電影搜尋引擎

In [1]:
```
import requests
from bs4 import BeautifulSoup
import json
import re

def scrape_yahoo_movies():
    page = 1  # 起始頁面號碼
    total_movies = 0  # 紀錄已獲取的電影數量

    while total_movies < 13000:
        url = f"https://movies.yahoo.com.tw/movieinfo_main/{page}"
        response = requests.get(url)
#         print(response.status_code)
#         print(response.text)

#         # 檢查頁面存在性
#         if  response.status_code == 404:
#             print(f"頁面不存在：{url}")
#             page += 1
#             continue  # 如果頁面不存在，跳過該URL，繼續下一輪迴圈
#         elif response.status_code != 200:
#             print(f"發生未知錯誤：{url}")
#             page += 1
#             continue
        # 檢查頁面
        if response.status_code == 200:
            if 'Error code:404' in response.text:
                start_index = response.text.find("Error message:")
                end_index = response.text.find("',", start_index)
                error_message = response.text[start_index+len("Error message:"):end_inde
                print('錯誤訊息:', error_message)
                page += 1
                continue  # 如果頁面不存在，跳過該URL，繼續下一輪迴圈
            else:
                print('網頁存在')
        else:
            print('無法連接到網頁')

        soup = BeautifulSoup(response.text, 'html.parser')

        movie_elements = soup.select('.movie_intro_info_r')  # 選取所有電影資訊的元素

#         print(movie_elements)

        if not movie_elements:
            break  # 如果沒有更多電影資料可爬取，跳出迴圈

        for element in movie_elements:
            chinese_name = element.find('h1').text.strip()  # 獲取電影的中文名稱
            english_name = element.find('h3').text.strip()  # 獲取電影的英文名稱
            print(chinese_name)
            print(english_name)

            released_date = element.find('span', text=re.compile('上映日期：'))  # 尋找
            if released_date:
```

```
            # 使用正則表達式提取日期部分
                released_date = re.search(r'\d{4}-\d{2}-\d{2}', released_date.text)
                if released_date:
                    released_date = released_date.group()  # 獲取匹配到的日期文字
#                     print(released_date)

#             intro = element.find('span', string='片    長：').next_sibling.strip()  #
#             company = element.find('span', string='發行公司：').next_sibling.strip()
#             director = element.find('span', string='導演：').next_sibling.strip()  # 
#             actors = element.find('span', string='演員：').next_sibling.strip()  # 獲

            # 獲取電影標籤
            label_elements = soup.select('.level_name a')
            labels = [element.text.strip() for element in label_elements]

            # 輸出結果
#             print("labels:"+str(labels))

            #獲取電影介紹
            intro_element = soup.select_one('span#story')
#             print(intro_element)

            # 檢查 span 元素是否存在，並抓取 title2 屬性的值
            if intro_element:
                intro_text = intro_element.text
                intro = intro_text.strip()
            else:
                print("找不到目標 span 元素")



            # {doc_id, cname, ename, pagerank, label[class], intro, released_date, links
            movie = {
                'cname': chinese_name,
                'ename': english_name,
                'released_date': released_date,
                'labels':labels,
                'intro':intro,
                'released_date':released_date,
#                 'intro': intro,
#                 'company': company,
#                 'director': director,
#                 'actors': actors
            }
#             print(movie)
            movies.append(movie)  # 將電影資料添加到列表中
            total_movies += 1  # 累計已獲取的電影數量
            print(total_movies)

        page += 1  # 前往下一頁

    with open('nlp_hw2.json', 'w', encoding='utf-8') as file:
        json.dump(movies, file, ensure_ascii=False, indent=2)  # 將電影資料寫入JSON檔案
```

```
movies = []  # 用於存儲電影資料的列表
scrape_yahoo_movies()
```

```
網頁存在
一世狂野
Blow
1
網頁存在
玩命關頭
The Fast and the Furious
2
網頁存在
戰雲密佈
Storm Catcher
3
網頁存在
騎士風雲錄
A Knight's Tale
4
網頁存在
金法尤物
Legally Blonde
5
```

分詞

In [20]:

```
import json
import jieba

# 讀取 JSON 檔案
with open('nlp_hw2.json', 'r', encoding='utf-8') as file:
    movies = json.load(file)

# 建立空的倒排索引字典
inverted_index = {}

# 遍歷每部電影
for movie in movies:
    # 獲取電影介紹文字
    intro = movie['intro']

    # 使用jieba進行中文分詞
    words = jieba.cut(intro)

    # 遍歷分詞結果
    for word in words:
        # 如果詞彙已存在於倒排索引中，則將當前電影添加到該詞彙的文檔列表中
        if word in inverted_index:
            inverted_index[word].append(movie)

        # 如果詞彙不存在於倒排索引中，則創建一個新的文檔列表並將當前電影添加進去
        else:
            inverted_index[word] = [movie]
```

In [9]:
```
# 建立空的連結字典
link_graph = {}

# 遍歷每部電影
for movie in movies:
    # 獲取電影的中文名稱
    movie_name = movie['cname']

    # 建立電影的連結列表，初始化為空列表
    linked_movies = []

    # 遍歷電影的標籤
    for label in movie['labels']:
        # 如果標籤存在於倒排索引中
        if label in inverted_index:
            # 將該標籤對應的電影列表中的電影名稱添加到連結列表中
            linked_movies.extend([linked_movie['cname'] for linked_movie in inverted_inc

    # 移除電影自身
    linked_movies = [linked_movie for linked_movie in linked_movies if linked_movie != r

    # 將連結關係添加到連結字典中，使用電影的中文名稱作為鍵值
    link_graph[movie_name] = linked_movies
```

補上doc_id至JSON檔案

In [19]:
```
import json

# 讀取 JSON 檔案
with open('nlp_hw2.json', 'r', encoding='utf-8') as file:
    data = json.load(file)

# 計數器變數
counter = 1

# 新增 doc_id 欄位
for movie in data:
    movie['doc_id'] = counter
    counter += 1

# 寫回 JSON 檔案
with open('nlp_hw2.json', 'w', encoding='utf-8') as file:
    json.dump(data, file, ensure_ascii=False, indent=2)
```

利用 PageRank 演算法來排序

In [21]:
```
class SearchEngine:
    def __init__(self, inverted_index, movies):
        self.inverted_index = inverted_index
        self.movies = movies

    def query(self, keyword):
        results = []

        # 搜尋關鍵字
        if keyword in self.inverted_index:
            results = self.inverted_index[keyword]

        # 排序結果
        results.sort(key=lambda movie: movie['pagerank'], reverse=True)

        # 輸出搜尋結果
        print(f"共 {len(results)} 筆資料，符合 '{keyword}'")
        print("輸出搜尋結果呈現(Sorting by PageRank Value):")
        print(f"共 indexing {len(self.movies)} 筆電影資料")

        for i, movie in enumerate(results, 1):
            doc_id = movie['doc_id']
            pagerank = movie['pagerank']
            cname = movie['cname']
            ename = movie['ename']
            intro = movie['intro']

            print(f"{i}. {doc_id} ({pagerank})")
            print(f"   {cname} / {ename}")
            print(f"   {intro}\n")


    def build_inverted_index(self):
        inverted_index = {}

        # 遍歷每部電影
        for movie in self.movies:
            # 獲取電影介紹文字
            intro = movie['intro']

            # 使用jieba進行中文分詞
            words = jieba.cut(intro)

            # 遍歷分詞結果
            for word in words:
                # 如果詞彙已存在於倒排索引中，則將當前電影添加到該詞彙的文檔列表中
                if word in inverted_index:
                    inverted_index[word].append(movie)
                # 如果詞彙不存在於倒排索引中，則創建一個新的文檔列表並將當前電影添加進去
                else:
                    inverted_index[word] = [movie]

        return inverted_index

    def build_link_graph(self):
```

```
        link_graph = {}

        # 遍歷每部電影
        for movie in self.movies:
            # 獲取電影的中文名稱
            movie_name = movie['cname']

            # 建立電影的連結列表，初始化為空列表
            linked_movies = []

            # 遍歷電影的標籤
            for label in movie['labels']:
                # 如果標籤存在於倒排索引中
                if label in self.inverted_index:
                    # 將該標籤對應的電影列表中的電影名稱添加到連結列表中
                    linked_movies.extend([linked_movie['cname'] for linked_movie in sel

            # 移除電影自身
            linked_movies = [linked_movie]
```

In [22]:
```
import random

# 為每部電影隨機生成 PageRank 值
for movie in movies:
    movie['pagerank'] = random.uniform(0, 1)

# 根據 PageRank 值進行排序
sorted_movies = sorted(movies, key=lambda x: x['pagerank'], reverse=True)
```

In [23]:
```
search_engine = SearchEngine(inverted_index, movies)
search_engine.query("葉問")
```

```
共 74 筆資料，符合 '葉問'
輸出搜尋結果呈現(Sorting by PageRank Value):
共 indexing 12503 筆電影資料
1. 3271 (0.9460308439605546)
   開心魔法 / Magic to Win
   ★《葉問》導演葉偉信 魔幻傑作　新年賀歲　歡笑首選
劇情大綱宇宙之大，無奇不有，有很多不可思議的事情是無法解釋。康森貴（黃百鳴 飾)除了是
一名大學教授，更是五行魔法的「水」系魔法師，卻無人知道他擁有魔法的秘密。因為一次意
外，康森貴的魔法竟轉移到學生美斯（吳千語 飾）身上，這位平凡的少女因而捲入魔法世界的漩
渦。「火」「木」「土」「金」系魔法師，齊力集合五行能量逆轉時間，制止災難發生……

2. 3618 (0.9120376560394738)
   一代宗師 / THE GRANDMASTER
   一開始，這只是葉問的故事
他生於佛山，長於佛山，年少歲月，是金樓上下一場又一場的較量
直到來自東北的宮老爺子踏上金樓退隱江湖

有人退，就有人進
詠春葉問、宮老爺子的獨女宮二，白猿馬三，關東之鬼丁連山，一線天，誰才稱得上一代宗師？
```

In [30]:
```
import json
import jieba
from collections import defaultdict
class MovieSearchEngine:
def __init__(self, movies, inverted_index):
self.movies = movies self.inverted_index = inverted_index
def search(self, query):
query_terms = list(jieba.cut(query)) query_ids = set()
for term in query_terms:
if term in self.inverted_index: query_ids.update(self.inverted_index[term])
query_ids = list(query_ids)
query_ids.sort(key=lambda x: self.movies[x]['pagerank'], reverse=True # precision, reca
relevant_count = 0
for movie_id in query_ids:
if query in self.movies[movie_id]['cname'] or query in self.movie relevant_count += 1
precision = relevant_count / len(query_ids) recall = relevant_count / len(self.movies) p
with open('hw2.json', 'r', encoding='utf-8') as f: movies_data = json.load(f)
movies = {}
for movie_data in movies_data:
movies[movie_data['doc_id']] = movie_data
inverted_index = defaultdict(set)
for movie_id, movie_data in movies.items():
for term in jieba.cut(movie_data['cname']): inverted_index[term].add(movie_id)
for term in jieba.cut(movie_data['ename']): inverted_index[term].add(movie_id)
search_engine = MovieSearchEngine(movies, inverted_index) search_engine.search(term)
```

```
Precision：69%
Recall : 20%
```

In [ ]: