1. Import neccessery libraries

In [41]:
```
import numpy as np
import pandas as pd
from matplotlib import pyplot as plt
import seaborn as sns
from sklearn.metrics import pairwise_distances
```

Problem

Recommend a best book based on the ratings

2.Import data

In [42]:
```
book_data = pd.read_csv('book.1.csv',encoding =  "ISO-8859-1")
book_data
```

Out[42]:

	Unnamed: 0	User.ID	Book.Title	Book.Rating
0	1	276726	Classical Mythology	5
1	2	276729	Clara Callan	3
2	3	276729	Decision in Normandy	6
3	4	276736	Flu: The Story of the Great Influenza Pandemic...	8
4	5	276737	The Mummies of Urumchi	6
...	...	...	...	...
9995	9996	162121	American Fried: Adventures of a Happy Eater.	7
9996	9997	162121	Cannibal In Manhattan	9
9997	9998	162121	How to Flirt: A Practical Guide	7
9998	9999	162121	Twilight	8
9999	10000	162129	Kids Say the Darndest Things	6

10000 rows × 4 columns

In [43]:
```
bd1 = book_data.iloc[:,1:]
```

In [44]:
```
bd1.columns = ['UserID','Title','bookrating']
```

In [45]:
```
bd1.head()
```

Out[45]:

	UserID	Title	bookrating
0	276726	Classical Mythology	5
1	276729	Clara Callan	3

	UserID	Title	bookrating
2	276729	Decision in Normandy	6
3	276736	Flu: The Story of the Great Influenza Pandemic...	8

3. Data understanding

In [46]:
```
bd1.shape
```

Out[46]:
```
(10000, 3)
```

In [47]:
```
bd1.isna().sum()
```

Out[47]:
```
UserID        0
Title         0
bookrating    0
dtype: int64
```

In [48]:
```
bd1.dtypes
```

Out[48]:
```
UserID         int64
Title         object
bookrating     int64
dtype: object
```

In [49]:
```
bd1.describe(include='all')
```

Out[49]:

	UserID	Title	bookrating
count	10000.000000	10000	10000.00000
unique	NaN	9659	NaN
top	NaN	Fahrenheit 451	NaN
freq	NaN	5	NaN
mean	95321.249800	NaN	7.56630
std	117645.703609	NaN	1.82152
min	8.000000	NaN	1.00000
25%	2103.000000	NaN	7.00000
50%	3757.000000	NaN	8.00000
75%	162052.000000	NaN	9.00000
max	278854.000000	NaN	10.00000

In [50]:
```
bd1['UserID'].unique()
```

Out[50]:
```
array([276726, 276729, 276736, ..., 162113, 162121, 162129], dtype=int64)
```

In [51]:
```
bd1['UserID'].nunique()
```

```
Out[51]: 2182
```

what are the Titles this datasets holds?

```
In [52]: bd1['Title'].unique()
```

```
Out[52]: array(['Classical Mythology', 'Clara Callan', 'Decision in Normandy', ...,
                'How to Flirt: A Practical Guide', 'Twilight',
                'Kids Say the Darndest Things'], dtype=object)
```

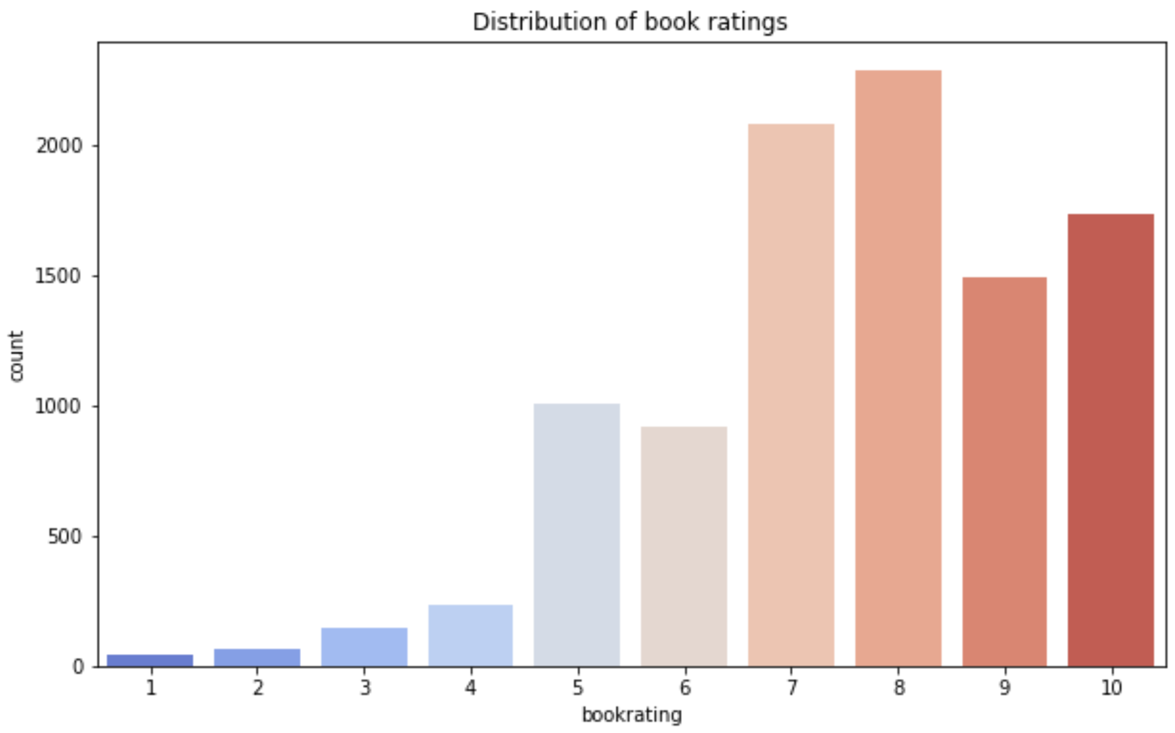
```
In [53]: bd1['Title'].nunique()
```

```
Out[53]: 9659
```

```
In [54]: palette = sns.color_palette("coolwarm", 10)
```

```
In [55]: fig, ax = plt.subplots(figsize=(10, 6))
         sns.countplot(x='bookrating', data=bd1, palette=palette)
         ax.set_title('Distribution of book ratings')

         plt.show()
```



The majority of ratings is between 5 and 10. Most often users tend to rate books for 8. Second the most frequent score is 7.

4. COLLABORATIVE FILTERING - UBCF

1. UBCF with correlation matrix as a metric

In [56]:

```
ubcf_data = pd.pivot_table( data=bd1,values='bookrating',index='Title', co
ubcf_data.columns = bd1.UserID.unique()
ubcf_data
```

Out[56]:

	276726	276729	276736	276737	276744	276745	276747	2767
Title								
Jason, Madison &amp	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
Other Stories;Merril;1985;McClelland &amp	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
Repairing PC Drives &amp	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
'48	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
'O Au No Keia: Voices from Hawai'I's Mahu and Transgender Communities	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
...	...	...	...	...	...	...	...	
\Surely You're Joking, Mr. Feynman!\: Adventures of a Curious Character	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
\Well, there's your problem\: Cartoons	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
iI Paradiso Degli Orchi	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
stardust	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
Ã?Â?bermorgen.	0.0	0.0	0.0	0.0	0.0	0.0	0.0	

9659 rows × 2182 columns

In [57]:

```
ubcf_data.corr().round(2)
```

Out[57]:

	276726	276729	276736	276737	276744	276745	276747	276748	276751	276754
276726	1.0	-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
276736	-0.0	-0.0	1.0	-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
276744	-0.0	-0.0	-0.0	-0.0	1.0	-0.0	-0.0	-0.0	-0.0	-0.0
...	...	...	...	...	...	...	...	...	...	...
162107	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0
162109	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0
162113	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0
162121	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0
162129	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0

2182 rows × 2182 columns

2. UBCF by using Euclideon Distance as a Metric

In [58]:

```
ubcf_data_ecd = pd.pivot_table(data=bd1,values='bookrating',index='UserID',
ubcf_data_ecd.index=bd1.UserID.unique()
ubcf_data_ecd
```

Out[58]:

Title	Jason, Madison &amp	Other Stories;Merril;1985;McClelland &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	01-0 A Nov Millen
276726	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	
276736	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	
276744	0.0	0.0	0.0	0.0	0.0	0.0	
...	...	...	...	...	...	...	
162107	0.0	0.0	0.0	0.0	0.0	0.0	
162109	0.0	0.0	0.0	0.0	0.0	0.0	
162113	0.0	0.0	0.0	0.0	0.0	0.0	
162121	0.0	0.0	0.0	0.0	0.0	0.0	
162129	0.0	0.0	0.0	0.0	0.0	0.0	

2182 rows × 9659 columns

In [59]:

```
euclideon_ubcf = pairwise_distances(X = ubcf_data_ecd,metric='euclidean')
euclideon_ubcf_df = pd.DataFrame(data = euclideon_ubcf)
euclideon_ubcf_df.index = bd1.UserID.unique()
euclideon_ubcf_df.columns = bd1.UserID.unique()
euclideon_ubcf_df
```

Out[59]:

	276726	276729	276736	276737	276744	276745	276747	276748
276726	0.000000	16.031220	16.031220	17.916473	17.521415	17.378147	18.439089	16.431677
276729	16.031220	0.000000	8.485281	11.661904	11.045361	10.816654	12.449900	9.219544
276736	16.031220	8.485281	0.000000	11.661904	11.045361	10.816654	12.449900	9.219544
276737	17.916473	11.661904	11.661904	0.000000	13.638182	13.453624	14.798649	12.206556
276744	17.521415	11.045361	11.045361	13.638182	0.000000	12.922848	14.317821	11.618950
...	...	...	...	...	...	...	...	...

	276726	276729	276736	276737	276744	276745	276747	276748
162107	16.881943	10.000000	10.000000	12.806248	12.247449	12.041595	13.527749	10.630146
162109	17.378147	10.816654	10.816654	13.453624	12.922848	12.727922	14.142136	11.401754
162113	29.051678	25.670995	25.670995	26.888659	26.627054	26.532998	27.239677	25.922963
162121	16.881943	10.000000	10.000000	12.806248	12.247449	12.041595	13.527749	10.630146
162129	22.737634	18.220867	18.220867	19.899749	19.544820	19.416488	20.371549	18.574176

3. UBCF by using Cosine Distance as a Metric

In [60]:
```
ubcf_data_cosine = pd.pivot_table(data=bd1,values='bookrating',index='User
ubcf_data_cosine.index=bd1.UserID.unique()
ubcf_data_cosine
```

Out[60]:

Title	Jason, Madison &amp	Other Stories;Merril;1985;McClelland &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	01-0 A Nov Millen
276726	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	
276736	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	
276744	0.0	0.0	0.0	0.0	0.0	0.0	
...	...	...	...	...	...	...	
162107	0.0	0.0	0.0	0.0	0.0	0.0	
162109	0.0	0.0	0.0	0.0	0.0	0.0	
162113	0.0	0.0	0.0	0.0	0.0	0.0	
162121	0.0	0.0	0.0	0.0	0.0	0.0	
162129	0.0	0.0	0.0	0.0	0.0	0.0	

2182 rows × 9659 columns

In [61]:
```
cosine_distances = 1 - pairwise_distances(X = ubcf_data_cosine,metric='cos
cosine_distances_df = pd.DataFrame(data = cosine_distances)
cosine_distances_df.index = bd1.UserID.unique()
cosine_distances_df.columns = bd1.UserID.unique()
cosine_distances_df
```

Out[61]:

	276726	276729	276736	276737	276744	276745	276747	276748	276751	276754
276726	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

	276726	276729	276736	276737	276744	276745	276747	276748	276751	276754
276729	0.0	1.0	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
276737	0.0	0.0	0.0	1.0	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
...	...	...	...	...	...	...	...	...	...	...
162107	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
162109	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
162113	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
162121	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
162129	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Let's filter the data with first 50 users.

In [62]:

```
first_50_users = cosine_distances_df.iloc[:50,:50]
first_50_users
```

Out[62]:

	276726	276729	276736	276737	276744	276745	276747	276748	276751	276754
276726	1.0	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
276736	0.0	0.0	1.0	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
276744	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0
276745	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0
276747	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0
276748	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0
276751	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0
276754	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0
276755	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276760	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276762	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276768	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276772	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276774	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276780	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276786	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276788	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

	276726	276729	276736	276737	276744	276745	276747	276748	276751	276754
276796	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276798	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276800	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276804	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276808	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276811	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276812	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276813	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276814	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276820	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276822	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276827	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276828	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276830	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276832	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276835	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276837	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276842	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276847	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276848	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276850	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276853	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276854	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276857	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276859	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276861	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276862	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276863	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276866	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276870	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276872	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

In [63]:

```
np.fill_diagonal(a = first_50_users.to_numpy(),val = 0)
```

In [64]:

```
first_50_users
```

Out[64]:

	276726	276729	276736	276737	276744	276745	276747	276748	276751	276754
276726	0.0	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	0.0
276736	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	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	0.0
276745	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276747	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276748	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276751	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276754	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276755	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276760	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276762	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276768	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276772	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276774	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276780	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276786	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276788	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276796	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276798	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276800	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276804	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276808	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276811	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276812	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276813	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276814	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276820	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276822	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276827	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276828	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276830	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276832	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276835	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

	276726	276729	276736	276737	276744	276745	276747	276748	276751	276754
276837	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276842	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276847	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276848	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276850	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276853	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276854	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276857	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276859	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276861	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276862	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276863	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276866	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
276870	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

In [65]:
```
first_50_users.idxmax()
```

Out[65]:
```
276726    276726
276729    276726
276736    276726
276737    276726
276744    276726
276745    276726
276747    276726
276748    276726
276751    276726
276754    276726
276755    276726
276760    276726
276762    276726
276768    276726
276772    276726
276774    276726
276780    276726
276786    276726
276788    276726
276796    276726
276798    276726
276800    276726
276804    276726
276808    276726
276811    276726
276812    276726
276813    276726
276814    276726
276820    276726
276822    276726
276827    276726
276828    276726
276830    276726
```

```
276832    276726
276835    276726
276837    276726
276842    276726
276847    276726
276848    276726
276850    276726
276853    276726
276854    276726
276857    276726
276859    276726
276861    276726
276862    276726
276863    276726
276866    276726
276870    276726
276872    276726
```

Let's try to see how 5th and 17th user are correlated.

In [66]:
```
book_data[(bd1['UserID'] == 5) | (bd1['UserID']==17)]
```

Out[66]:

	Unnamed: 0	User.ID	Book.Title	Book.Rating
2413	2414	17	Conversations With Dogbert: A Dilbert Book	7
2414	2415	17	The Dilbert Bunch: A Dilbert Book (Main Street...	5
2415	2416	17	You Don't Need Experience if You've Got Attitude	6
2416	2417	17	The Boss: Nameless, Blameless and Shameless (A...	3

Let's try to see how 134th and 17th user are correlated

In [67]:
```
book_data[(bd1['UserID'] == 130) | (bd1['UserID']==17)]
```

Out[67]:

	Unnamed: 0	User.ID	Book.Title	Book.Rating
2413	2414	17	Conversations With Dogbert: A Dilbert Book	7
2414	2415	17	The Dilbert Bunch: A Dilbert Book (Main Street...	5
2415	2416	17	You Don't Need Experience if You've Got Attitude	6
2416	2417	17	The Boss: Nameless, Blameless and Shameless (A...	3

In [ ]:
```

```

In [ ]:
```

```