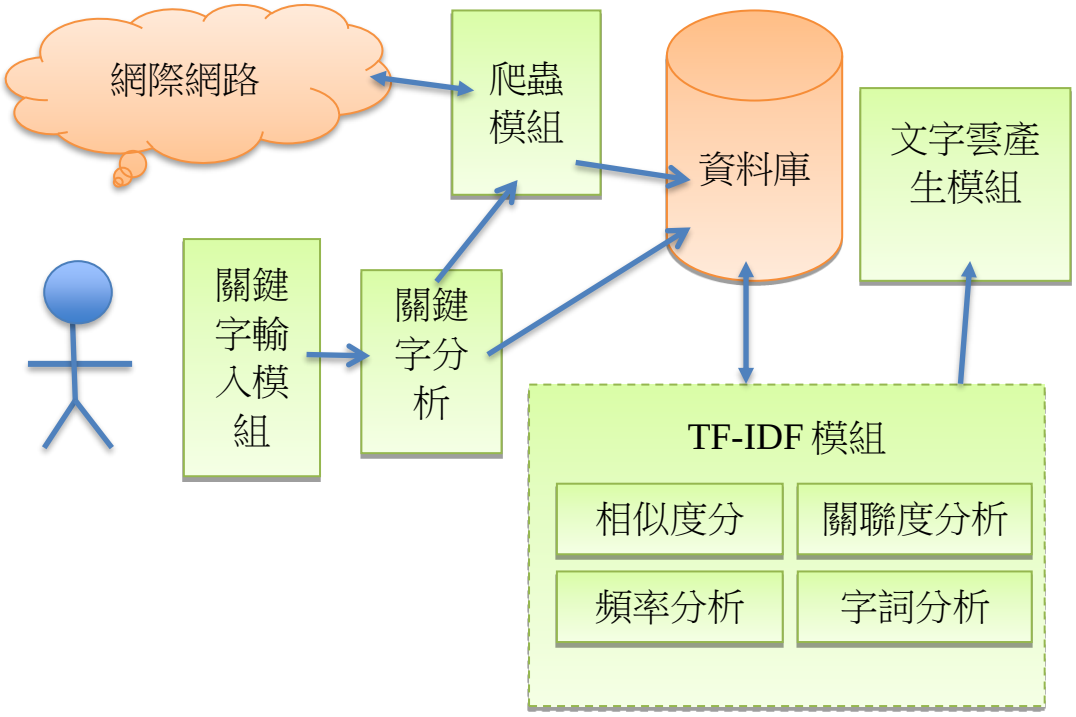
四、權重應該使用什麼表示方式呢？最直覺的方式就是使用該詞在文件中所出現的次數。因此，倘若我們有四份文件，這五份文件裡總共出現了五個不同的詞，那麼就分別將它們表示成為四個向量，每個向量中的元素，其權重為該對應之詞出現於該文件中的次數。(來源:iThome)

文件號	詞 1	詞 2	詞 3	詞 4	詞 5
1	3	0	0	2	4
2	10	3	0	0	1
3	0	1	0	1	1
4	0	0	1	5	2

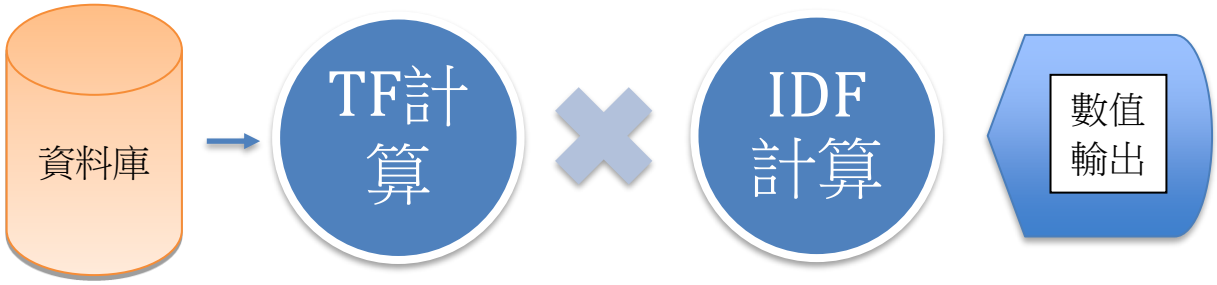
五、CKIP CoreNLP（來源:中研院）

參、研究方法

一、系統架構



2. tf-idf 介紹



（一）詞頻（term frequency，tf）

$$\text{詞頻} = \frac{\text{該語詞在文本中的出現次數}}{\text{文本中所有語詞的出現次數之和}}$$

（圖二）

（二）逆向文件頻率（inverse document frequency，idf）

$$\text{逆文件頻率} = \log_{10}\left(\frac{\text{文本的數量}}{\text{包含該語詞的文本數量}}\right)$$

（圖三）

例如: 一個文檔有 1 萬個字，其中包含 donkey 這個詞有 300 個詞，因此

$$tf = 300 / 10000 = 0.03$$

假設文檔現在有 1000 個，其中有 10 個包含 doukey 這個文本，

$$idf = \log10(1000 / 10) = 2$$

因此為

$$tf\text{-}idf = 0.03 \times 2 = 0.06$$

這邊寫了一個範例去舉例 tf-idf 的計算過程以及得出的數據

```
 13
 14    # Documents
 15    doc_0 = 'Today is a nice day'
 16    doc_1 = 'Today is a bad day'
 17    doc_2 = 'Today I want to play all day'
 18    doc_3 = 'I went to play all day yesterday'
 19    doc_all = [doc_0, doc_1, doc_2, doc_3]
 20
 21
 22    # TF-IDF
```

問題　輸出　偵錯主控台　終端機　JUPYTER

```
1646

[4 rows x 11 columns]
                0         1         2         3
all          0.000000  0.000000  0.707107  0.707107
bad          0.000000  1.000000  0.000000  0.000000
day          0.588348  0.588348  0.392232  0.392232
is           0.707107  0.707107  0.000000  0.000000
nice         1.000000  0.000000  0.000000  0.000000
play         0.000000  0.000000  0.707107  0.707107
to           0.000000  0.000000  0.707107  0.707107
today        0.639602  0.639602  0.426401  0.000000
want         0.000000  0.000000  1.000000  0.000000
went         0.000000  0.000000  0.000000  1.000000
yesterday    0.000000  0.000000  0.000000  1.000000
```

3.程式片段說明

（四）資料來源:利用 Beautiful soup 套件將資料從魔鏡歌詞網爬出

```
pipenv install beautifulsoup4
```

註解:BeautifulSoup 是一個 Python 的函式庫，可以從 HTML 或 XML 檔案中分析資料，也可拿來修復未閉合標籤等錯誤的文件。

五月天(Mayday)

派對動物

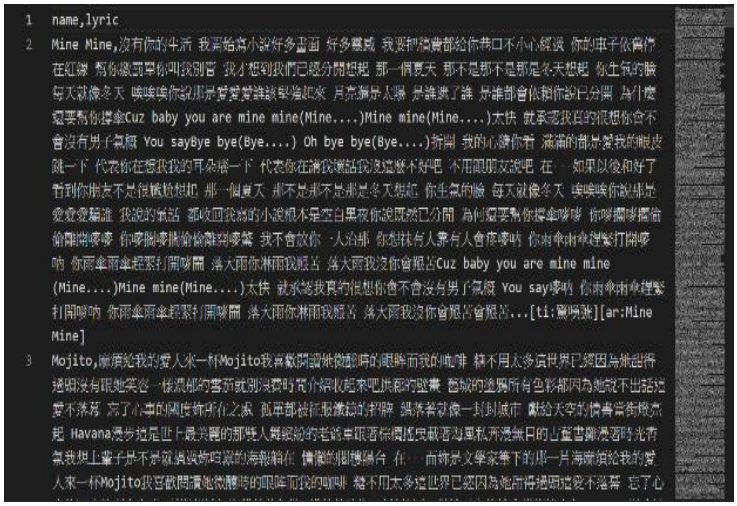
評論(0)　修改　小　中　大　巨

作詞：阿信
作曲：阿信
編曲：潘信維 Lil Pan、錢誠、Dj Ground、五月天 Mayday

Let's go party party all night, oh oh oh oh
Hey lonely lonely goodbye, oh oh oh oh
我們都有覺悟 要瘋狂到日出
我們 天生 就是 派對動物

（圖四）

（五）存入 csv



（圖五）

（六）讀取資料集

```
for filename in filenames:
    path = f"{storage_folder}/{filename}"
    with open(path,encoding="utf-8") as file:
        #每位歌手的歌詞以字串形式連接
        lyric = ""
        for i,line in enumerate(file.readlines()):
            if i==0:continue
            title, content = line.split(",",1)
            lyric += content
        singer = filename.split(".")[0]
        singers[singer] = lyric
```

（圖六）

歷遍 clean data 內所有歌手的歌詞，每位歌手的歌詞以字串形式連接，裝納進 singers 這個字典中，得到的結果會如下圖所示。

Key	Type	Size	Value
Jay Chou	str	1	沒有你的生活 我開始寫小說好多畫面
JayJay	str	1	狹小的廚房擁擠的沙發竟然能在你走後
Mayday	str	1	天再沒有時間 能去延後再沒有後路 能

（圖七）

(七)以 jieba 套件分詞

jieba 中文斷詞所使用的演算法是基於 Trie Tree 結構去生成句子中中文字所有可能成詞的情況，然後使用動態規劃（Dynamic programming）算法來找出最大機率的路徑，這個路徑就是基於詞頻的最大斷詞結果。對於辨識新詞（字典詞庫中不存在的詞）則使用了 HMM 模型（Hidden Markov Model）及 Viterbi 算法來辨識出來。

read_dictionary:

讀取字典，除了 jieba 內建字典，我尚且手動新增了字典在 dictionary 資料夾內。

read_stop_words:

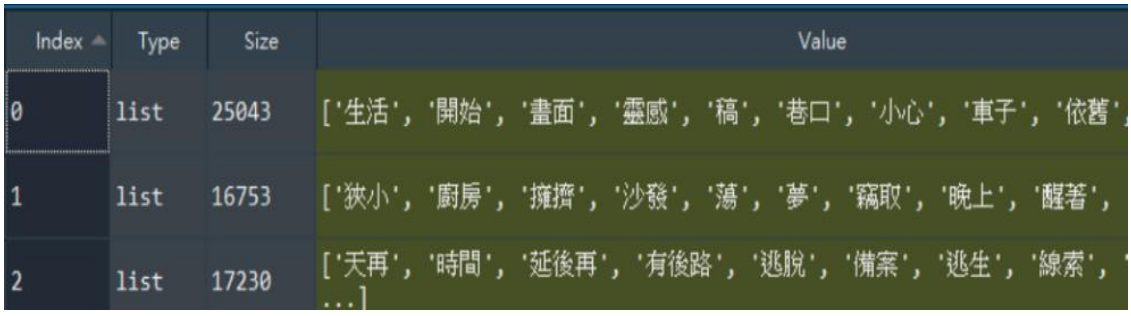
讀取停用詞，停用詞是不必要、無意義的語詞，比如說『然後』、『因此』、『了』等語助詞。

remove_stop_words:

移除上述讀取的停用詞。

```
25  ws = WordSegment(singers.values(),stop_words_path=stop_words_path)
26  ws.read_dictionary()
27  ws.read_stop_words()
28  ws.remove_stop_words()
```

（圖七）



Index	Type	Size	Value
0	list	25043	['生活', '開始', '畫面', '靈感', '稿', '巷口', '小心', '車子', '依舊',
1	list	16753	['狹小', '廚房', '擁擠', '沙發', '蕩', '夢', '竊取', '晚上', '醒著',
2	list	17230	['天再', '時間', '延後再', '有後路', '逃脫', '備案', '逃生', '線索', ...

（圖八）

(八)統計詞頻

```
32    words_count = []
33  for file in text_files:
34      count = {}
35      for word in fil  (variable) count: dict
36          if word in count:
37              count[word] += 1
38          else:
39              count[word] = 1
40      words_count.append(count)
41
```

(圖九)

words_count 是列表，依序代表周杰倫、林俊傑、五月天，列表內是字典，表示歌手語詞的次數，有了次數之後便能統計 words_frequency 頻率。

(圖十)

(九)統計逆文件頻率

occurrences_of_word 是一個字典，鍵值是語詞，值則是該語詞在幾個歌手歌詞中出現過，比如說『中世紀』語詞只有周杰倫的歌詞才有，映射關係就是{"中世紀":1}，得到這層映射關係，我便可用迴圈逐一取得歌手歌詞的語詞，除以文件數量(也就是歌詞文本數量，在此是 3)，再將結果取以 10 為底的對數。

```
53  all_words = []
54  for word in words_count:
55      all_words.extend(list(word.keys()))
56
57  occurrences_of_word = {}
58  for word in all_words:
59      if word in occurrences_of_word:
60          occurrences_of_word[word] += 1
61      else:
62          occurrences_of_word[word] = 1
63
64  inverse_document_frequency = []
65  for word_count in words_count:
66      #出現過的次數
67      invFre = {}
68      for word in word_count.keys():
69          occurrences = occurrences_of_word[word]
70          invFre[word] = math.log(round((len(words_count)/occurrences),4))
71      inverse_document_frequency.append(invFre)
```

（圖十一）

（十）詞頻 X 逆文件頻率

最後，得到的詞頻和逆文件頻率相乘，得到的 all_tf_idf，就是三位歌手的 tf-idf 的值了！

```
73  ##tf*idf
74  all_tf_idf = []
75  for i,words in enumerate(words_frequency):
76      tf_idf = {}
77      for word,freq in words.items():
78          tf_idf[word] = freq*inverse_document_frequency[i][word]
79      all_tf_idf.append(tf_idf)
```

（圖十二）

（十一）WordCloud 套件視覺化

根據 all_tf_idf 中得到的值，透過 WordCloud 視覺化出來，各項視覺化的變數、間距、顏色以及圖片大小皆可以自行客製化。這邊的重點是字體一定要設定，否則會出現亂碼。

```
path = ["images/JayChou.png","images/JayJay.png","images/Mayday.png"]
for i,tf_idf in enumerate(all_tf_idf):
    WordCloud(collocations=False,
              font_path="C:\Windows\Fonts\msjhbd.ttc", # 字體設定(是中文一定要設定，否則會是亂碼)
              #font_path='NotoSansCJKjp-Black.otf', 字體設定(是中文一定要設定，否則會是亂碼)
              width=600, # 圖片寬度
              height=600, # 圖片高度
              background_color = "white" , #圖片底色
              margin=2 # 文字之間的間距
              ).generate_from_frequencies(tf_idf).to_image().save(path[i])
```

（圖十三）

肆、研究分析與結果

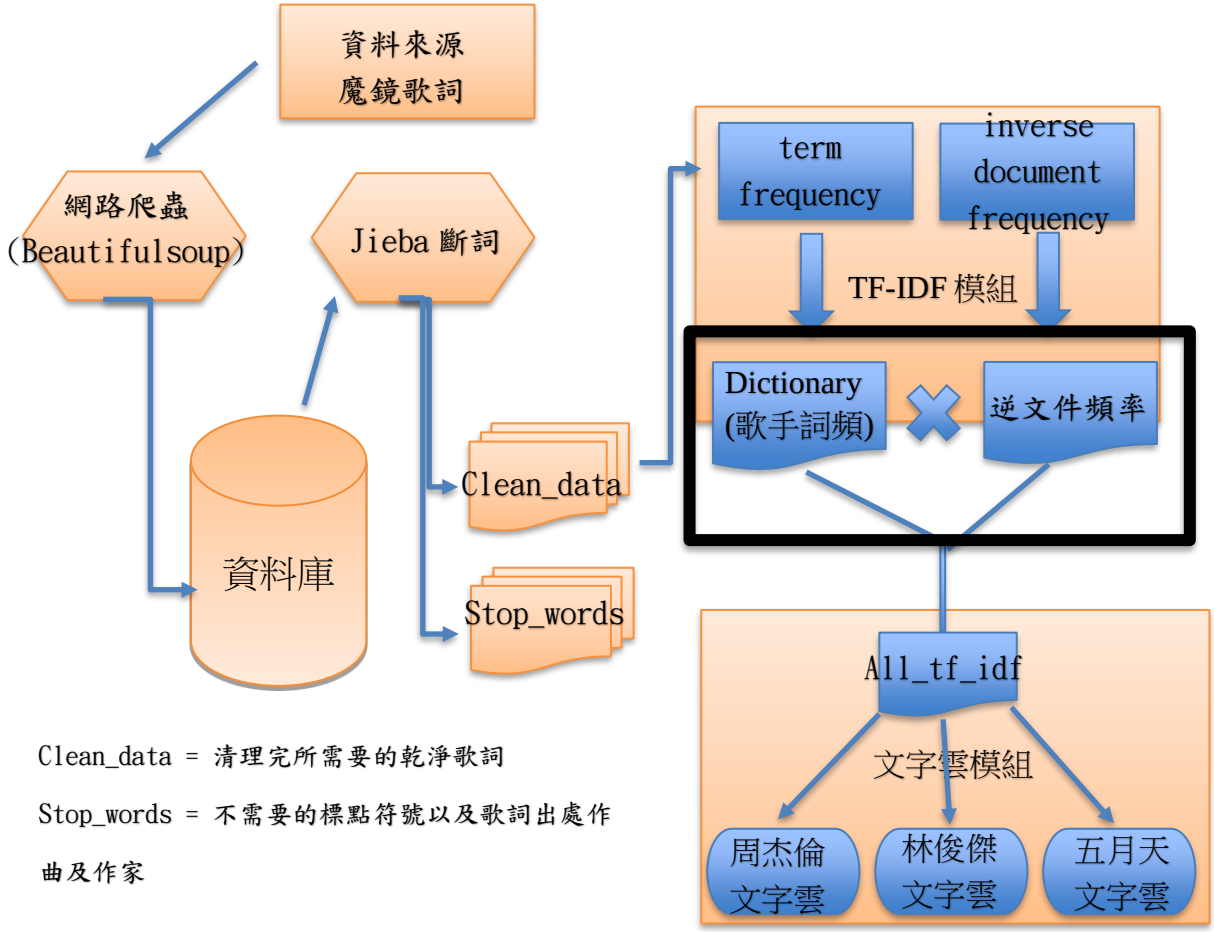
分別得出三張文字雲，我們可以從用詞理解到五月天對於用詞較為狂野，腦中也有浮現出一些較為熱血的曲風，林俊傑的曲風則是讓有有一種憂鬱且封閉的感覺，也透漏著自己的壓力，適合失戀的時候聽，則周杰倫的曲風較為偏向於古著，比較有趣且散發出自己的特色。

周杰倫　　林俊傑　　五月天

使用 open source 方式呈現:
https://github.com/chris911024/tf-idf-cloud-text

如果一首歌反覆出現同一個語詞，詞頻會很高，詞頻越高，占比就越大，也就是為什麼周杰倫、公公、凍結、刺痛、抓狂這些詞的占比這麼大的原因，一首歌的語詞數量相較於一篇文章短許多，即便除以總文本語詞數量仍無法客觀衡量，而另一個逆文件頻率也面臨到相同的問題，在文數量(歌手數量)不夠多的情況，逆文件頻率也無法精準衡量"獨特性"。

一、系統流程



Clean_data = 清理完所需要的乾淨歌詞

Stop_words = 不需要的標點符號以及歌詞出處作曲及作家

伍、研究結論與建議

可以將此技術應用到各個搜索引擎，在這個數據量龐大的世代裡，可以使用數據分析以及視覺化的量化方式，明白到消費者的喜好項目，以及套用在各個領域皆可以作為商用應用，讓設計者以及調查機構能夠快速掌握出民眾的喜愛以及最常搜尋的字眼跟主題，用來抓住大部分民眾的心，我喜歡聽歌，所以產生出”這些歌手分別以那些歌詞出現的次數最多?”以及”他們的特色是甚麼?”的問題，使用網路爬蟲先取得龐大的資料，jieba 先將取得到的資料進行分詞，再使用了 TF-IDF 演算法做出量化，計算出歌詞出現的頻率，一系列的探勘最後以視覺化的方式由數量大到小進行排列構圖，明白這些歌手使用的歌詞頻分別那些最多，因為歌詞本文較短，語詞次數換增大，想到的辦法是加入特定限制，例如一首歌的詞與次數有上限，畢竟我探索的是一個歌手”整體”會在歌詞中運用到甚麼詞彙，多少能夠反映出歌手的風格，如果只有一首歌有失偏頗。

陸、參考文獻

1. About David Huang 指出:
[文件探勘] TF-IDF 演算法：快速計算單字與文章的關聯
https://taweihuang.hpd.io/2017/03/01/tfidf/

2. Friedrich1942 指出:
[常見的自然語言處理技術] 重不重要？TF-IDF 會告訴你
https://ithelp.ithome.com.tw/articles/10267287

3. 楊德倫 指出:
文字探勘之前處理與 TF-IDF 介紹
https://www.cc.ntu.edu.tw/chinese/epaper/0031/20141220_3103.html

4. Clay 指出:
[NLP] 文字探勘中的 TF-IDF 技術
https://clay-atlas.com/blog/2020/08/01/nlp-%E6%96%87%E5%AD%97%E6%8E%A2%E5%8B%98%E4%B8%AD%E7%9A%84-tf-idf-%E6%8A%80%E8%A1%93/

5. 丹尼爾胡 指出:
TF-IDF 文件加權與實作
https://ithelp.ithome.com.tw/articles/10214726

6. havohej 指出:
如何使用 Python 製作文字雲
https://tech.havocfuture.tw/blog/python-wordcloud-jieba

7. Clay 指出:
[Python] 使用 wordcloud 套件快速產生文字雲
https://clay-atlas.com/blog/2019/11/25/python-chinese-tutorial-cloudword-demo/

8. 恩哥 python 指出:
【wordcloud】用 python 繪製文字雲：抓取 yahoo 新聞用 jieba+wordcloud 繪製自己的文字雲 看完文章 5 分鐘馬上會寫 code
https://pixnashpython.pixnet.net/blog/post/28128736-%E6%96%87%E5%AD%97%E9%9B%B2

9. Aaron Ho 指出:
Python 大數據分析(二)
https://hackmd.io/@aaronlife/python-bigdata-02

10. Jacky Lu 指出:
筆記 for Python (Jieba + Wordcloud)
https://medium.com/@fsflyingsoar/%E7%AD%86%E8%A8%98-for-python-jieba-wordcloud-b814f5e04e01

11. Stace's Blog 指出:
文字雲 word cloud in python
http://stacepsho.blogspot.com/2018/06/word-cloud-in-python.html

12. fukuball 指出:
如何使用 jieba 結巴中文分詞程式
https://coderwall.com/p/38wtgw/jieba

13. 行銷搬進大程式 指出:
Jieba 切詞基本概念
https://marketingliveincode.com/?p=2310

14. fukuball 指出:
JIEBA 結巴中文斷詞
https://speakerdeck.com/fukuball/jieba-jie-ba-zhong-wen-duan-ci

15. 中研院指出:
CKIP
https://ckip.iis.sinica.edu.tw/