Buổi 3: Gợi ý dựa trên nội dung

Mục tiêu:

- Củng cố lý thuyết về phương pháp gợi ý dựa trên nội dung
  - Dữ liệu văn bản: Sử dụng chỉ số tf-idf
  - Vector đặc trưng: Sử dụng đánh giá của người dùng

A. Hướng dẫn thực hành

1. Dữ liệu dạng text - Sử dụng chỉ số tf-idf

a) Cho biến dữ liệu chứa 4 items có thông tin như sau:

```
documents = (
"The sky is blue",
"The sun is bright",
"The sun in the sky is bright",
"We can see the shining sun, the bright sun"
)
```

b) Chuyển đổi dữ liệu về dạng vector các từ khoá

```
from sklearn.feature_extraction.text import TfidfVectorizer
tfidf_vectorizer = TfidfVectorizer()
tfidf_matrix = tfidf_vectorizer.fit_transform(documents)
tfidf_vectorizer.get_feature_names() # liệt kê các từ được sử dụng làm vector đặc trưng
tfidf_matrix.toarray()  # hiển thị ma trận tài liệu – các từ khoá là vector đặc trưng
print tfidf_matrix.shape
```

```
[u'blue', u'bright', u'can', u'in', u'is', u'see', u'shining', u'sky', u'sun', u'the', u'we']
>>> print tfidf_matrix.shape
(4, 11)
```

c) Ví dụ người dùng đã chọn tài liệu số 2: " the sun is bright", cần tìm độ tương tự của tài liệu số 2 và các tài liệu còn lại. => Tính giá trị cosin dựa vào vector tfidf của các tài liệu để xác định tài liệu gần nhất.

```
>>> from sklearn.metrics.pairwise import cosine_similarity
>>> cosine_similarity(tfidf_matrix[1:2], tfidf_matrix)
array([[0.36651513, 1.        , 0.72875508, 0.54139736]])
```

Với kết quả trên ta thấy tài liệu số 3 là tài liệu có thể sử dụng để gợi ý cho người dùng này với giá trị cos = 0.73 lớn hơn hẳn so tài liệu 1 và 4.

2. Dữ liệu đặc trưng của item là thể loại phim

a) Import các thư viện cần thiết và đọc dữ liệu phim (movies)

```
import pandas as pd
import numpy as np
import math
ratings = pd.read_csv("ratings.csv")
movies = pd.read_csv("movies.csv")
```

```
>>> movies.head()
   movieId                               title                                        genres
0        1                    Toy Story (1995)  Adventure|Animation|Children|Comedy|Fantasy
1        2                      Jumanji (1995)                 Adventure|Children|Fantasy
2        3             Grumpier Old Men (1995)                              Comedy|Romance
3        4            Waiting to Exhale (1995)                    Comedy|Drama|Romance
4        5  Father of the Bride Part II (1995)                                  Comedy
>>>
```

b) Đặc trưng (thuộc tính) của mỗi item dựa theo thể loại

Lấy thông tin "genre" của từng phim để chuyển đổi về dạng vector các giá trị 0,1

```
>>> movies.head()
   movieId                               title                                        genres
0        1                    Toy Story (1995)  Adventure|Animation|Children|Comedy|Fantasy
1        2                      Jumanji (1995)                 Adventure|Children|Fantasy
2        3             Grumpier Old Men (1995)                              Comedy|Romance
3        4            Waiting to Exhale (1995)                    Comedy|Drama|Romance
4        5  Father of the Bride Part II (1995)                                  Comedy
>>>
```

c) Lọc dữ liệu từ file "movies" để lấy thông tin thể loại

```
m = len(movies)
tt = movies.genres
genre_list= []
item_prof = [[]]
for i in range(1,5):
    aa = tt.ix[i].split('|')
    genre_list = genre_list + aa
    item_prof.append(aa)
```

```
>>> item_prof
[[], ['Adventure', 'Children', 'Fantasy'], ['Comedy', 'Romance'],
 ['Comedy', 'Drama', 'Romance'], ['Comedy']]
```

d) Chuyển về định dạng kiểu vector (movie, các thể loại)

```
from mlxtend.preprocessing import TransactionEncoder
te = TransactionEncoder()
te_ary = te.fit(item_prof).transform(item_prof)
te_ary = te_ary.astype("int")
df = pd.DataFrame(te_ary, columns=te.columns_)
df.head()
movieID = movies.movieId[0:5]
df.index = movieID
df.head()
```

```
         Adventure  Children  Comedy  Drama  Fantasy  Romance
movieId
1                0         0       0      0        0        0
2                1         1       0      0        1        0
3                0         0       1      0        0        1
4                0         0       1      1        0        1
5                0         0       1      0        0        0
```

e) Tính khoảng cách từ item của người dùng hiện hành với các item còn lại trong tập dữ liệu

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}},$$

Thông tin user cần gợi ý: đã xem bộ phim thứ 100 trong tập dữ liệu
Tính giá trị cosine của bộ phim thứ 100 với 99 phim đầu tiên trong tập dữ liệu để tìm ra bộ phim gần nhất với người dùng này

```
from sklearn.metrics.pairwise import cosine_similarity
cosine_similarity(df.ix[100:101], df.ix[1:99])
```

B. **Bài tập:**

Để làm tăng độ chính xác của hệ thống gợi ý, dựa vào nội dung ở câu thực hành số 2, bổ sung thêm 1 số thông tin sau:

- Người dùng đã xem 4 bộ phim thay vì 1 bộ phim
- Bổ sung thêm đánh giá của người dùng trên bộ phim thay vì chỉ là thông tin xem bộ phim mà không biết độ yêu thích cho các bộ phim đã xem

Cho thông tin profile người dùng như bên dưới: người dùng đã xem 4 bộ phim với đánh giá như trong bảng, anh chị hãy gợi ý 5 bộ phim phù hợp nhất với người dùng này trong tập dữ liệu phim ở ví dụ trên.

Tên phim	**Đánh giá**	**Thể loại**
Phim 1	6 điểm	Drama,Thriller
Phim 2	10 điểm	Adventure, Comedy, Crime, Romance
Phim 3	8 điểm	Comedy
Phim 4	8 điểm	Comedy