Problem statement.

Build a recommender system by using cosine simillarties score.

In [200]: ▶|
```python
import pandas as pd
import numpy as np
import re
import nltk
import matplotlib.pyplot as plt
%matplotlib inline
import warnings
import os
import requests
import seaborn as sns
from sklearn.metrics import pairwise_distances

from nltk.corpus import stopwords
nltk.download("stopwords")

from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics.pairwise import cosine_similarity


from scipy.sparse import csr_matrix
from sklearn.neighbors import NearestNeighbors # unsupervised ML algo


from PIL import Image

warnings.filterwarnings('ignore')
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\Admin\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

In [175]: ▶|
```python
book_data = pd.read_csv("books.csv",encoding='latin-1')
book_data.head()
```

Out[175]:

|   | User.ID | Book.Title | Book.Rating |
|---|---------|------------|-------------|
| 0 | 276726 | Classical Mythology | 5 |
| 1 | 276729 | Clara Callan | 3 |
| 2 | 276729 | Decision in Normandy | 6 |
| 3 | 276736 | Flu: The Story of the Great Influenza Pandemic... | 8 |
| 4 | 276737 | The Mummies of Urumchi | 6 |

In [176]: ▶|
```python
books_ = pd.read_csv("books_data.csv")
books_.book_title.unique()
```

Out[176]:
```
array(['Classical Mythology', 'Clara Callan', 'Decision in Normandy', ...,
       'Why stop?: A guide to Texas historical roadside markers',
       "The Are You Being Served? Stories: 'Camping In' and Other Fiascoes",
       'Dallas Street Map Guide and Directory, 2000 Edition'],
      dtype=object)
```
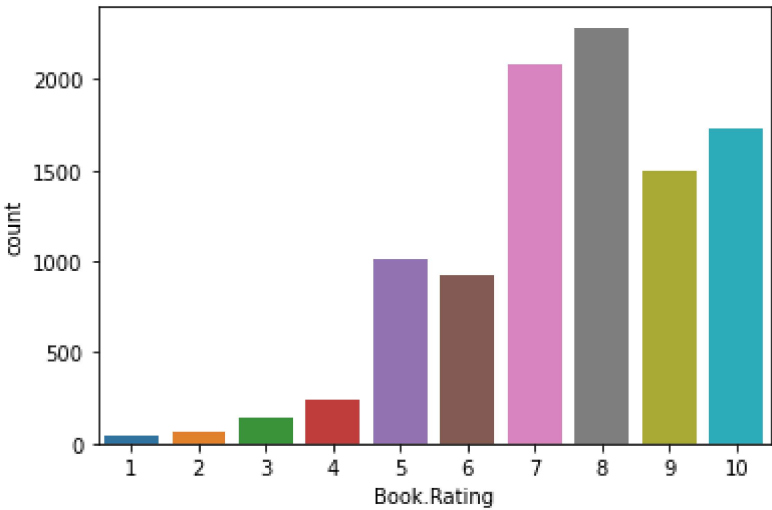
In [177]: ▶|
```python
books_[books_.book_title == "Clara Callan"].iloc[0].img_l
```

Out[177]: `'http://images.amazon.com/images/P/0002005018.01.LZZZZZZZ.jpg'`

In [178]: ▶|
```python
book_data['Book.Title'] = book_data['Book.Title'].apply(lambda x: x.replace("&amp;", "") if "&amp;" in x els
```

In [183]:  ▶|
```python
sns.countplot(x='Book.Rating', data=book_data)
```

Out[183]:  `<AxesSubplot:xlabel='Book.Rating', ylabel='count'>`



In [ ]:  ▶|
```python
title=[]
images=[]
for i in book_data["Book.Title"].values:
    # print(i)
    if i in list(books_["book_title"].unique()):
        #print(i + " present")
        title.append(i)
        images.append(books_[books_.book_title == i].iloc[0].img_l)
    else:
        title.append(i)
        images.append("")
```

In [184]:  ▶|
```python
book_data["URL"] = images
book_data.tail()
```

Out[184]:

|  | User.ID | Book.Title | Book.Rating | URL |
|---|---|---|---|---|
| **9995** | 162121 | American Fried: Adventures of a Happy Eater. | 7 | http://images.amazon.com/images/P/0140040188.0... |
| **9996** | 162121 | Cannibal In Manhattan | 9 | http://images.amazon.com/images/P/0330305336.0... |
| **9997** | 162121 | How to Flirt: A Practical Guide | 7 | http://images.amazon.com/images/P/0843133805.0... |
| **9998** | 162121 | Twilight | 8 | http://images.amazon.com/images/P/0671644076.0... |
| **9999** | 162129 | Kids Say the Darndest Things | 6 | http://images.amazon.com/images/P/0553110438.0... |

Item based collaborative filtering

In [196]:  ▶|
```python
# Renaming columns
book_data = book_data.rename({'User.ID':'user_id', 'Book.Title':'book_title', 'Book.Rating':'book_rating'},
book_data.head()
```

Out[196]:

|  | user_id | book_title | book_rating | URL |
|---|---|---|---|---|
| **0** | 276726 | Classical Mythology | 5 | http://images.amazon.com/images/P/0195153448.0... |
| **1** | 276729 | Clara Callan | 3 | http://images.amazon.com/images/P/0002005018.0... |
| **2** | 276729 | Decision in Normandy | 6 | http://images.amazon.com/images/P/0060973129.0... |
| **3** | 276736 | Flu: The Story of the Great Influenza Pandemic... | 8 | http://images.amazon.com/images/P/0374157065.0... |
| **4** | 276737 | The Mummies of Urumchi | 6 | http://images.amazon.com/images/P/0393045218.0... |

In [197]: ▶|
```python
books_df = book_data.pivot_table(index='user_id',
                                 columns='book_title',
                                 values='book_rating').reset_index(drop=True)
books_df.head()
```

Out[197]:

| book_title | Jason, Madison &amp | Stories;Merril;1985;McClelland &amp | Other Stories &amp | Repairing PC Drives &amp | '48 | 'O Au No Keia: Voices from Hawai'I's Mahu and Transgender Communities | ...AND THE HORSE HE RODE IN ON : THE PEOPLE V. KENNETH STARR | 01-01-00: A Novel of the Millennium | 1,401 More Things That P*Ss Me Off | 10 Commandments Of Dating |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN |
| 1 | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN |
| 2 | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN |
| 3 | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN |
| 4 | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN |

5 rows × 9659 columns

In [198]: ▶|
```python
books_df.index = book_data['user_id'].unique()
```

In [199]: ▶|
```python
books_df.fillna(0, inplace=True)
books_df.head()
```

Out[199]:

| book_title | Jason, Madison &amp | Stories;Merril;1985;McClelland &amp | Other Stories &amp | Repairing PC Drives &amp | '48 | 'O Au No Keia: Voices from Hawai'I's Mahu and Transgender Communities | ...AND THE HORSE HE RODE IN ON : THE PEOPLE V. KENNETH STARR | 01-01-00: A Novel of the Millennium | 1,401 More Things That P*Ss Me Off | 10 Commandments Of Dating |
|---|---|---|---|---|---|---|---|---|---|---|
| 276726 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 276729 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 276736 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 276737 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 276744 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |

5 rows × 9659 columns

In [202]: ▶|
```python
#Calculating Cosine Similarity between Users
user_similarity = 1 - pairwise_distances(books_df.values,metric='cosine')
#user_similarity
```

In [203]: ▶|
```python
user_sim_df = pd.DataFrame(user_similarity)

user_sim_df.index = book_data.user_id.unique()
user_sim_df.columns = book_data.user_id.unique()
user_sim_df.head()
```

Out[203]:

| | 276726 | 276729 | 276736 | 276737 | 276744 | 276745 | 276747 | 276748 | 276751 | 276754 | ... | 162085 | 162091 | 162092 | 162095 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 276726 | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... | 0.0 | 0.0 | 0.0 | 0.0 |
| 276729 | 0.0 | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... | 0.0 | 0.0 | 0.0 | 0.0 |
| 276736 | 0.0 | 0.0 | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... | 0.0 | 0.0 | 0.0 | 0.0 |
| 276737 | 0.0 | 0.0 | 0.0 | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... | 0.0 | 0.0 | 0.0 | 0.0 |
| 276744 | 0.0 | 0.0 | 0.0 | 0.0 | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... | 0.0 | 0.0 | 0.0 | 0.0 |

5 rows × 2182 columns

In [204]: ▶|
```python
# Filling diagonal values with 0
np.fill_diagonal(user_similarity, 0)
user_sim_df.iloc[0:5, 0:5]
```

Out[204]:

|         | 276726 | 276729 | 276736 | 276737 | 276744 |
|---------|--------|--------|--------|--------|--------|
| 276726  | 0.0    | 0.0    | 0.0    | 0.0    | 0.0    |
| 276729  | 0.0    | 0.0    | 0.0    | 0.0    | 0.0    |
| 276736  | 0.0    | 0.0    | 0.0    | 0.0    | 0.0    |
| 276737  | 0.0    | 0.0    | 0.0    | 0.0    | 0.0    |
| 276744  | 0.0    | 0.0    | 0.0    | 0.0    | 0.0    |

In [268]: ▶|
```python
#Most Similar Users
user_sim_df.idxmax(axis=1)[0:10]
```

Out[268]:
```
276726    276726
276729    276726
276736    276726
276737    276726
276744    276726
276745    276726
276747    276726
276748    161677
276751    276726
276754    276726
dtype: int64
```

In [254]: ▶|
```python
similar_ = book_data[(book_data['user_id']==276729) | (book_data['user_id']==276726)]
similar_
```

Out[254]:

|   | user_id | book_title          | book_rating | URL                                          |
|---|---------|---------------------|-------------|----------------------------------------------|
| 0 | 276726  | Classical Mythology | 5           | http://images.amazon.com/images/P/0195153448.0... |
| 1 | 276729  | Clara Callan        | 3           | http://images.amazon.com/images/P/0002005018.0... |
| 2 | 276729  | Decision in Normandy | 6          | http://images.amazon.com/images/P/0060973129.0... |

In [275]: ▶|
```python
def book_recommender(user1, user2):
    similar_users = book_data[(book_data['user_id']==user1) | (book_data['user_id']==user2)]
    #print(similar_users)
    fig, axs = plt.subplots(1, similar_users.shape[0],figsize=(15,5))
    fig.suptitle('Books recommendation based on user similarity', size = 22)


    for i in range(len(similar_users['book_title'].tolist())):


        url = similar_users.iloc[i].URL
        #print(url)
        if url !="":
            #print(url)
            im = Image.open(requests.get(url, stream=True).raw)
            axs[i].imshow(im)
            axs[i].axis("off")
            axs[i].set_title('{0} \n Rating: {1}  \n User ID: {2}'.format(similar_users.iloc[i].book_title,

            fig.show()

        else:
            axs[i].axis("off")
            """axs[i].set_title('{0} \n Rating: {1}'.format(similar_users.iloc[i].book_title, similar_users.
                        y=-0.18,color="red",fontsize=10)"""
            fig.show()
```
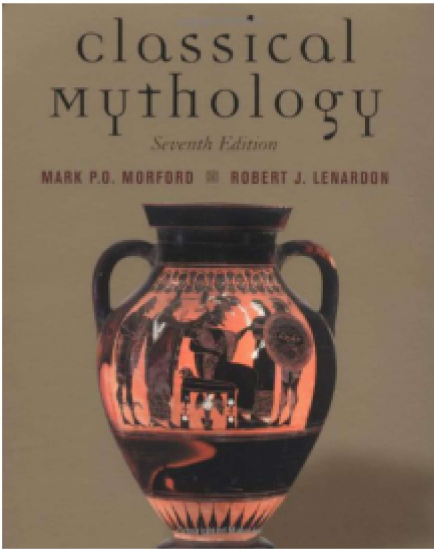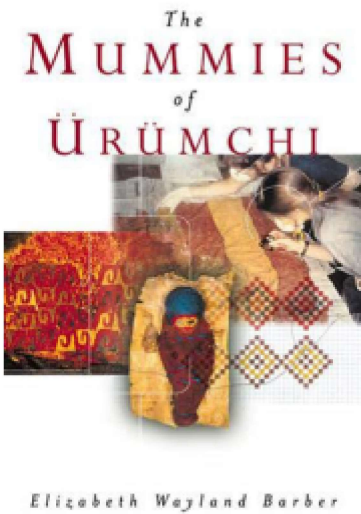
In [276]:  ▶| `book_recommender(276737, 276726)`

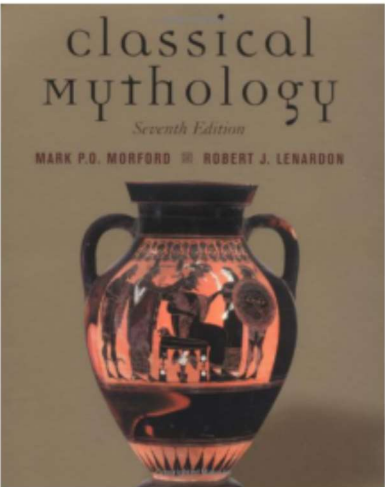## Books recommendation based on user similarity



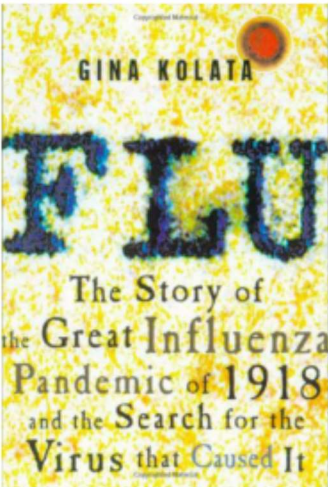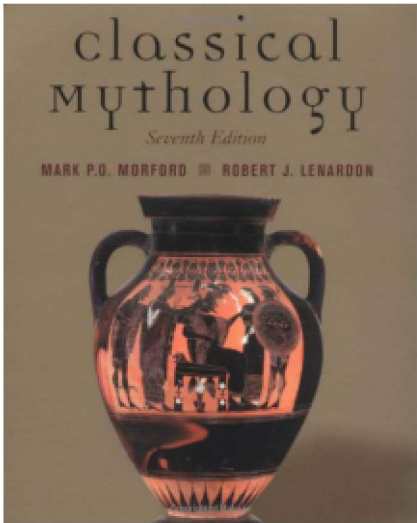**Classical Mythology**
**Rating: 5**
**User ID: 276726**



**The Mummies of Urumchi**
**Rating: 6**
**User ID: 276737**

In [277]:  ▶| `book_recommender(276736, 276726)`

## Books recommendation based on user similarity



**Classical Mythology**
**Rating: 5**
**User ID: 276726**



**Flu: The Story of the Great Influenza Pandemic of 1918 and the Search for the Virus That Caused It**
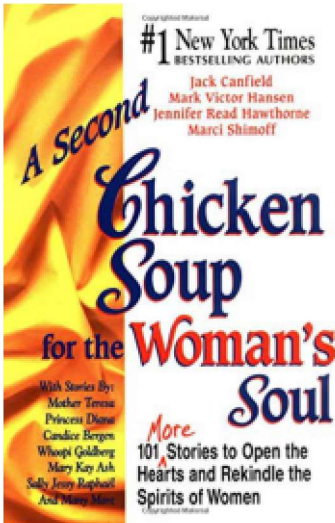**Rating: 8**
**User ID: 276736**

In [282]: ▶| `book_recommender(276754,276726)`

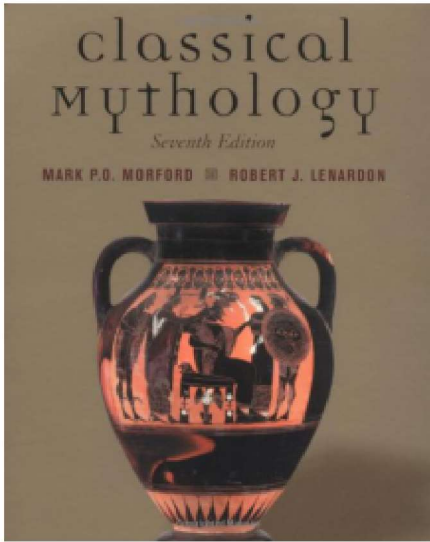## Books recommendation based on user similarity
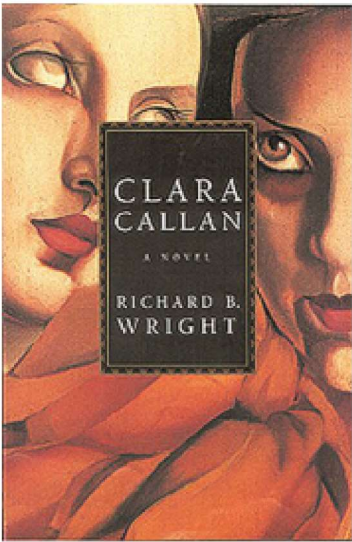


**Classical Mythology**
Rating: 5
User ID: 276726

**A Second Chicken Soup for the Woman's Soul (Chicken Soup for the Soul Series)**
Rating: 8
User ID: 276754

In [278]: ▶| `book_recommender(276729, 276726)`

## Books recommendation based on user similarity
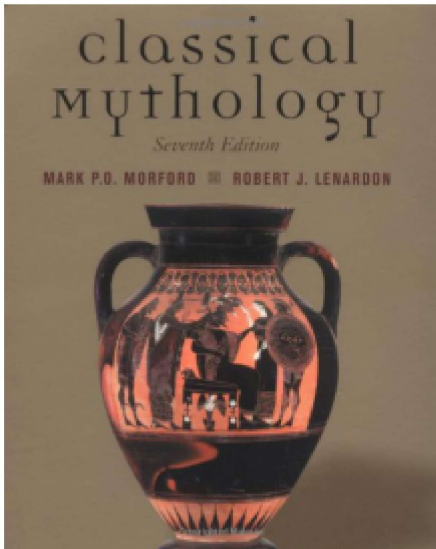


**Classical Mythology**
Rating: 5
User ID: 276726
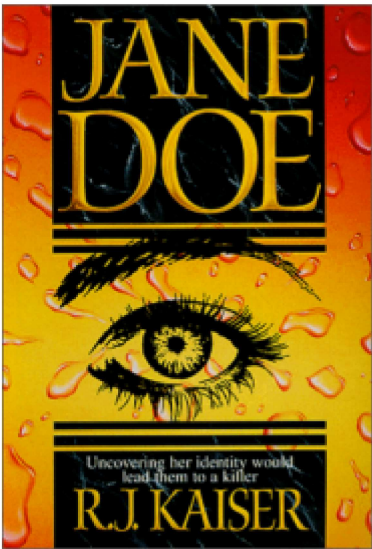
**Clara Callan**
Rating: 3
User ID: 276729

**Decision in Normandy**
Rating: 6
User ID: 276729

In [279]: ▶| `book_recommender(276751, 276726)`

## Books recommendation based on user similarity



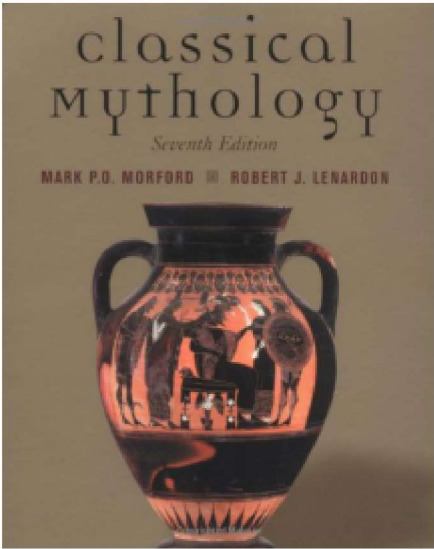**Classical Mythology**
Rating: 5
User ID: 276726

**Jane Doe**
Rating: 8
User ID: 276751

In [280]:  ▶|  `book_recommender(276744, 276726)`

## Books recommendation based on user similarity



Classical Mythology
Rating: 5
User ID: 276726

The Kitchen God's Wife
Rating: 7
User ID: 276744

In [209]:  ▶|  `user1=book_data[book_data['user_id']==276726]`

In [210]:  ▶|  `user2=book_data[book_data['user_id']==276729]`

In [211]:  ▶|  `pd.merge(user1,user2,on='book_title',how='outer')`

Out[211]:

|   | user_id_x | book_title | book_rating_x | URL_x | user_id_y | book_rating_y | |
|---|-----------|------------|---------------|-------|-----------|---------------|---|
| 0 | 276726.0 | Classical Mythology | 5.0 | http://images.amazon.com/images/P/0195153448.0... | NaN | NaN | |
| 1 | NaN | Clara Callan | NaN | NaN | 276729.0 | 3.0 | http://images.amazor |
| 2 | NaN | Decision in Normandy | NaN | NaN | 276729.0 | 6.0 | http://images.amazor |