Recommender system using collaborative filtering

MARK9417 Machine Learning & Data Mining

Group members:
z5145438 Boxuan Hu
z5133879 Yan Wang
z5168805 Kuo Shi
z5155096 Jingwen Zhang

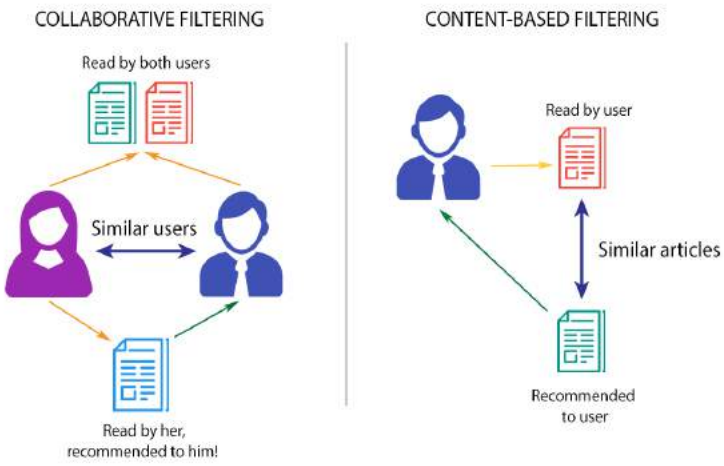
The Code, README and Data are in the google drive, here is the link:
https://drive.google.com/open?id=1WmGPU7uBINspfnwuF159GMopYbrtZoEk
Please download all files and directory in this link and place them in a directory.

Introduction

As recommender systems are in trend from last two decades, lots of big organizations and companies now use recommender systems to make prediction based on their users' historical behaviours, helping you find everything from movies to jobs, restaurants to hospitals, even romance. For example, Amazon, using behavioural and demographic data, these systems make predictions about what users will be most interested in at a particular time, resulting in high-quality, ordered, personalized suggestions. Recommender systems are practically a necessity for keeping a site content current, useful, and interesting to your visitors.

Hot Network Questions from Cross Validated
A personalized Recommended for You list from Amazon

To build a recommender system, the most widely used methods are user-based and item-based collaborative filtering. For this book recommender system, we will use the book dataset which contains 278,858 users providing 1,149,780 ratings about 271,379 books, collected by Cai-Nicolas Ziegler from the Book-Crossing community. After implementing the system, customers will receive personalized plans with top recommended books.



Related Work

Applications of collaborative filtering typically involve very large data sets. Collaborative filtering methods have been applied to many different kinds of data including: sensing and monitoring data, such as in mineral exploration, environmental sensing over large areas or multiple sensors; financial data, such as financial service institutions that integrate many financial sources; or in electronic commerce and web applications where the focus is on user data, etc.

With a user-based approach or item-based approach, the system can predict the ratings of a person may give to an item by considering the similar persons or items.

Implementation

We use two collaborative filtering algorithms to recommend book list for customers. One is User-Based Collaborative Filtering and the other is Item-Based Collaborative Filtering.

Firstly, we read the .csv file into a Pandas DataFrame.
For the user dataset, we indexed each user with their ID. In this project, we are only interested in the ID of each user.

For the book dataset, we indexed each book with their ISBN. Same as the user dataset, we only need the ISBN, Title, Author, Year of publication and Publisher of each books. So, we keep these columns and drop other columns.

For the ratings dataset, it is the main dataset that we need to provide recommendation. However, by exploring this dataset, we found that there are lots of ratings represented by 0, and it also has some ISBN are not in the book dataset. These data can be seemed as dirty data. So, we moved these rows in this dataset.

The first 5 rows of each DataFrame are shown below:

	Location	Age
1	usa	21.0
2	usa	18.0
3	russia	21.0
4	portugal	17.0
5	united kingdom	21.0

User DataFrame

	Title	Author	Year of Publication	Publisher
0195153448	Classical Mythology	Mark P. O. Morford	2002	Oxford University Press
0002005018	Clara Callan	Richard Bruce Wright	2001	HarperFlamingo Canada
0060973129	Decision in Normandy	Carlo D'Este	1991	HarperPerennial
0374157065	Flu: The Story of the Great Influenza Pandemic...	Gina Bari Kolata	1999	Farrar Straus Giroux
0393045218	The Mummies of Urumchi	E. J. W. Barber	1999	W. W. Norton &amp

Book DataFrame

	User-ID	ISBN	Rating
1	276726	0155061224	5
3	276729	052165615X	3
4	276729	0521795028	6
8	276744	038550120X	7
16	276747	0060517794	9

Rating DataFrame

Next, we start to implement the two algorithms respectively.
For a Collaborative Filtering Based Recommendation System, firstly we need to generate a user-item rating matrix from the Rating DataFrame. We noticed that most of the values in the ratings matrix are NaN, like the picture shows below.

ISBN	0000913154	0001046438	000104687X	0001047213	0001047973	000104799X	0001048082	0001053736	0001053744	0001055607	..
userID											
2033	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	..
2110	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	..
2276	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	..
4017	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	..
4385	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	..

In order to let the algorithm to handle this data, we replace these values as 0. In addition, because of the extremely sparsity and large of this matrix, we need to reduce the size of it. In this program, we choose to consider the users who have rated at least 100 books and books which have at least 100 ratings.

Here is the matrix generated under such condition. The indices are ID of users, the columns are ISBN of books.

ISBN	0060392452	0060502258	0060915544	0060928336	0060930535	0060934417	0060938455	0060959037	0060976845	0060987103	...	0743418174	0786868
User-ID													
100459	9.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	
100906	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	
101209	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	
101606	0.0	0.0	0.0	8.0	0.0	0.0	0.0	0.0	6.0	0.0	...	0.0	
101851	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	

User-Based Collaborative Filtering
After generating the user-item rating matrix, we are going to implement UBCF algorithm. At first, we need to write a function to compute the similarity between two users. We choose two algorithms, one is Cosine similarity and the other is Pearson-correlation similarity.

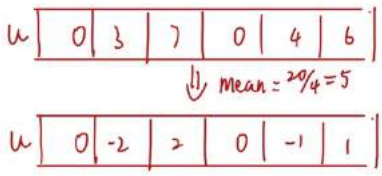
Simple introduction of these two methods:
Cosine similarity is defined as follow:

$$S_{j,u} = \cos(\vec{j}, \vec{u}) = \frac{\vec{j} \cdot \vec{u}}{\|\vec{j}\|_2 * \|\vec{u}\|_2}$$

Where " · " denotes the dot product between the two vectors, $\vec{j}$ and $\vec{u}$ and denotes the user's vector of user j and user u.
Pearson-correlation similarity is defined as follow:

$$S_{j,u} = \frac{\sum_{i=1}^{n}(j_i - \bar{j})(u_i - \bar{u})}{\sqrt{\sum_{i=1}^{n}(j_i - \bar{j})^2} * \sqrt{\sum_{i=1}^{n}(u_i - \bar{u})^2}}$$

Where $\bar{j}$ denotes the average rating in the vector of user j. This algorithm is also called Centred Cosine similarity, when we implemented this algorithm, firstly we convert each user vector into a vector centred on 0. The process can be described on the picture shows below:



Firstly, we get the mean value of the rating which is not 0, then we minus each of these value by the mean value. Therefore, we can use this vector and the function of Cosine similarity to calculate the similarity between two users.

In practise, firstly, we generate the vector of the user u who we decided to provide recommendation. Then we can compute the similarity between this user and other user in the user-item rating matrix. After that, we choose the top-k users who has also rating the item i as the top k most similar users of user u. Then we can run the algorithm below.

The principle of User-Based Collaborative Filtering is to predict the value of $R_{u,i}$ (which denotes the ratings of user u on item i) by computing the weighted sum of the top-k similar user's rating on the same item i. The function is:

$$R_{u,i} = \frac{\sum_{j=1}^{k} R_{j,i} * S_{j,u}}{\sum_{j=1}^{k} S_{j,u}}$$

Here $S_{j,u}$ denotes the similarity between user j and user u.
For example, suppose we got the matrix like this table. We choose to use the top-3 similar users to predict the value of $R_{2,3}$. Suppose the value of the similarity between user 2 and each user are in the rightest column, we can predict $R_{2,3}$ like it shows in this picture.

	0	1	2	3	similarity
0	3	7	4	9	0.2
1	7	0	5	3	0.3 ✓ top-3
2	7	5	5	0 4.17	1
3	5	6	8	5	0.5 ✓
4	5	8	8	8	0.1
5	7	7	0	4	0.4 ✓

$R_{2,3} = \frac{0.3\times3 + 0.5\times5 + 0.4\times4}{0.3+0.5+0.4} = 4.17$

Therefore, in order to provide recommendation for a user u, we run this algorithm through all the $R_{u,i}$ which value is 0 and predict the rating for it. After this process is done, we rank the top-10 $R_{u,i}$ and extract their related book's ISBN. Finally, we use these 10 ISBN to get the book's information from the book dataset.

Item-Based Collaborative Filtering

This method is a bit like the UBCF. In order to facilitate our implementation of this algorithm, we convert the user-item rating matrix into the item-user rating matrix by using matrix transpose. Here is the item-user rating matrix:

User-ID	100459	100906	101209	101606	101851	102647	102702	102967	104399	104636	...	94853	94951	95010	95359	95932	96448	97754	97874
ISBN																			
0060392452	9.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	10.0	0.0	0.0	0.0	10.0
0060502258	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	9.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0060915544	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	9.0	0.0	0.0	0.0	0.0
0060928336	0.0	0.0	0.0	8.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	8.0	0.0	0.0	0.0
0060930535	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	0.0	0.0	0.0	0.0	0.0
0060934417	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	9.0	0.0	0.0	0.0	0.0

Instead of focusing on friends on user u, in IBCF we focus on all the books rated by the user. Simple introduction of this algorithm:

The value of book i is rated by user u can be denoted as $R_{i,u}$. This algorithm calculated this value by considering all the books that are rated by u. It calculated the similarity between i and these books, find the top-k similar books with i and replace $R_{i,u}$ with the weighted average of ratings of the top-k books.

Suppose we would like to provide recommendation for user u. Firstly, we get the list of ISBNs rated by u as well as their related ratings. For example, for the user whose ID is "100459", the result shows as follow:

	0060392452	0142000205	0345339681	0375727345	0380789035	0385720106	0440211727	0446310786	0446364193
100459	9.0	7.0	8.0	8.0	9.0	8.0	10.0	9.0	8.0

The columns names are ISBN of books rated by this user.

Then we extract the list of ISBNs (represent as B) in this DataFrame and create a matrix combined with the vector of each books in this list. The matrix shows as follow:

User-ID	100459	100906	101209	101606	101851	102647	102702	102967	104399	104636	...
0060392452	9.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...
0142000205	7.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...
0345339681	8.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...
0375727345	8.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...
0380789035	9.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...
0385720106	8.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...
0440211727	10.0	0.0	0.0	0.0	9.0	10.0	0.0	0.0	0.0	10.0	...

Then, for each book which are not rated by the user u, we generated the book vector and find the top-k similar books from the list B. We can use cos similarity and Pearson correlation similarity as similarity function to find the top-k similar books. (described before)

After that, we can calculate the weighted sum of these top-k similar books as $R_{i,u}$.

$$R_{i,u} = \frac{\sum_{b=1}^{k} R_{b,u} * S_{b,i}}{\sum_{j=1}^{k} S_{b,i}}$$

$R_{b,u}$ denotes as the value of rating for book b rated by user u.
$S_{b,i}$ denotes as the value of similarity between book b and book i

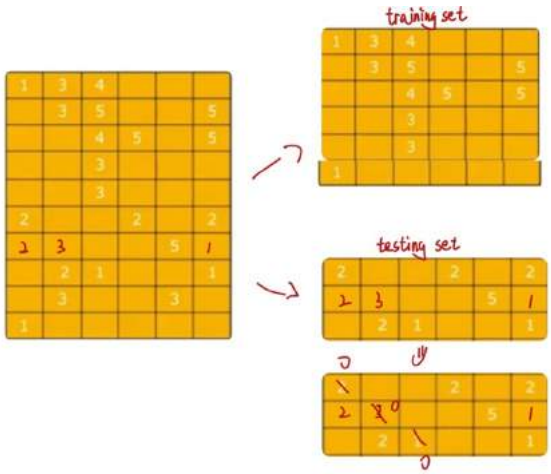
After going through each book i and predict the value of $R_{i,u}$, we rank the top-10 $R_{i,u}$ and extract their related book's ISBN and make a recommendation.

Results

We use the person whose ID is 6575 as a sample input to make a recommendation. Here we use 4 algorithms to make a recommendation separately. The results of these 4 algorithms are shown on the Appendix (f1, f2, f3, f4).

From these results, we found that the recommendation result of these 4 algorithms are quite different between each other. Based on looking at the running time, we found that IBCF would take more time than UBCF. One takes around 14s and the other takes around 2s. But we cannot find any useful information on the accuracy of these algorithms. Therefore, we implemented a measurement algorithm to calculate the accuracy of these algorithms.

We divided the user-item rating matrix into 2 parts, one is used for training and the other is used for testing. In practise, we randomly take 10% of the whole dataset as testing dataset. For each user in the testing dataset, we randomly change one no-zero rating in their vector to 0 and use the training dataset to predict the value of this cell. This process can be described by this picture:



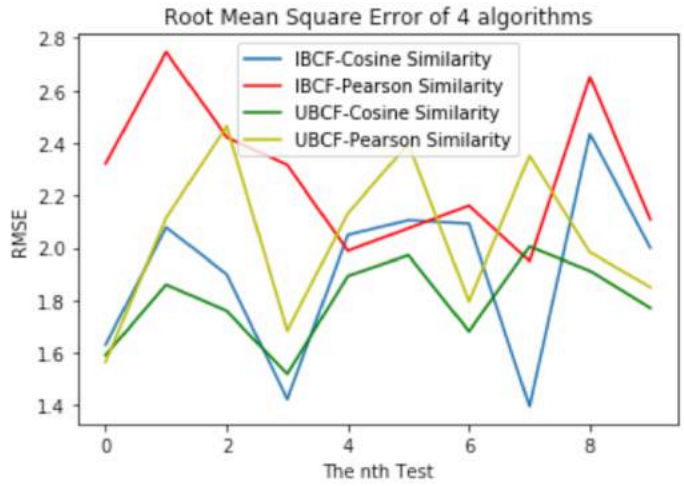
Then we calculate the difference between the real value y_i and the predicted value $h(x_i)$. After we go through the whole test dataset which has m users. We can calculate Root Mean Square Error (RMSE) by using the function:

$$RMSE(X, h) = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(h(x_i) - y_i)^2}$$

We decide to run this process using these 4 algorithms 10 times and calculated the average RMSE of each algorithm. Here is the result:

```
The average RMSE for IBCF(Cosine Similarity) is:  1.9105630380650216
The average RMSE for IBCF(Pearson Similarity) is:  2.2738431533942904
The average RMSE for UBCF(Cosine Similarity) is:  1.795989325972967
The average RMSE for UBCF(Pearson Similarity) is:  2.0331698445694113
Total time used is: 1071.06165599823s
```

We also provide a line graph and a table to describe the change of RMSE during the 10 times:



	0	1	2	3	4	5	6	7	8	9
IC	1.630340	2.078065	1.898707	1.422406	2.049424	2.105369	2.092736	1.395869	2.433741	1.998973
IP	2.320506	2.746608	2.421888	2.316137	1.988851	2.074422	2.161363	1.949467	2.650869	2.108321
UC	1.591020	1.859321	1.758902	1.518717	1.892265	1.971638	1.681229	2.004781	1.912020	1.770000
UP	1.564611	2.114413	2.464116	1.683320	2.131351	2.395871	1.794736	2.351022	1.982741	1.849519

From this test, we deduce that for this book data, User-Based Collaborative Filtering performs better than Item-Based Collaborative Filtering. What is more, Cosine Similarity has a better performance than Pearson Similarity on this dataset.

Discussion

As can be seen from the results in the RMSE of 4 algorithms graph, the results produced by Cosine Similarity outperform the Pearson Similarity. This is because RMSE is an algorithm which would penalize large error, it's is easy for Pearson Similarity to produce larger error than Cosine Similarity in this sparse dataset. Since when predicting, we only extract the top-k similar users with conditions of rating the specific book before, Pearson Similarity may choose some similar users with negative correlation, this would result in calculating some abnormal value, and finally penalty by the RMSE.

In addition, User-Based Collaborative Filtering has a better result than Item-Based Collaborative Filtering. As mentioned before, the dataset is sparse, we calculated that 97% cells in both the user-user matrix and item-item matrix are NaN values. This may result in

which algorithm has more rows of data, it would be more likely to find high similarity instance and result in a better-quality performance. After filtering, we have 891 users and 445 books. So, in the extremely sparse condition, User-Based Collaborative Filtering has more instance to compare the similarity than Item-Based Collaborative Filtering.

Conclusion

The huge volume of information flowing on the web has given rise to the need for information filtering techniques. Recommender systems are effectively used to filter out excess information and to provide personalized services to users by employing sophisticated, well thought-out prediction algorithms.

In this assignment, we explored two algorithm which are commonly used and compared their performance on the books rating dataset. However, there are a lot of algorithms that can be used in the recommendation system waiting for us to study. Through this assignment, we understand that each data set has the most suitable algorithm for it, and we need to find them through a lot of tests.

Future Work

This search has implemented two basic types of filtering algorithm with two similarity computation algorithms. We found that the result of recommendation is not good because of the extremely sparsity of this dataset. The next step we are going to do is expanding the data set with more accurate and useful book information instead of the current redundant data set, eventually reduces data sparsity.

The recommender system can be used in many areas, for examples, implements on a book selling or book reading website, the system should simultaneously generate data by user operation. If a user clicks or reads or buys (is interested in) two books of the personalized recommendation book list, the system should re-evaluate the user's preference and provide a new book list based on real time data instead of the fixed data set.

References

Kim Falk, Jan 2019, "Practical Recommender Systems", ISBN 9781617292705

Reena Pagare, Anita Shinde, Jun 2012, "International Journal of Computer Applications", A Study of Recommender System Techniques, vol. 47– No.16

Bin Wang, May 2018, "Comparison of User-Based and Item-Based Collaborative Filtering"

Chhavi Saluja, Mar 2018, "My Journey to building Book Recommendation System"

Nadia F. Al-Bakri, Soukaena Hassan Hashim, Jun 2018 , "Reducing Data Sparsity in Recommender Systems", Journal of Al-Nahrain University, vol.21 (2), pp.138-147

Haifeng Liu, Zheng Hu, Ahmad Mian, Hui Tian, Xuzhen Zhu, Jan 2014, “Knowledge-Based Systems”, A new user similarity model to improve the accuracy of collaborative filtering vol. 56, pages 156-166

Appendix

UBCF with Pearson Correlation Similarity (f1):

Result of using UBCF with Pearson Correlation Similarity

```
stime = time.time()
ubMatrix = generateUserBooksMatrix(bookRatings,50,50)
ubMatrix = ubMatrix[ubMatrix.nunique(axis=1)!=2]
outbooks = user_basedCF("6575","correlation",10,ubMatrix)
print(f"It Takes {time.time()-stime}s")
books.loc[outbooks]

```

```
It Takes 2.573383331298828s
```

	Title	Author	Year of Publication	Publisher
0446679593	Suzanne's Diary for Nicholas	James Patterson	2002	Warner Books
0451203771	Scarlet Feather	Maeve Binchy	2002	Signet Book
0515135062	Three Fates	Nora Roberts	2004	Jove Books
0553574574	Beach Music	Pat Conroy	1996	Bantam Books
0743203631	Gap Creek: The Story Of A Marriage	Robert Morgan	2000	Touchstone
0345339738	The Return of the King (The Lord of the Rings,...	J.R.R. TOLKIEN	1986	Del Rey
080213825X	Four Blondes	Candace Bushnell	2001	Grove Press
0553274295	Where the Red Fern Grows	Wilson Rawls	1984	Random House Children's Books
0345339711	The Two Towers (The Lord of the Rings, Part 2)	J.R.R. TOLKIEN	1986	Del Rey
0440414806	Holes (Yearling Newbery)	LOUIS SACHAR	2000	Yearling

UBCF with Cosine Similarity (f2):

Result of using UBCF with Cosine Similarity

```
stime = time.time()
ubMatrix = generateUserBooksMatrix(bookRatings,50,50)
ubMatrix = ubMatrix[ubMatrix.nunique(axis=1)!=2]
outbooks = user_basedCF("6575","cos",10,ubMatrix)
print(f"It Takes {time.time()-stime}s")
books.loc[outbooks]
```

```
It Takes 2.404301881790161s
```

	Title	Author	Year of Publication	Publisher
059035342X	Harry Potter and the Sorcerer's Stone (Harry P...	J. K. Rowling	1999	Arthur A. Levine Books
0446310786	To Kill a Mockingbird	Harper Lee	1988	Little Brown &amp
0553274295	Where the Red Fern Grows	Wilson Rawls	1984	Random House Children's Books
0142004235	East of Eden (Oprah's Book Club)	John Steinbeck	2003	Penguin Books
0345339738	The Return of the King (The Lord of the Rings,...	J.R.R. TOLKIEN	1986	Del Rey
0345361792	A Prayer for Owen Meany	John Irving	1990	Ballantine Books
0064400557	Charlotte's Web (Trophy Newbery)	E. B. White	1974	HarperTrophy
0440998050	A Wrinkle in Time	Madeleine L'Engle	1976	Laure Leaf
0439064864	Harry Potter and the Chamber of Secrets (Book 2)	J. K. Rowling	1999	Scholastic
0345348036	The Princess Bride: S Morgenstern's Classic Ta...	WILLIAM GOLDMAN	1987	Del Rey

IBCF with Pearson Correlation Similarity (f3):

Result of using IBCF with Pearson Correlation Similarity

```
stime = time.time()
ibMatrix = generateUserBooksMatrix(bookRatings,50,50).T
ibMatrix = ibMatrix[ibMatrix.nunique(axis=1)!=2]
outputdic = item_basedCF("6575","correlation",10,ibMatrix)
outbooks = sorted(outputdic, key=lambda x: outputdic[x], reverse=True)[:10]
print(f"It Takes {time.time()-stime}s")
books.loc[outbooks]

```

```
It Takes 14.056365013122559s
```

	Title	Author	Year of Publication	Publisher
0312306326	Visions of Sugar Plums	Janet Evanovich	2002	St. Martin's Press
0399150897	Blow Fly: A Scarpetta Novel	Patricia Cornwell	2003	Putnam Publishing Group
0312966091	Three To Get Deadly : A Stephanie Plum Novel (...	Janet Evanovich	1998	St. Martin's Paperbacks
0060932759	Daughter of Fortune	Isabel Allende	2000	Perennial
0312971346	High Five (A Stephanie Plum Novel)	Janet Evanovich	2000	St. Martin's Paperbacks
0345339711	The Two Towers (The Lord of the Rings, Part 2)	J.R.R. TOLKIEN	1986	Del Rey
0312421273	The Corrections: A Novel	Jonathan Franzen	2002	Picador USA
080411868X	Welcome to the World, Baby Girl!	Fannie Flagg	1999	Ballantine Books
0312976275	Hot Six : A Stephanie Plum Novel (A Stephanie ...	Janet Evanovich	2001	St. Martin's Paperbacks
0446612790	2nd Chance	James Patterson	2003	Warner Vision

IBCF with Cosine Similarity (f4):

Result of using IBCF with Cosine Similarity

```
stime = time.time()
ibMatrix = generateUserBooksMatrix(bookRatings,50,50).T
ibMatrix = ibMatrix[ibMatrix.nunique(axis=1)!=2]
outputdic = item_basedCF("6575","cos",10,ibMatrix)
outbooks = sorted(outputdic, key=lambda x: outputdic[x], reverse=True)[:10]
books.loc[outbooks]
print(f"It Takes {time.time()-stime}s")
books.loc[outbooks]
```

```
It Takes 14.169964075088501s
```

	Title	Author	Year of Publication	Publisher
0312983867	Hard Eight : A Stephanie Plum Novel (A Stephan...	Janet Evanovich	2003	St. Martin's Paperbacks
0312971346	High Five (A Stephanie Plum Novel)	Janet Evanovich	2000	St. Martin's Paperbacks
0515135062	Three Fates	Nora Roberts	2004	Jove Books
0312983271	Full House (Janet Evanovich's Full Series)	Janet Evanovich	2002	St. Martin's Paperbacks
0312966091	Three To Get Deadly : A Stephanie Plum Novel (...	Janet Evanovich	1998	St. Martin's Paperbacks
0446612790	2nd Chance	James Patterson	2003	Warner Vision
0312976275	Hot Six : A Stephanie Plum Novel (A Stephanie ...	Janet Evanovich	2001	St. Martin's Paperbacks
0446605484	Roses Are Red (Alex Cross Novels)	James Patterson	2001	Warner Vision
0515132187	The Villa	Nora Roberts	2002	Jove Books
0451167317	The Dark Half	Stephen King	1994	Signet Book