Hw#1, NLP@CGU Spring 2023

LINK: paste your link here

https://colab.research.google.com/drive/14Rsq8nAHRPWEWRsOZHxYcEBGFbIlOCwP?usp=sharing

Student ID: B0928001 **Name**: 賴霆瑞

```
import sys
!{sys.executable} -m pip install jieba
import jieba
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: jieba in /usr/local/lib/python3.9/dist-packages (0.42.1)
```

```
# 讀取文本文件
with open('/content/drive/MyDrive/user_text.txt', 'r', encoding='utf-8') as f:
    content = f.read()
```

```
# 去除標點符號
import requests
import string
from zhon.hanzi import punctuation

content = content.replace(' ', '')
content = content.replace('\t', '')
for i in string.punctuation:
    content = content.replace(i, '')
for i in punctuation:
    content = content.replace(i, '')
spacial_punctuation = ['▂', '━', '┤', '←', '┌', 'へ', '—', '::', '_', '‿', '☁']
for i in spacial_punctuation:
    content = content.replace(i, '')
```

```
### 自定義辭典
jieba.load_userdict(r'/content/drive/MyDrive/dict.txt.big')
```

```
### 手動添加語料庫
jieba.add_word('為什麼')
```

```
# 切分文章（一行算一個文章）
All_articles = content.split('\n')
lines = len(All_articles)
print("文章數：", lines)
```

```
文章數： 418202
```

```
### 導入模組
import jieba
from collections import Counter
import pandas as pd
import jieba.analyse
import matplotlib.pyplot as plt
from wordcloud import WordCloud
import numpy as np
import math
import matplotlib as mpl
import matplotlib.pyplot as plt
from matplotlib.font_manager import fontManager
```

```
### 使用分詞工具斷詞，使用jieba
tk_articles = []
for articles in All_articles:
    tk_articles.append((jieba.lcut(articles), len(articles)))
```

```
### test tokenize result
print(tk_articles[:1])
```

```
[(['為什麼', '聖', '結石', '會', '被', '酸', '而', '這群人', '不會', '質感', '劇本', '成員', '都', '差', '很多', '好嗎', '不要', '拿', '腎結石',
```

```
### 統計前一百個高頻字詞
high_freq_List = []
Num = 0
for articles in tk_articles:
    counter = Counter(articles[0])
    for item in counter.items():
        high_freq_List.append((Num , item[0], item[1] / articles[1]))
        Num += 1
high_freq_List = sorted(high_freq_List, key=lambda item:item[2], reverse = True)
print(high_freq_List[:10])
```

```
[(2766962, '咩', 1.0), (5000790, '人', 1.0), (5748340, '人', 1.0), (1765316, '啦', 0.9), (2788465, '噢', 0.8636363636363636), (1042690, '喔', 0.
```

```
### 統計前一百個TF-IDF權重高的字詞
## 先計算IDF
IDF_List = {}
for articles in tk_articles:
    counter = Counter(articles[0])
    for item in counter.items():
        pre_idf = IDF_List.get(item[0])
        if pre_idf:
            IDF_List.update({item[0]: pre_idf + item[1]})
        else:
            IDF_List[item[0]] = item[1]
for IDF in IDF_List.items():
    IDF_List[IDF[0]] = math.log(lines / IDF[1], 10)

# 依照IDF權重排列
IDFs = sorted(IDF_List.items(), key=lambda item:item[1], reverse=True)
IDFs = IDFs
TPK100_IDF = {}
for lt in IDFs:
    TPK100_IDF[lt[0]] = lt[1]


### 計算TF-IDF
tf_idf_List = []
for item in high_freq_List:
    tf_idf_List.append((item[0], item[1], item[2] * IDF_List[item[1]]))
tf_idf_List = sorted(tf_idf_List, key=lambda item:item[2], reverse=True)
print(tf_idf_List[:10])
```

```
[(1428267, '鞚', 3.365130205980905), (2788465, '噢', 3.0349702990992062), (652983, '龘', 2.9055784426779128), (1069996, '歹', 2.868186350925134
```

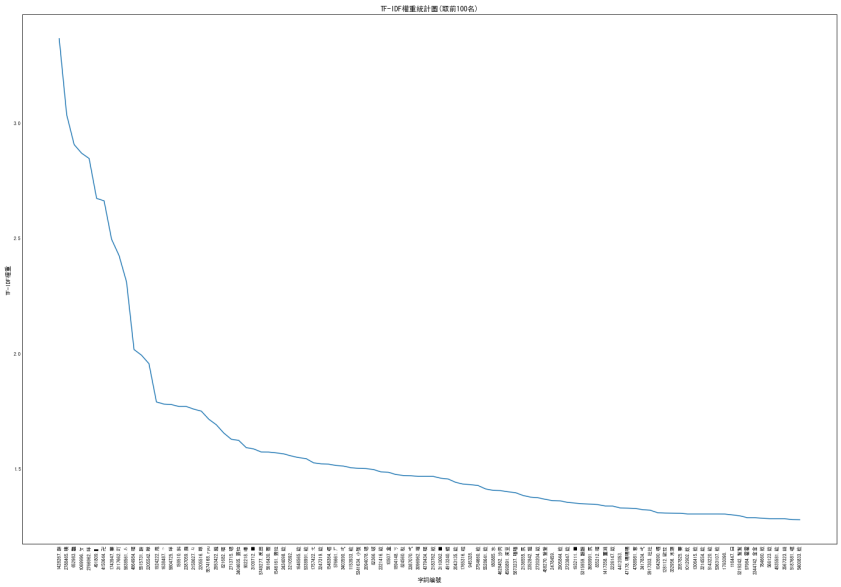
```
plt.rcParams['font.sans-serif'] = ['SimHei']
import warnings
warnings.filterwarnings("ignore", message="Glyph .* missing from current font.")



# TF-IDF權重統計圖
x_axis = []
y_axis = []

for item in tf_idf_List[:100]:
    x_axis.append(str(item[0]) + '.' + item[1])
    y_axis.append(item[2])

plt.figure(figsize = (24,16))
plt.plot(x_axis, y_axis)
plt.title("TF-IDF權重統計圖(取前100名)")
plt.xlabel("字詞編號")
plt.xticks(rotation = 90)
plt.ylabel("TF-IDF權重")
plt.show()
```

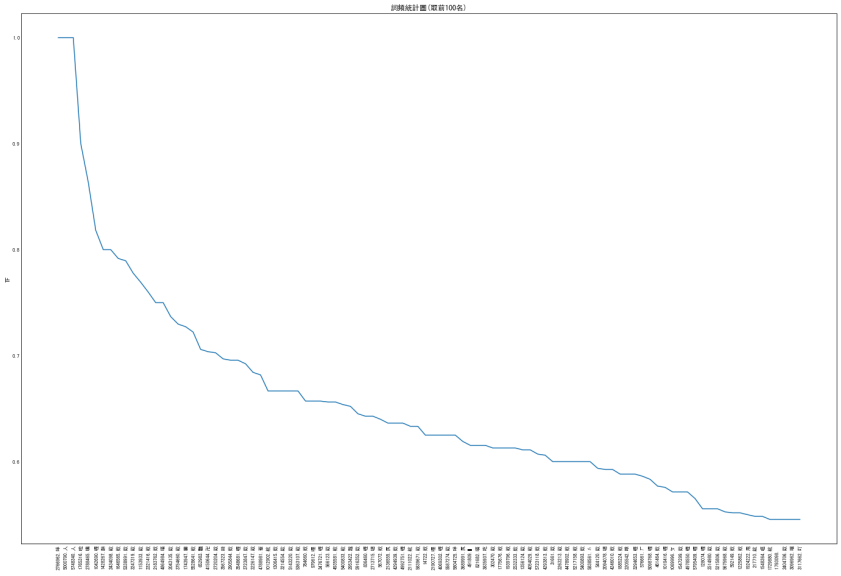



```
##  詞頻統計圖
x_axis  =  []
y_axis  =  []

for  item  in  high_freq_List[:100]:
        x_axis.append(str(item[0])  +  '.'  +  item[1])
        y_axis.append(item[2])

plt.figure(figsize  =  (24,16))
plt.plot(x_axis,  y_axis)
plt.title("詞頻統計圖(取前100名)")
plt.xlabel("字詞編號")
plt.xticks(rotation  =  90)
plt.ylabel("TF")
```

```
Text(0, 0.5, 'TF')
```



```
###  製作取前32個文字雲  (Frequency)  fig#3
from  wordcloud  import  WordCloud,  STOPWORDS
import  numpy  as  np
import  matplotlib.pyplot  as  plt
from  PIL  import  Image
import  jieba
import  jieba.analyse
from  collections  import  Counter  #  次數統計

dictfile  =  r"/content/drive/MyDrive/dict.txt.big"    #  字典檔
stopfile  =  r"/content/drive/MyDrive/stop_words.htm"    #  stopwords
fontpath  =  r"/content/drive/MyDrive/SimHei.ttf"    #  字型檔

jieba.set_dictionary(dictfile)
jieba.analyse.set_stop_words(stopfile)

X  =  1
freq  =  {}
for  l  in  high_freq_List[:32]:
        freq[str(X)  +  '.'  +  l[1]]  =  l[2]
        X  +=  1
print(freq)  #  計算出現的次數

wordcloud  =  WordCloud(background_color="white",  contour_width=3,  contour_color='steelblue',  font_path=  fontpath).generate_from_frequencies(freq)
plt.figure()
plt.imshow(wordcloud,  interpolation="bilinear")
plt.axis("off")
plt.show()
```

```
{'1.咩': 1.0, '2.人': 1.0, '3.人': 1.0, '4.啦': 0.9, '5.噢': 0.8636363636363636, '6
```


```
###  製作取前32個文字雲  (TF-IDF)  fig#3
from  wordcloud  import  WordCloud,  STOPWORDS
import  numpy  as  np
import  matplotlib.pyplot  as  plt
from  PIL  import  Image
import  jieba
import  jieba.analyse
from  collections  import  Counter  #  次數統計

dictfile  =  r"/content/drive/MyDrive/dict.txt.big"    #  字典檔
stopfile  =  r"/content/drive/MyDrive/stop_words.htm"    #  stopwords
fontpath  =  r"/content/drive/MyDrive/SimHei.ttf"    #  字型檔

jieba.set_dictionary(dictfile)
jieba.analyse.set_stop_words(stopfile)
```

```
X = 1
freq = {}
for l in tf_idf_List[:32]:
        freq[str(X) + '.' + l[1]] = l[2]
        X += 1
print(freq) # 計算出現的次數

wordcloud = WordCloud(background_color="white", contour_width=3, contour_color='steelblue', font_path= fontpath).generate_from_frequencies(freq)
plt.figure()
plt.imshow(wordcloud, interpolation="bilinear")
plt.axis("off")
plt.show()
```

```
{'1.韓': 3.365130205980905, '2.噢': 3.0349702990992062, '3.龘': 2.9055784426779128
```

Colab 付費產品 - 按這裡取消合約

✓ 0 秒 完成時間：晚上9:50