學號:B0928031

姓名:鄭茹云

學系:人工智慧三甲

連結: https://colab.research.google.com/drive/1CsqyxejWMgr62E_c8nE1i1PRYCJPdj8x?usp=share_link

```
import requests
from bs4 import BeautifulSoup
import networkx as nx
import json

base_url = "https://movies.yahoo.com.tw/movieinfo_main/"
start_page = 1
end_page = 500

G = nx.DiGraph()

for page in range(start_page, end_page+1):
        url = base_url + str(page)
        response = requests.get(url)
        soup = BeautifulSoup(response.content, "html.parser")
        if not soup.find("div", class_="movie_intro_info"):
                continue
        # add the URL
        G.add_node(url)

        doc_id = page
        cname = soup.find("div", class_="movie_intro_info").find("h1").text.strip()
        ename = soup.find("div", class_="movie_intro_info_r").find("h3").text.strip()
        label = soup.find("div", class_="level_name").text.strip()
        intro = soup.find("span", id="story").text.strip().replace('\n\n', '')
        released_date = soup.find("div", class_="movie_intro_info_r").find_all("span")[0].text.strip().repla

        links = []
        for link in soup.find_all("a"):
                href = link.get("href")
                text = link.text.strip()
                if href and text:
                        links.append(href)
                        G.add_edge(url, href)

        # calculate
        pagerank = nx.pagerank(G)

        # create a dictionary
        movie_info = {
                "doc_id": doc_id,
                "cname": cname,
                "ename": ename,
                "pagerank": pagerank[url], # use the URL as the key to get the PageRank
                "label": label,
                "intro": intro,
                "released_date": released_date,
                "links": links # store only the hrefs of the links
        }

        #JSON file
        with open("movie.json", "a", encoding="utf-8") as f:
```

```
                json.dump(movie_info, f, ensure_ascii=False)
                f.write("\n") # add a new line at the end of each JSON object to separate them


import json
import jieba
from collections import defaultdict

# load data
with open('movie.json', 'r', encoding='utf-8') as f:
    movies = [json.loads(line) for line in f]


def tokenize(text):
    return list(jieba.cut(text))

inverted_index = defaultdict(list)
for movie in movies:
    tokens = tokenize(movie['intro'] + movie['cname'] + movie['ename'] + movie['label'])
    for token in tokens:
        inverted_index[token].append(movie['doc_id'])

# save to a JSON file
with open('invert.json', 'w', encoding='utf-8') as f:
    json.dump(inverted_index, f, ensure_ascii=False)
    f.write("\n") # add a new line at the end of each JSON object to separate them
```

```
Building prefix dict from the default dictionary ...
DEBUG:jieba:Building prefix dict from the default dictionary ...
Dumping model to file cache /tmp/jieba.cache
DEBUG:jieba:Dumping model to file cache /tmp/jieba.cache
Loading model cost 1.074 seconds.
DEBUG:jieba:Loading model cost 1.074 seconds.
Prefix dict has been built successfully.
DEBUG:jieba:Prefix dict has been built successfully.
```

```
import json

movies = []
with open("movie.json", "r", encoding="utf-8") as f:
    for line in f:
        movie_info = json.loads(line)
        movies.append(movie_info)

# sort PageRank scores
movies_sorted = sorted(movies, key=lambda x: x["pagerank"], reverse=True)

for i, movie in enumerate(movies_sorted[:]):
    print(f"{i+1}. {movie['cname']} ({movie['released_date']}) - PageRank score: {movie['pagerank']:.
```

```
345. 永遠的莉莉亞 (2003-07-12) - PageRank score: 0.0003
346. 特務辦囍事 (2003-08-08) - PageRank score: 0.0003
347. 單刀直入 (2003-09-05) - PageRank score: 0.0003
348. 鬼魅 (2003-08-18) - PageRank score: 0.0003
349. 愛情趴趴走 (2003-09-10) - PageRank score: 0.0003
350. 千年女優 (2003-08-01) - PageRank score: 0.0003
351. 28天毀滅倒數 (2003-10-24) - PageRank score: 0.0003
352. 奶爸安親班 (2003-08-22) - PageRank score: 0.0003
353. 海底總動員 (2003-08-08) - PageRank score: 0.0003
354. 反恐特警組 (2003-08-29) - PageRank score: 0.0003
355. 甜蜜的永遠 (2003-08-29) - PageRank score: 0.0003
356. 紐約黑幫 (2003-08-22) - PageRank score: 0.0003
357. 古墓奇兵：風起雲湧 (2003-08-15) - PageRank score: 0.0003
358. 我很想你 (2003-08-08) - PageRank score: 0.0003
359. 槓上富家女 (2003-11-14) - PageRank score: 0.0003
360. 決戰異世界 (2003-09-26) - PageRank score: 0.0003
361. 美麗·壞東西 (2003-09-05) - PageRank score: 0.0003
362. 咒怨 (2003-08-22) - PageRank score: 0.0003
363. Oh! Happy Day (2003-08-29) - PageRank score: 0.0003
364. 好萊塢重案組 (2003-09-10) - PageRank score: 0.0003
365. 雙雄 (2003-09-10) - PageRank score: 0.0003
366. 向左走向右走 (2003-08-30) - PageRank score: 0.0003
367. 老師你好 (2003-09-19) - PageRank score: 0.0003
368. 偷天換日 (2003-09-19) - PageRank score: 0.0003
369. 兩個頭一個大 (2003-09-19) - PageRank score: 0.0003
370. 天地英雄 (2003-10-03) - PageRank score: 0.0003
371. 天降奇兵 (2003-10-09) - PageRank score: 0.0003
372. 怒海爭鋒：極地征伐 (2003-11-28) - PageRank score: 0.0003
373. 奔騰年代 (2003-09-26) - PageRank score: 0.0003
374. 辣媽辣妹 (2003-10-09) - PageRank score: 0.0003
375. 美國派之昏禮 (2003-10-03) - PageRank score: 0.0003
376. 讓愛飛翔 (2003-10-09) - PageRank score: 0.0003
377. 火柴人 (2003-10-09) - PageRank score: 0.0003
378. 記得我愛你 (2003-10-09) - PageRank score: 0.0003
379. 浴血叢林 (2003-10-17) - PageRank score: 0.0003
380. 蒙娜麗莎的微笑 (2003-12-31) - PageRank score: 0.0003
381. 黑道千金逼我嫁 (2004-02-04) - PageRank score: 0.0003
382. 神經殺手 (2003-10-13) - PageRank score: 0.0003
383. 四百擊 (2003-10-03) - PageRank score: 0.0003
384. 夏日之戀 (2003-10-03) - PageRank score: 0.0003
385. 追殺比爾 (2003-10-24) - PageRank score: 0.0003
386. 真情假愛 (2003-10-17) - PageRank score: 0.0003
387. 一不住二不休 (2003-11-21) - PageRank score: 0.0003
388. 毛骨悚然：鬼擋路 (2003-11-21) - PageRank score: 0.0003
389. 咒怨 2 (2003-11-28) - PageRank score: 0.0003
390. 誰領風騷 (2003-11-22) - PageRank score: 0.0003
391. 無間道 II (2003-11-14) - PageRank score: 0.0003
```

輸出搜尋結果呈現

```
import json
import re

def search_movies(query):
    with open("movie.json", "r", encoding="utf-8") as f:
        movies = [json.loads(line) for line in f]

    pageranks = {movie["doc_id"]: movie["pagerank"] for movie in movies}

    matches = [movie for movie in movies if any([query.lower() in str(value).lower() for key,

    sorted_matches = sorted(matches, key=lambda x: pageranks.get(x["doc_id"], 0), reverse=True)

    relevant_docs = set([movie['doc_id'] for movie in movies if query.lower() in str(movie).lower
    retrieved_docs = set([movie['doc_id'] for movie in matches])
    true_positives = relevant_docs.intersection(retrieved_docs)
```

```
        precision = len(true_positives) / len(matches) if len(matches) > 0 else 0
        recall = len(true_positives) / len(relevant_docs) if len(relevant_docs) > 0 else 0

        # print the search results and evaluation metrics
        print("您的搜尋結果 (Sorting by PageRank Value)：")
        print(f"共 {len(sorted_matches)} 筆，符合'{query}' - - - 共 indexing {len(movies)} 筆電影資料
        for movie in sorted_matches:
            intro = movie.get("intro", "")
            print(f"{movie['doc_id']} ({pageranks.get(movie['doc_id'], 0)}): {movie['cname']} ({movie['

        print(f"\nPrecision: {precision:.0%}")
        print(f"Recall: {recall:.0%}")



search_movies("黑暗")
```

```
您的搜尋結果 (Sorting by PageRank Value)：
共 11 筆，符合'黑暗' - - - 共 indexing 436 筆電影資料
48 (0.0018229912567740 41): 魔戒首部曲 (The Lord of the Rings:The Fellowship of the Ring) - 本片是英國鬼才
94 (0.0010948153980800 04): 鬼地方. (The Hole) - 一名高中女生步履蹣跚、跌跌撞撞地走過學校的長廊，她全身傷
  沒想到這個方法果然奏效，隔天早上，地洞的門打開了，四個受困的年輕人重獲自由，此時麥克緊緊擁著麗莎，動作
  於是馬汀也將整個事件的來龍去脈陳述一遍，詎料這個版本竟於麗莎的截然不同，這個恐怖的說法裡是一份意亂情迷
136 (0.0007811939265487015): 記憶拼圖 (Memento) - 黑暗中突然出現了一道閃光，一名男子被開槍射中頭部，兇手
233 (0.0005258282413363092): 魔咒女王 (Qween Of The Damned) - 傳說中的吸血鬼雷斯塔（史都華唐森飾）在墳墓
336 (0.0003807485855940173): 魔戒二部曲：雙城奇謀. (The Lord of Ring: The Two Towers) - 在【魔戒首部曲】
362 (0.0003479865866980227): 悄悄告訴她 (Talk to Her) - 紅色的幕簾升起，出現在舞台上的是舞蹈家皮娜包許（P
393 (0.0003210480111590462): 鬼潛艇 (Below) - 二次大戰期間，美軍潛艦「USS虎鯊號」奉命達成任務，回程途中船
396 (0.0003177275852999 9946): 綠巨人浩克 (The Hulk) - 假如有一個人一直在你身邊保護你，每當你被憤世嫉俗的
407 (0.0003072471219755 1716): 千機變 (The Twins Effect) - 常周遊列國，光明獵人（vampire slayer），是一種
411 (0.0003039408850282345): 鬼魅 (A Tale of Two Sisters) - 一對如花似玉的富家姐妹秀薇和秀蓮，在父親結識
425 (0.0002913733378427 23): 咒怨 (JU-ON: The Grudge) - 從事老人看護義工的一名女大學生，在訪問郊區的一棟房

Precision: 100%
Recall: 100%
```