In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline

import warnings
warnings.filterwarnings('ignore')
```

In [2]:
```
users = pd.read_csv('BX-Users.csv', encoding='latin-1')
```

In [3]:
```
users.shape
```

Out[3]:
```
(278859, 3)
```

In [4]:
```
users.head()
```

Out[4]:

	user_id	Location	Age
0	1	nyc, new york, usa	NaN
1	2	stockton, california, usa	18.0
2	3	moscow, yukon territory, russia	NaN
3	4	porto, v.n.gaia, portugal	17.0
4	5	farnborough, hants, united kingdom	NaN

In [5]:
```
books = pd.read_csv('BX-Books.csv', encoding='latin-1')
```

In [6]:
```
books.head()
```

Out[6]:

	isbn	book_title	book_author	year_of_publication	publisher
0	195153448	Classical Mythology	Mark P. O. Morford	2002	Oxford University Press
1	2005018	Clara Callan	Richard Bruce Wright	2001	HarperFlamingo Canada
2	60973129	Decision in Normandy	Carlo D'Este	1991	HarperPerennial
3	374157065	Flu: The Story of the Great Influenza Pandemic...	Gina Bari Kolata	1999	Farrar Straus Giroux
4	393045218	The Mummies of Urumchi	E. J. W. Barber	1999	W. W. Norton &amp; Company

In [7]:
```
books.shape
```

Out[7]:
```
(271379, 5)
```

In [8]:
```
books.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 271379 entries, 0 to 271378
Data columns (total 5 columns):
 #   Column               Non-Null Count   Dtype 
---  ------               --------------   ----- 
 0   isbn                 271379 non-null  object
 1   book_title           271379 non-null  object
 2   book_author          271378 non-null  object
 3   year_of_publication  271379 non-null  object
 4   publisher            271377 non-null  object
dtypes: object(5)
memory usage: 10.4+ MB
```

In [9]:
```
users.shape
```
Out[9]:
```
(278859, 3)
```

In [10]:
```
users.isnull().sum(axis=0)
```
Out[10]:
```
user_id          0
Location         1
Age         110763
dtype: int64
```

In [11]:
```
users.dropna(how='any', inplace=True)
```

In [12]:
```
users.isnull().sum(axis=0)
```
Out[12]:
```
user_id     0
Location    0
Age         0
dtype: int64
```

In [13]:
```
ratings=pd.read_csv('BX-Book-Ratings.csv', encoding='latin-1',nrows=10000)
```

In [14]:
```
ratings.head()
```
Out[14]:

	user_id	isbn	rating
0	276725	034545104X	0
1	276726	155061224	5
2	276727	446520802	0
3	276729	052165615X	3
4	276729	521795028	6

In [15]:
```
ratings.shape
```
Out[15]:
```
(10000, 3)
```

In [16]:
```
ratings.describe()
```

Out[16]:

	user_id	rating
count	10000.000000	10000.000000
mean	265844.379600	1.974700
std	56937.189618	3.424884
min	2.000000	0.000000
25%	277478.000000	0.000000
50%	278418.000000	0.000000
75%	278418.000000	4.000000
max	278854.000000	10.000000

In [17]:
```
ratings.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 3 columns):
 #   Column   Non-Null Count  Dtype 
---  ------   --------------  ----- 
 0   user_id  10000 non-null  int64 
 1   isbn     10000 non-null  object
 2   rating   10000 non-null  int64 
dtypes: int64(2), object(1)
memory usage: 234.5+ KB
```

In [18]:
```
#Take a quick look at the number of unique users and books
ratings_books=pd.merge(ratings,books,on='isbn')
```

In [19]:
```
ratings_books.head()
```

Out[19]:

	user_id	isbn	rating	book_title	book_author	year_of_publication	publisher
0	276725	034545104X	0	Flesh Tones: A Novel	M. J. Rose	2002	Ballantine Books
1	276726	155061224	5	Rites of Passage	Judith Rae	2001	Heinle
2	276727	446520802	0	The Notebook	Nicholas Sparks	1996	Warner Books
3	278418	446520802	0	The Notebook	Nicholas Sparks	1996	Warner Books
4	276729	052165615X	3	Help!: Level 1	Philip Prowse	1999	Cambridge University Press

In [20]:
```
ratings_books.shape
```

Out[20]:
```
(8701, 7)
```

In [ ]:

In [21]:
```
unique_users=len(ratings_books['user_id'].unique())
unique_users
```

Out[21]:
```
828
```

In [22]:
```
unique_books=len(ratings_books['isbn'].unique())
unique_books
```

Out[22]:
```
8051
```

In [23]:
```
# Convert and print length of isbn list
isbn_list = ratings_books.isbn.unique()
print(" Unique books:", len(isbn_list))
def get_isbn_numeric_id(isbn):
    itemindex = np.where(isbn_list==isbn)
    return itemindex[0][0]
```

```
 Unique books: 8051
```

In [ ]:

In [24]:
```
# Convert and print length of user_id list
userid_list = ratings_books.user_id.unique()
print(" Unique users:", len(userid_list))
def get_user_id_numeric_id(user_id):
    itemindex = np.where(userid_list==user_id)
    return itemindex[0][0]
```

```
 Unique users: 828
```

In [25]:
```
ratings_books['isbn_order'] = ratings_books['isbn'].apply(get_isbn_numeric_id)
```

In [26]:
```
ratings_books['user_order'] = ratings_books['user_id'].apply(get_user_id_numeric_i
```

In [27]:
```
ratings_books.head()
```

Out[27]:

	user_id	isbn	rating	book_title	book_author	year_of_publication	publisher	isbn_orde
0	276725	034545104X	0	Flesh Tones: A Novel	M. J. Rose	2002	Ballantine Books	
1	276726	155061224	5	Rites of Passage	Judith Rae	2001	Heinle	
2	276727	446520802	0	The Notebook	Nicholas Sparks	1996	Warner Books	
3	278418	446520802	0	The Notebook	Nicholas Sparks	1996	Warner Books	
4	276729	052165615X	3	Help!: Level 1	Philip Prowse	1999	Cambridge University Press	

In [28]:
```
ratings_books.columns
```

Out[28]:
```
Index(['user_id', 'isbn', 'rating', 'book_title', 'book_author',
       'year_of_publication', 'publisher', 'isbn_order', 'user_order'],
      dtype='object')
```

In [29]:
```
#Re-index the columns to build a matrix
new_columns=['isbn_order', 'user_order','user_id', 'isbn', 'rating', 'book_title',
ratings_books=ratings_books.reindex(columns=new_columns)
ratings_books.head()
```

Out[29]:

	isbn_order	user_order	user_id	isbn	rating	book_title	book_author	year_of_publicatio
0	0	0	276725	034545104X	0	Flesh Tones: A Novel	M. J. Rose	200
1	1	1	276726	155061224	5	Rites of Passage	Judith Rae	200
2	2	2	276727	446520802	0	The Notebook	Nicholas Sparks	199
3	2	3	278418	446520802	0	The Notebook	Nicholas Sparks	199
4	3	4	276729	052165615X	3	Help!: Level 1	Philip Prowse	199

In [30]:
```
#Split your data into two sets (training and testing)
```

In [31]:
```
from sklearn.model_selection import train_test_split
train_data,test_data=train_test_split(ratings_books,test_size=0.20)
```

In [32]:
```
# Create user-book matrix for training
train_data_matrix = np.zeros((unique_users, unique_books))
for line in train_data.itertuples():
    train_data_matrix[line[1]-1, line[2]-1] = line[3]

# Create user-book matrix for testing
test_data_matrix = np.zeros((n_users, n_books))
for line in test_data.itertuples():
    test_data_matrix[line[1]-1, line[2]-1] = line[3]
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
Input In [32], in <cell line: 3>()
      2 train_data_matrix = np.zeros((unique_users, unique_books))
      3 for line in train_data.itertuples():
----> 4     train_data_matrix[line[1]-1, line[2]-1] = line[3]
      6 # Create user-book matrix for testing
      7 test_data_matrix = np.zeros((n_users, n_books))

IndexError: index 7576 is out of bounds for axis 0 with size 828
```

In [33]:
```
# Importing pairwise_distances function
from sklearn.metrics.pairwise import pairwise_distances
user_similarity = pairwise_distances(train_data_matrix, metric='cosine')
item_similarity = pairwise_distances(train_data_matrix.T, metric='cosine')
```

In [34]:
```
user_similarity
```

Out[34]:
```
array([[0., 1., 1., ..., 1., 1., 1.],
       [1., 0., 1., ..., 1., 1., 1.],
       [1., 1., 0., ..., 1., 1., 1.],
       ...,
       [1., 1., 1., ..., 0., 1., 1.],
       [1., 1., 1., ..., 1., 0., 1.],
       [1., 1., 1., ..., 1., 1., 0.]])
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: