In [7]:

```
import requests
from bs4 import BeautifulSoup
import json
import jieba

url = 'https://movies.yahoo.com.tw/category.html'
response = requests.get(url)
html = response.text

soup = BeautifulSoup(html, 'html.parser')
links = soup.select('.video_category_list.category-list ._slickcontent a')
url_queue = []
for link in links:
    url_queue.append(link['href'])

def crawl_yahoo_movies(i):
    current_url = url_queue.pop(0)
    movie_id = current_url.split('/')[-1]
    url = f'https://movies.yahoo.com.tw/movieinfo_main/{movie_id}'
    response = requests.get(url)
    soup = BeautifulSoup(response.text, 'html.parser')
    movie_list = []
    #for movie_elem in soup.select('.release_list .release_info'):
    movie = {}
    url_parts = url.split("-")
    id_part = url_parts[-1].split("/")
    movie['doc_id'] = id_part[-1]  # doc_id
    movie['cname'] = soup.select_one('.movie_intro_info_r h1').text.strip()
    movie['ename'] = soup.select_one('.movie_intro_info_r h3').text.strip() #
    movie['pagerank'] = "" # PageRank
    movie['label'] = soup.select_one('.level_name').text.strip() # label
    movie['intro'] = soup.select_one('.gray_infobox_inner span').text.strip()
    movie['released_date'] = soup.find("div", class_="movie_intro_info_r").fi
    movie['links'] = url
    movie_list.append(movie)
    url_queue.append(current_url)
    return movie_list

all_movies = []
for i in range(1, 10001):
    all_movies.extend(crawl_yahoo_movies(i))
    if len(all_movies) >= 10000:
        break

# 建立Inverted Index
inverted_index = {}
for movie in all_movies:
    doc_id = movie['doc_id']
    cname = movie['cname']
    ename = movie['ename']
    pagerank = movie['pagerank']
    label = movie['label']
    intro = movie['intro']
    released_date = movie['released_date']
    links = movie['links']
    cname_words = jieba.lcut(cname) # 中文分詞
    intro_words = jieba.lcut(intro)
    for word in cname_words + intro_words:
        if word not in inverted_index:
            inverted_index[word] = []
        inverted_index[word].append(doc_id)
```

```
Building prefix dict from the default dictionary ...
Loading model from cache /var/folders/ry/phxc250s3lx_m2h646xkjb1h0000gn/T/jieb
a.cache
Loading model cost 0.700 seconds.
Prefix dict has been built successfully.
```

In [8]:

```
import networkx as nx

G = nx.DiGraph()
for movie in all_movies:
    G.add_node(movie['doc_id'])
    for term in movie['cname'].split() + movie['ename'].split():
        if term in inverted_index:
            for doc_id in inverted_index[term]:
                if doc_id != movie['doc_id']:
                    G.add_edge(doc_id, movie['doc_id'])

pagerank_values = nx.pagerank(G) # PageRank
for movie in all_movies:
    movie['pagerank'] = round(pagerank_values[movie['doc_id']], 5) # 取到小數第

all_movies.sort(key=lambda x: x['pagerank'], reverse=True)# 按照 PageRank 值從
```

```
/Users/hungshihching/opt/anaconda3/lib/python3.9/site-packages/scipy/__init__.
py:146: UserWarning: A NumPy version >=1.16.5 and <1.23.0 is required for this
version of SciPy (detected version 1.24.2
  warnings.warn(f"A NumPy version >={np_minversion} and <{np_maxversion}"
```

In [9]:

```
# 存JSON
with open('hw2.json', 'w', encoding='utf-8') as f:
    json.dump(all_movies, f, ensure_ascii=False, indent=4)

with open('inverted_index.json', 'w', encoding='utf-8') as f:
    json.dump(inverted_index, f)
```

In [16]:

```
import json

with open('hw2.json', 'r', encoding='utf-8') as f:
    all_movies = json.load(f)

term = input("請輸入搜尋關鍵字: ")
movies = [] # 放符合關鍵字的電影資料
matched_movies = 0 # 符合關鍵字數量
# search
for movie in all_movies:
    if term in movie['cname'] and term in movie['intro']: # cname & intro
        movies.append(movie)
        print("您的搜尋結果 (Sorting by PageRank Value)：共 ", len(movies), " 筆
        print("{} ({})中文片名: {}".format(movie['doc_id'], movie['pagerank'],
        print("{} ({})劇情介紹: {}".format(movie['doc_id'], movie['pagerank'],
        print("===" * 10)
        matched_movies += 1
    elif term in movie['cname']: # cname
        movies.append(movie)
        print("您的搜尋結果 (Sorting by PageRank Value)：共 ", len(movies), " 筆
        print("{} ({})中文片名: {}".format(movie['doc_id'], movie['pagerank'],
        print("===" * 10)
        matched_movies += 1
    elif term in movie['intro']: #intro
        movies.append(movie)
        print("您的搜尋結果 (Sorting by PageRank Value)：共 ", len(movies), " 筆
```

```
        print("{} ({})劇情介紹: {}".format(movie['doc_id'], movie['pagerank'],
        print("===" * 10)
        matched_movies += 1
```

```
請輸入搜尋關鍵字: 最美麗
您的搜尋結果 (Sorting by PageRank Value)：共 2 筆，符合” 最美麗 “  - - - 共 indexin
g 10000 筆電影資料
14979 (0.08333)中文片名: 最美麗的小事
                                                        (2023)
14979 (0.08333)劇情介紹: 《最美麗的小事》根據雪兒史翠德的暢銷小說集改編，女主角婚姻即將告
終。女兒幾乎不和她說話。曾經一片光明的寫作生涯無疾而終。所以當一個朋友建議她接手撰寫諮商專欄
時，她認為自己根本沒資格擔任這份工作…事實上，她也許是最有資格的人。
==============================
```

In [18]:

```
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
        # 計算 precision, recall
        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)
        print("Precision: {:.2%}".format(precision))
        print("Recall: {:.2%}".format(recall))

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: 70.00%
Recall: 20.00%
```

In [ ]: