In [1]:

```
import pandas as pd
```

In [2]:

```
book_data=pd.read_csv('book.csv')
book_data
```

Out[2]:

	ChildBks	YouthBks	CookBks	DoItYBks	RefBks	ArtBks	GeogBks	ItalCook	ItalAtlas
0	0	1	0	1	0	0	1	0	0
1	1	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0
3	1	1	1	0	1	0	1	0	0
4	0	0	1	0	0	0	1	0	0
...	...	...	...	...	...	...	...	...	...
1995	0	0	1	0	0	1	1	1	0
1996	0	0	0	0	0	0	0	0	0
1997	0	0	0	0	0	0	0	0	0
1998	0	0	1	0	0	0	0	0	0
1999	0	0	0	0	0	0	0	0	0

2000 rows × 11 columns

Initail investigation

In [3]:

```
book_data.shape
```

Out[3]:

```
(2000, 11)
```

In [4]:

```
book_data.dtypes
```

Out[4]:

```
ChildBks     int64
YouthBks     int64
CookBks      int64
DoItYBks     int64
RefBks       int64
ArtBks       int64
GeogBks      int64
ItalCook     int64
ItalAtlas    int64
ItalArt      int64
Florence     int64
dtype: object
```

In [5]:

```
book_data.isnull().sum()
```

Out[5]:

```
ChildBks     0
YouthBks     0
CookBks      0
DoItYBks     0
RefBks       0
ArtBks       0
GeogBks      0
ItalCook     0
ItalAtlas    0
ItalArt      0
Florence     0
dtype: int64
```

Number of features and records in the given data set is 11 and 2000 respesctively

There is no null values in the data set

The data type is imputed correctly

In [6]:

```
book_data.columns
```

Out[6]:

```
Index(['ChildBks', 'YouthBks', 'CookBks', 'DoItYBks', 'RefBks', 'ArtBks',
       'GeogBks', 'ItalCook', 'ItalAtlas', 'ItalArt', 'Florence'],
      dtype='object')
```

Model building

In [7]:

```
from sklearn.metrics import pairwise_distances
from scipy.spatial.distance import correlation,cosine
```

In [9]:

```
user_similarity=1-pairwise_distances(X=book_data.values,metric='cosine')
user_similarity
```

Out[9]:

```
array([[1., 0., 0., ..., 0., 0., 0.],
       [0., 1., 0., ..., 0., 0., 0.],
       [0., 0., 1., ..., 0., 0., 0.],
       ...,
       [0., 0., 0., ..., 1., 0., 0.],
       [0., 0., 0., ..., 0., 1., 0.],
       [0., 0., 0., ..., 0., 0., 1.]])
```

In [10]:

```
user_df=pd.DataFrame(user_similarity)
user_df
#converting array into dataframe
```

Out[10]:

	0	1	2	3	4	5	6	7	8	
0	1.000000	0.000000	0.0	0.516398	0.408248	0.000000	0.577350	0.408248	0.408248	0.5
1	0.000000	1.000000	0.0	0.447214	0.000000	0.577350	0.000000	0.000000	0.707107	0.5
2	0.000000	0.000000	1.0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.0
3	0.516398	0.447214	0.0	1.000000	0.632456	0.258199	0.447214	0.632456	0.316228	0.8
4	0.408248	0.000000	0.0	0.632456	1.000000	0.000000	0.000000	0.000000	0.000000	0.7
...	...	...	...	...	...	...	...	...	...	
1995	0.235702	0.000000	0.0	0.365148	0.577350	0.471405	0.000000	0.000000	0.000000	0.4
1996	0.000000	0.000000	0.0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.0
1997	0.000000	0.000000	0.0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.0
1998	0.000000	0.000000	0.0	0.447214	0.707107	0.000000	0.000000	0.000000	0.000000	0.5
1999	0.000000	0.000000	0.0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.0

2000 rows × 2000 columns

In [11]:

```
import numpy as np
np.fill_diagonal(a = user_similarity, val = 0)
# filling diagonal element with Zero, since comparing one with same doen't sounds better
```

In [13]:

```
user_15=user_df.iloc[:,:14]
user_15
# taking, first 15 users
```

Out[13]:

	0	1	2	3	4	5	6	7	8	
0	0.000000	0.000000	0.0	0.516398	0.408248	0.000000	0.577350	0.408248	0.408248	0.5
1	0.000000	0.000000	0.0	0.447214	0.000000	0.577350	0.000000	0.000000	0.707107	0.5
2	0.000000	0.000000	0.0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.0
3	0.516398	0.447214	0.0	0.000000	0.632456	0.258199	0.447214	0.632456	0.316228	0.8
4	0.408248	0.000000	0.0	0.632456	0.000000	0.000000	0.000000	0.000000	0.000000	0.7
...	...	...	...	...	...	...	...	...	...	
1995	0.235702	0.000000	0.0	0.365148	0.577350	0.471405	0.000000	0.000000	0.000000	0.4
1996	0.000000	0.000000	0.0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.0
1997	0.000000	0.000000	0.0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.0
1998	0.000000	0.000000	0.0	0.447214	0.707107	0.000000	0.000000	0.000000	0.000000	0.5
1999	0.000000	0.000000	0.0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.0

2000 rows × 14 columns

In [14]:

```
user_15.idxmax(axis=0)
# Finding the relations having high similarity score
```

Out[14]:

```
0      231
1       21
2        0
3      158
4       11
5       12
6       20
7     1658
8      140
9       15
10       0
11       4
12       5
13     864
dtype: int64
```

In [15]:

```
book_data[(book_data.index==7)|(book_data.index==1658)]
```

Out[15]:

	ChildBks	YouthBks	CookBks	DoItYBks	RefBks	ArtBks	GeogBks	ItalCook	ItalAtlas
7	0	1	0	0	1	0	0	0	0
1658	0	1	0	0	1	0	0	0	0

In [16]:

```
book_data[(book_data.index==1)|(book_data.index==21)]
```

Out[16]:

	ChildBks	YouthBks	CookBks	DoItYBks	RefBks	ArtBks	GeogBks	ItalCook	ItalAtlas	Ita
1	1	0	0	0	0	0	0	0	0	
21	1	0	0	0	0	0	0	0	0	

Inference

Here, reader 1 and reader 21 are highly correlated, we can recommend the books for the reader, by comparing the books readed by other person