# Capstone Project - 4

# Book Recommendation System

**Team Members**

Bindu Kovvada

Manoj Patil M

Gulzar

Saksham Tripathi

Deepak Kumar Gautam

# Table of content

- Importing the Relevant Libraries and Data Inspection
- Exploratory Data Analysis (Dataset-wise)
- Recommendation for New Users(Cold Start)
- Model Based Collaborative Filtering Recommender
- Memory Based Collaborative Filtering Recommender
  - Item-Item Based
  - Item-Item Based- Second Method
  - User-Item Based
- Content Based Filtering Recommender
  - On the basis of Title
  - On the basis of Author
  - On the basis of Search-Keyword
  - On the basis of Purchase-History
- Conclusion

# Problem Statement

During the last few decades, with the rise of Youtube, Amazon, Netflix, and many other such web services, recommender systems have taken more and more place in our lives. From e-commerce (suggest to buyers articles that could interest them) to online advertisement (suggest to users the right contents, matching their preferences), recommender systems are today unavoidable in our daily online journeys.

In a very general way, recommender systems are algorithms aimed at suggesting relevant items to users (items being movies to watch, text to read, products to buy, or anything else depending on industries). Recommender systems are really critical in some industries as they can generate a huge amount of income when they are efficient or also be a way to stand out significantly from competitors. The main objective is to create a book recommendation system for users.

# Data set information

The dataset is comprised of three csv files:: Users,   Books,  Ratings

**Users_dataset.**

- User-ID (unique for each user)
- Location (contains city, state and country separated by commas)
- Age
- Shape of Dataset - (278858, 3)

**Books_dataset.**

- ISBN (unique for each book)
- Book-Title
- Book-Author
- Year-Of-Publication
- Publisher
- Image-URL-S
- Image-URL-M
- Image-URL-L
-
Shape of Dataset - (271360, 8)

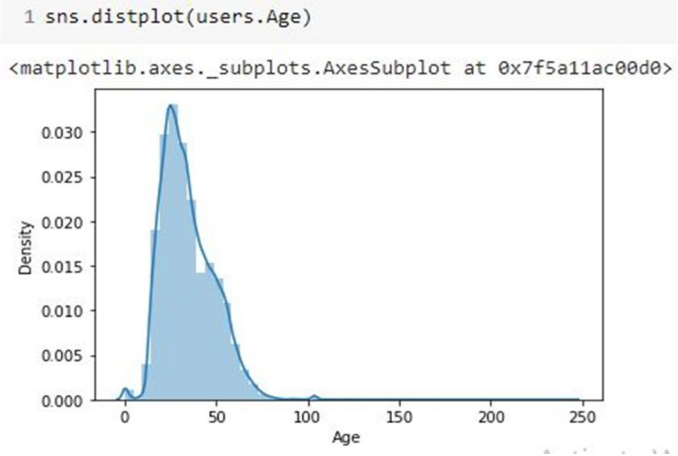
**Ratings_dataset.**

- User-ID
- Shape of Dataset - (1149780, 3)
- Book-Rating
- ISBN

# Exploratory Data Analysis (User Dataset)

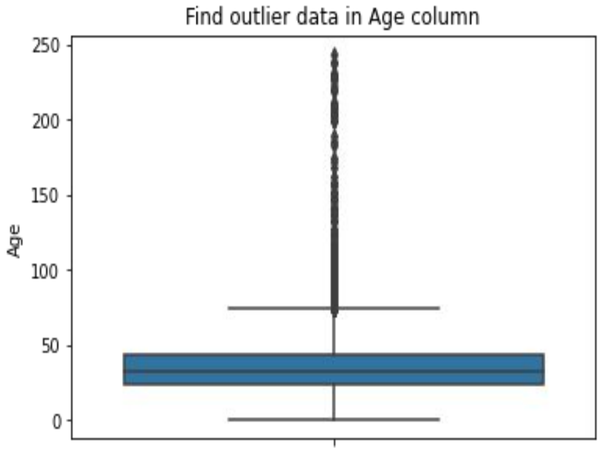
## Checking distribution of Age feature:

- Age in the dataset ranges from 0 To 250.
- Most of the users are of age 20-40 years.
- The Age range  distribution is right skewed
- Outliers are present in the Age column.



```
sns.distplot(users.Age)
```

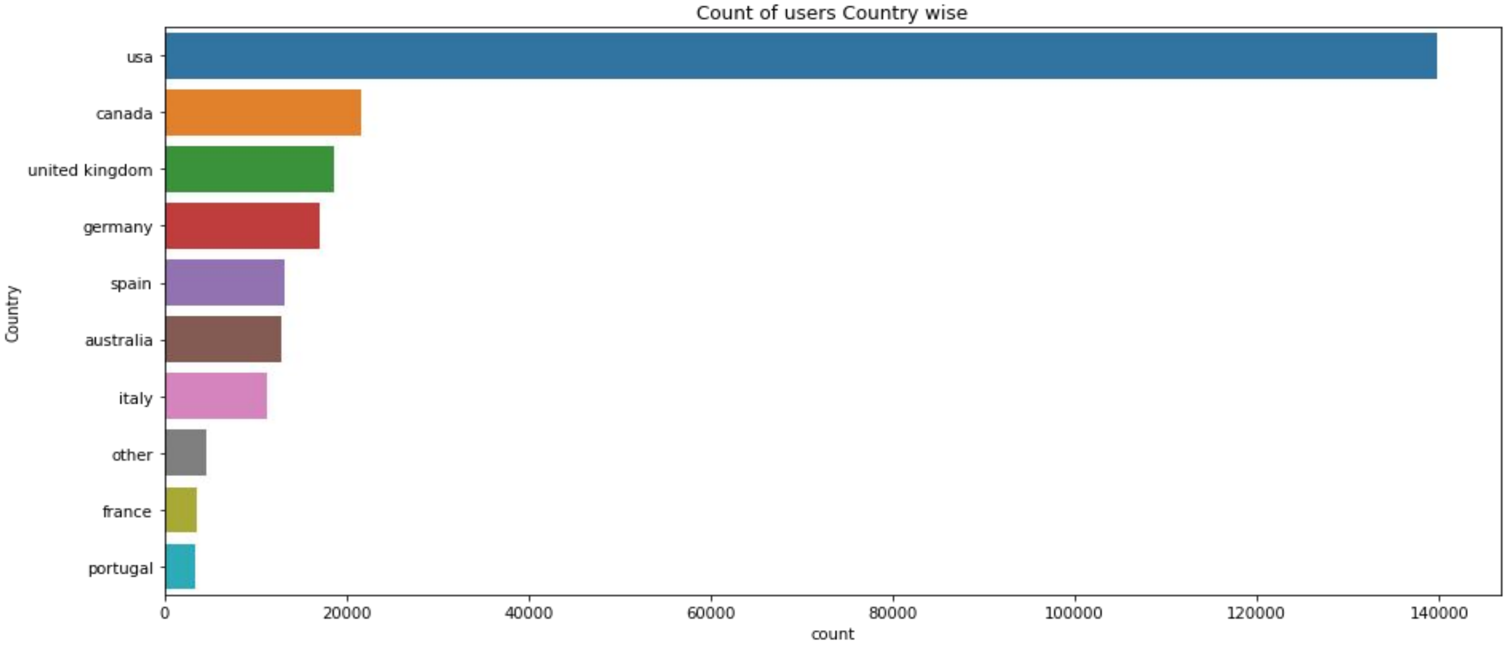
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5a11ac00d0>
```



Find outlier data in Age column

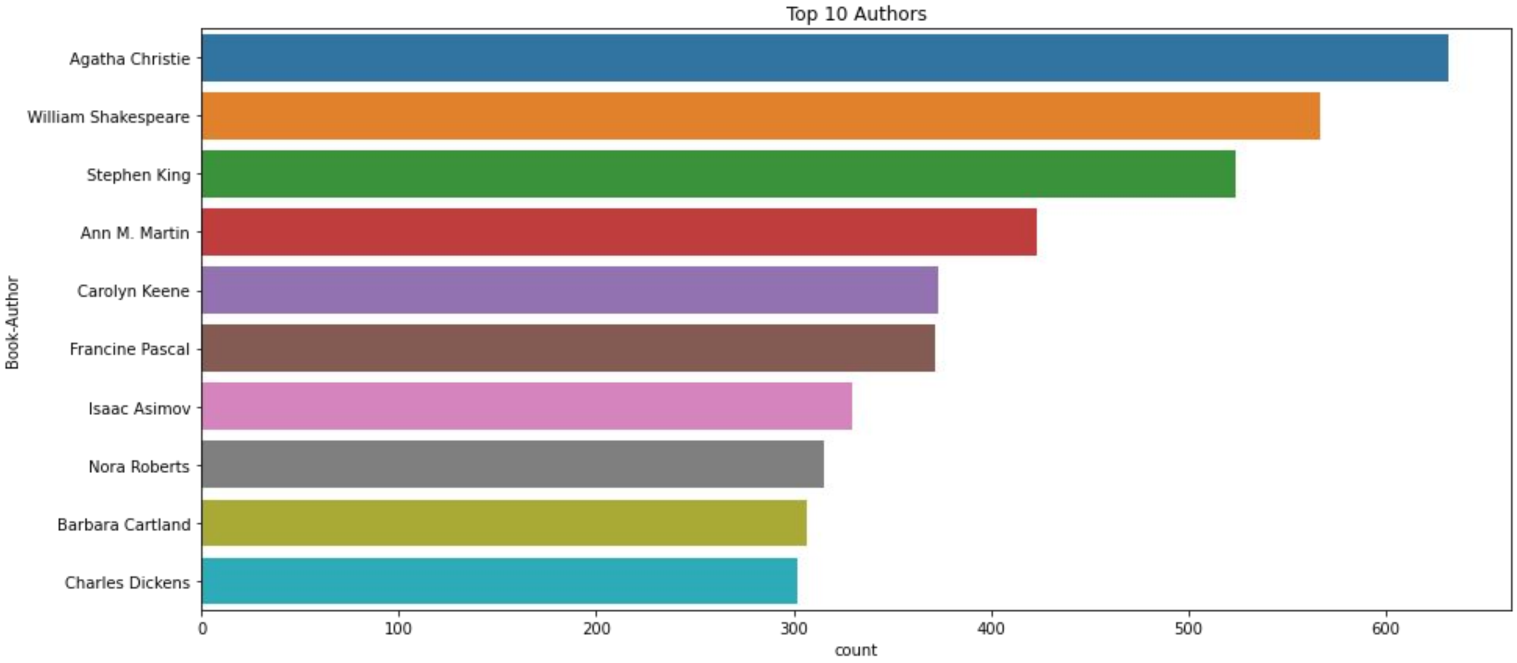
# Checking distribution of Location feature:

- Most active readers are from USA.



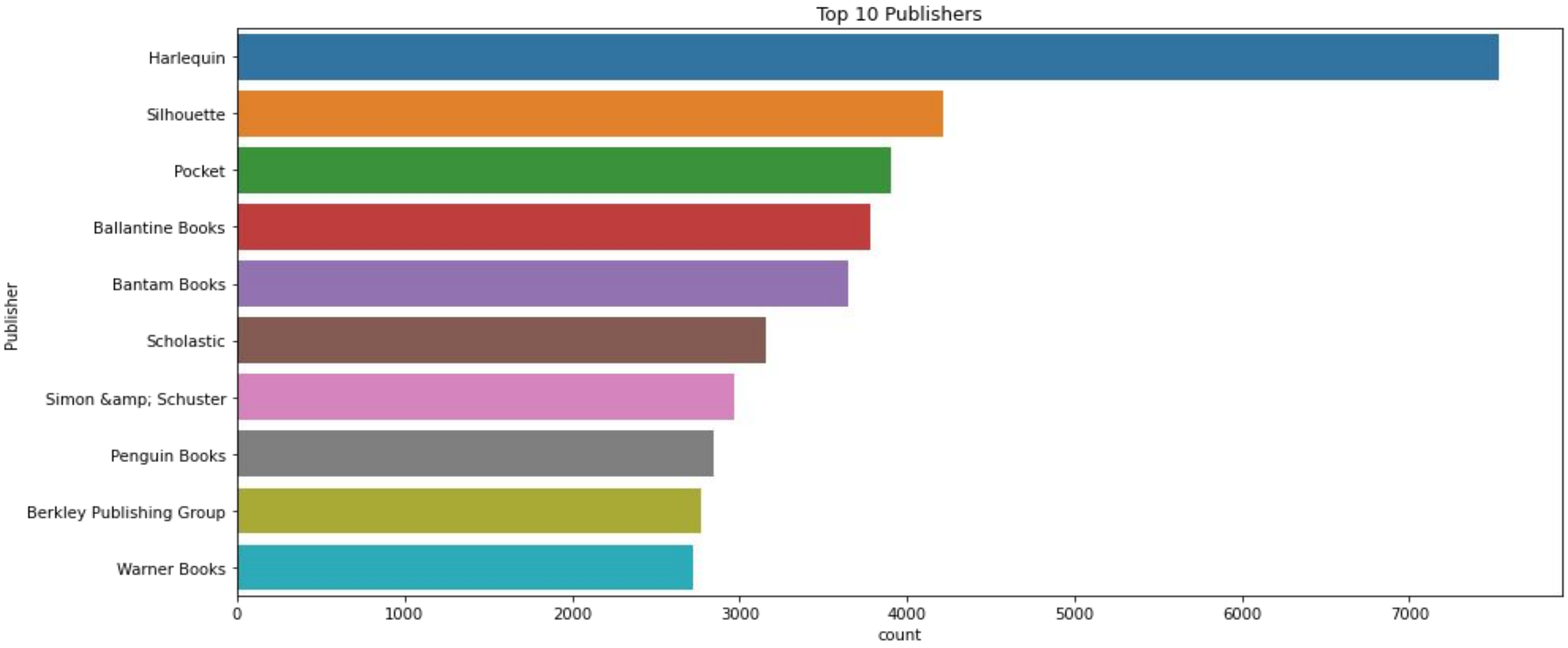
# Exploratory Data Analysis (Books Dataset)

Top 10 Authors which have written the most books :



Agatha Christie wrote highest number of books in our given dataset

# Top 10 Publisher which have published the most books :



Harlequin published highest number of books in our given dataset.

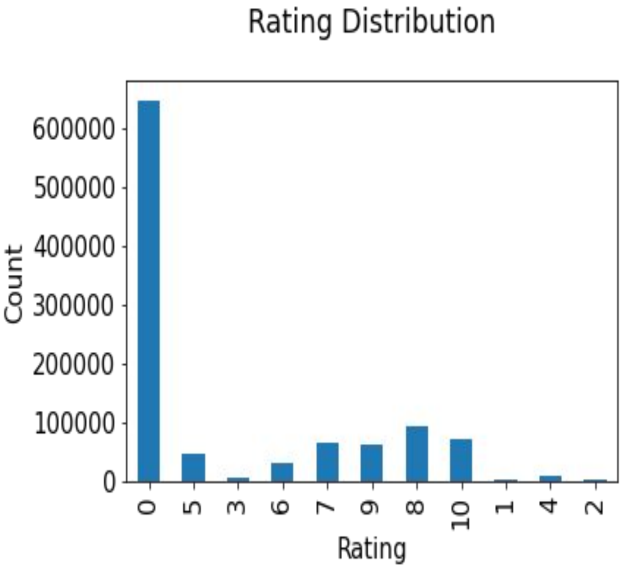
# Exploratory Data Analysis (Ratings Dataset)

As we can see from this bar graph, the ratings are very unevenly distributed, and the vast majority of ratings are 0 .

Book-Ratings Dataset contains the book rating information.

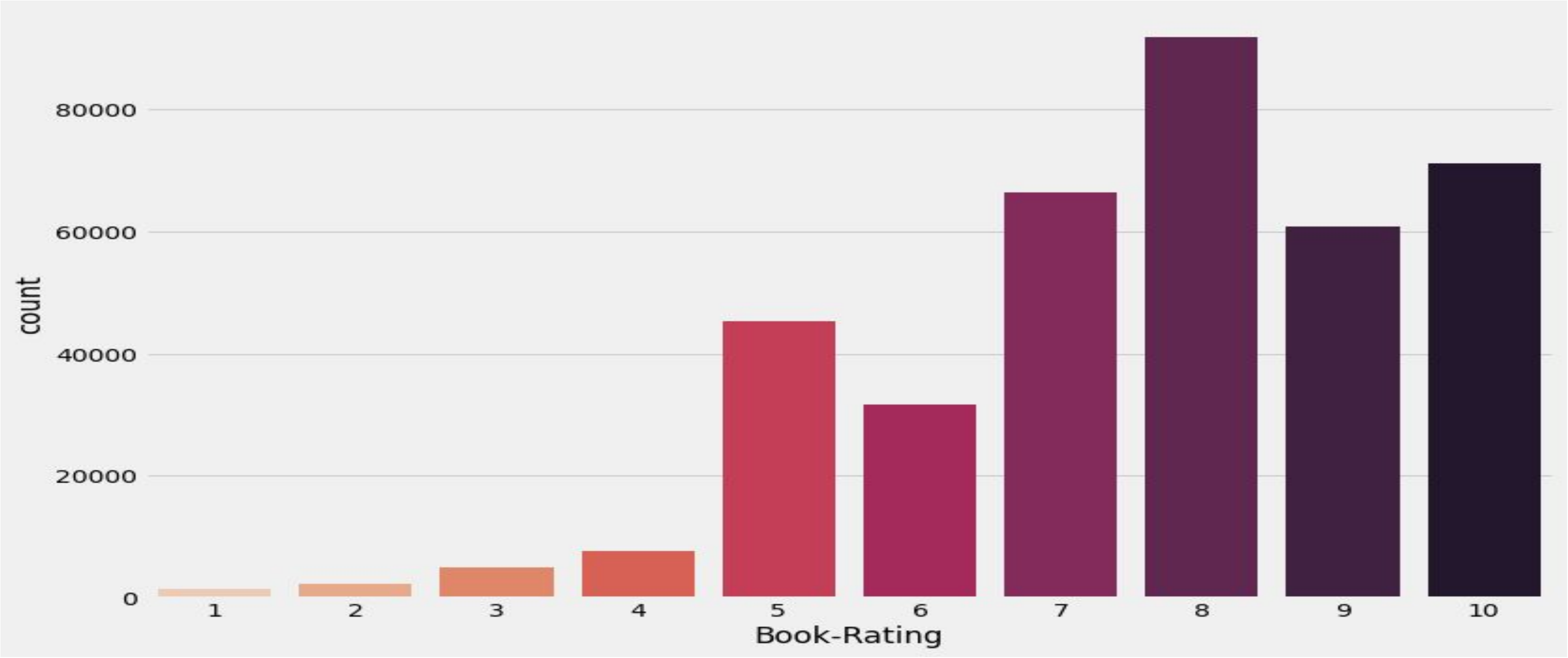
Ratings are either explicit, expressed on a scale from 1-10 higher values denoting higher appreciation, or implicit, expressed by 0.

Hence segregating implicit and explicit ratings datasets.

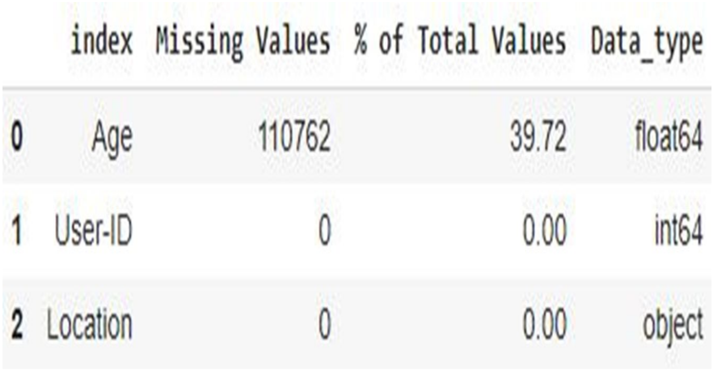


# Visualization Continue....

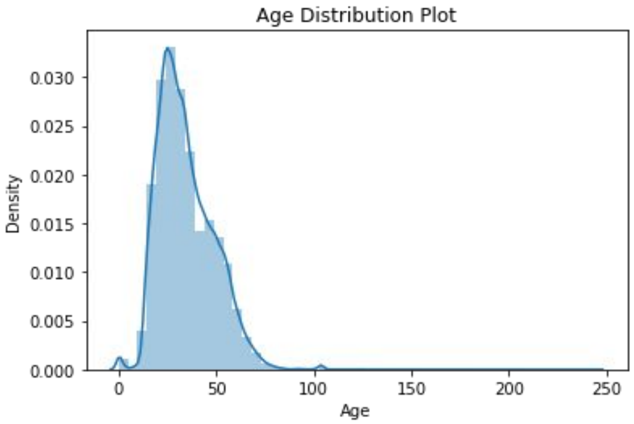
- Higher ratings are more common amongst users
- Rating 8 has been rated the highest number of times



# Data Cleaning



|   | index | Missing Values | % of Total Values | Data_type |
|---|---|---|---|---|
| 0 | Age | 110762 | 39.72 | float64 |
| 1 | User-ID | 0 | 0.00 | int64 |
| 2 | Location | 0 | 0.00 | object |



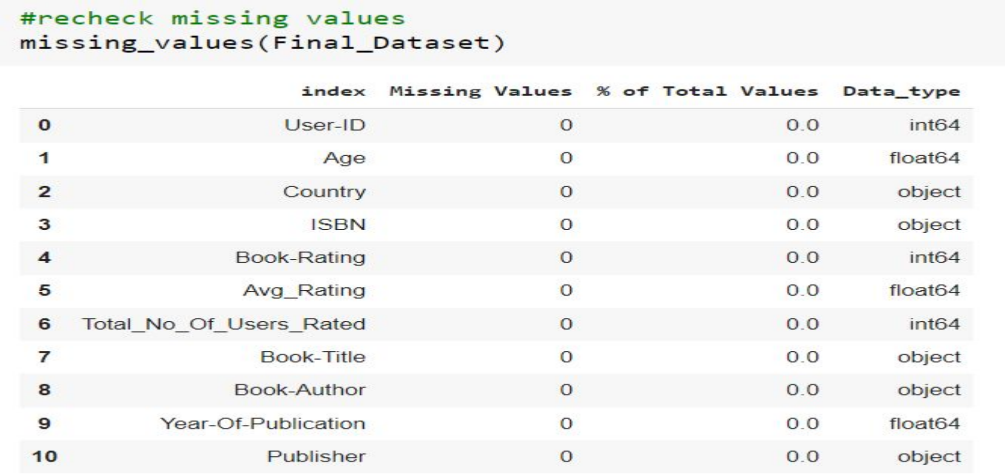
- Age column has 40% missing values.
- Age has positive Skewness (right tail) so we can use median to fill Nan values, but for this we don't like to fill Nan value just for one range of age. To handle this we'll use country column to fill Nan.
- As we all knew already that Age value's below 5 and above 100 do not make much sense as the can't read/rated our book so we can replace that.

# Merging All the three Datasets

Merging all the three datasets i.e Books, Users, Ratings dataset.

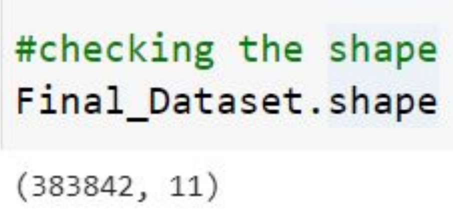
Rechecking Missing Values in the final dataset.

Checking Shape of the final dataset.



```
#recheck missing values
missing_values(Final_Dataset)
```

| | index | Missing Values | % of Total Values | Data_type |
|---|---|---|---|---|
| 0 | User-ID | 0 | 0.0 | int64 |
| 1 | Age | 0 | 0.0 | float64 |
| 2 | Country | 0 | 0.0 | object |
| 3 | ISBN | 0 | 0.0 | object |
| 4 | Book-Rating | 0 | 0.0 | int64 |
| 5 | Avg_Rating | 0 | 0.0 | float64 |
| 6 | Total_No_Of_Users_Rated | 0 | 0.0 | int64 |
| 7 | Book-Title | 0 | 0.0 | object |
| 8 | Book-Author | 0 | 0.0 | object |
| 9 | Year-Of-Publication | 0 | 0.0 | float64 |
| 10 | Publisher | 0 | 0.0 | object |



```
#checking the shape
Final_Dataset.shape
```

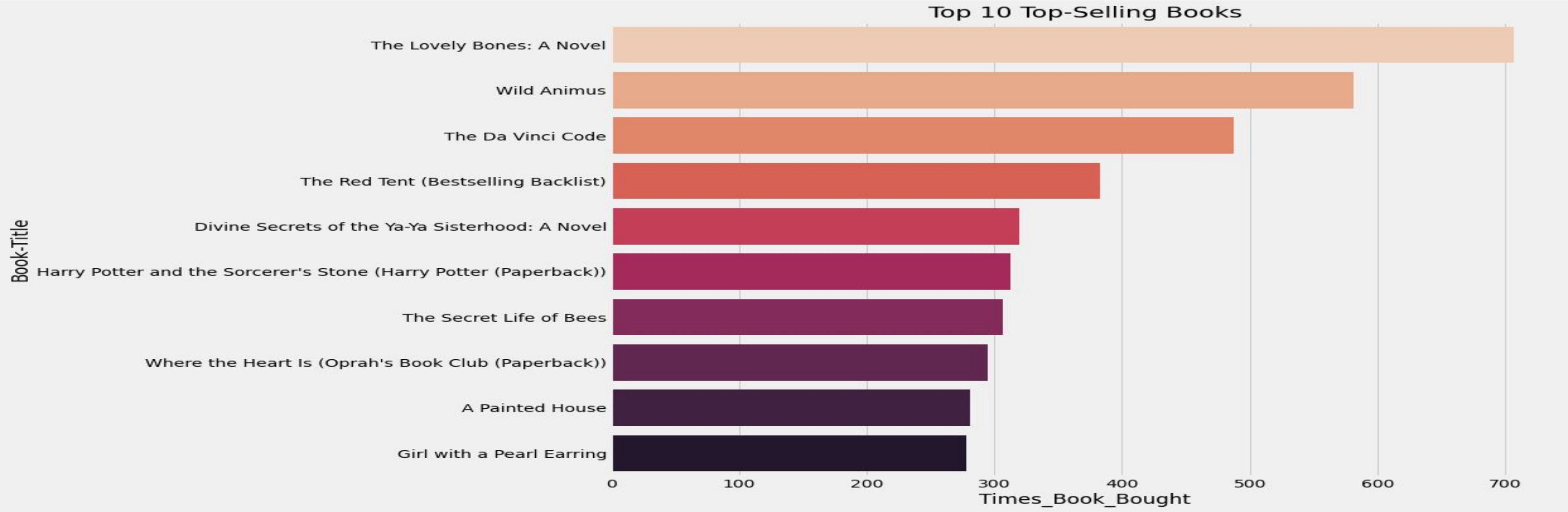
```
(383842, 11)
```

# Different Models

## 1:- Recommendation for New Users(Cold Start)

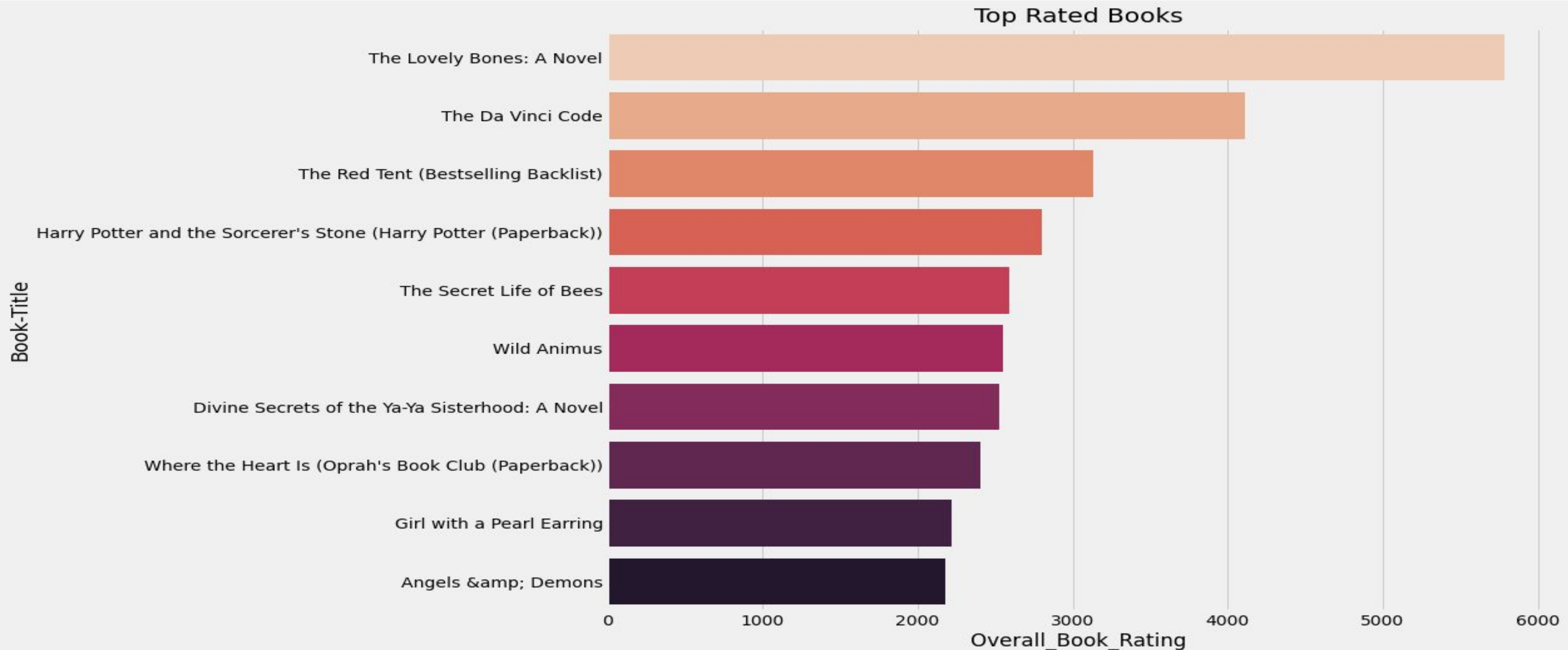
As we all know that collaborative filtering have cold start problem so it can't recommend books for fresh new user. So we can recommend them our top read/rated books as a new user.

- Top Selling Books



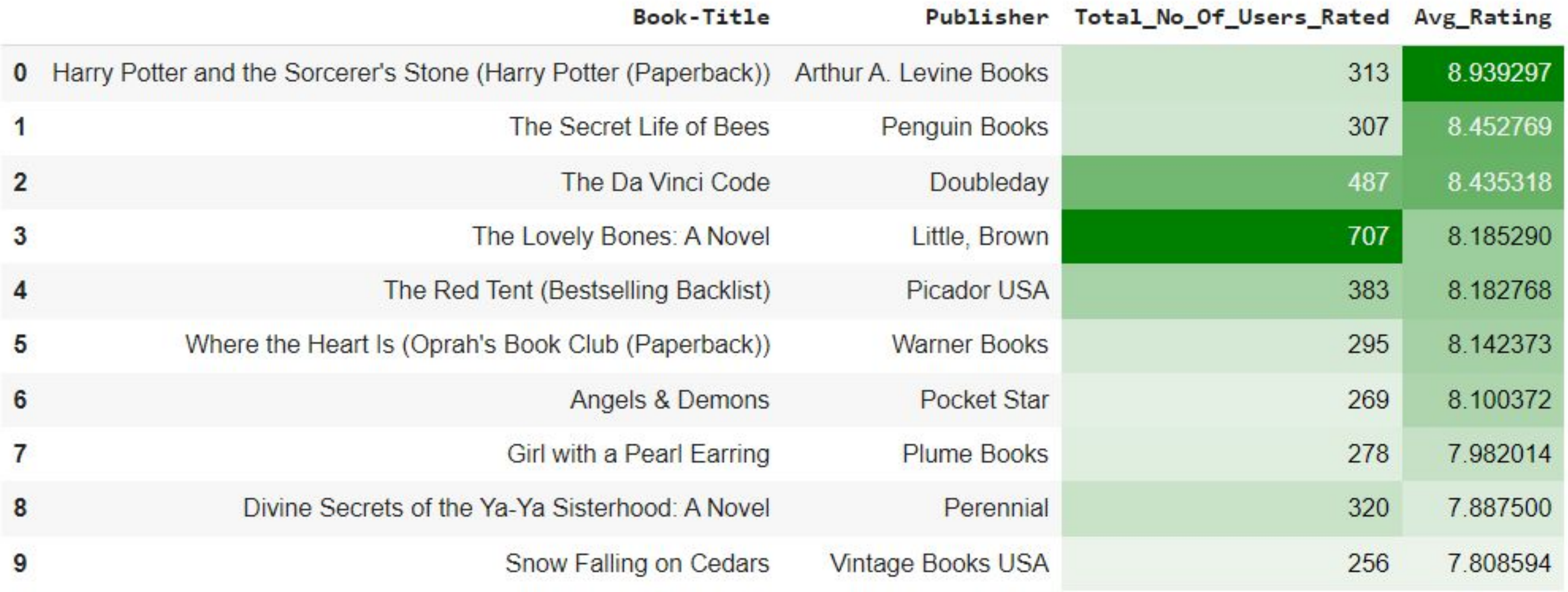
# 1:- Recommendation for New Users(Cold Start)

- Top Rated Books



# 1:- Recommendation for New Users(Cold Start)

- Top Rated & Sellings Books



| | Book-Title | Publisher | Total_No_Of_Users_Rated | Avg_Rating |
|---|---|---|---|---|
| 0 | Harry Potter and the Sorcerer's Stone (Harry Potter (Paperback)) | Arthur A. Levine Books | 313 | 8.939297 |
| 1 | The Secret Life of Bees | Penguin Books | 307 | 8.452769 |
| 2 | The Da Vinci Code | Doubleday | 487 | 8.435318 |
| 3 | The Lovely Bones: A Novel | Little, Brown | 707 | 8.185290 |
| 4 | The Red Tent (Bestselling Backlist) | Picador USA | 383 | 8.182768 |
| 5 | Where the Heart Is (Oprah's Book Club (Paperback)) | Warner Books | 295 | 8.142373 |
| 6 | Angels & Demons | Pocket Star | 269 | 8.100372 |
| 7 | Girl with a Pearl Earring | Plume Books | 278 | 7.982014 |
| 8 | Divine Secrets of the Ya-Ya Sisterhood: A Novel | Perennial | 320 | 7.887500 |
| 9 | Snow Falling on Cedars | Vintage Books USA | 256 | 7.808594 |

## 1:- Recommendation for New Users(Cold Start)

- Recommendation on the basis of Weighted Average(Popularity based Recommendation)

Book weighted average formula:

$$\textbf{Weighted Rating(WR)}=[vR/(v+m)]+[mC/(v+m)]$$

Where,

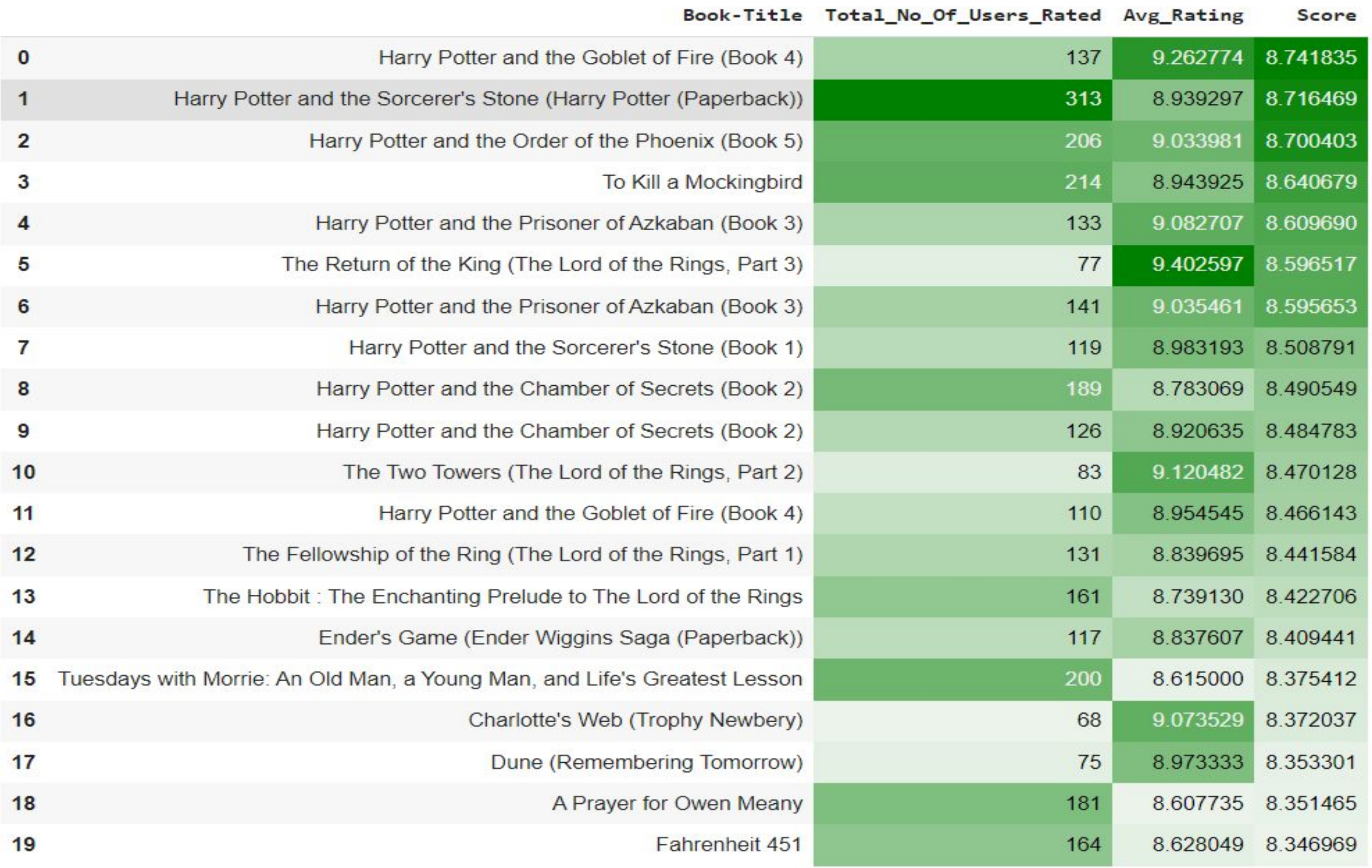
v is the number of votes for the books;

m is the minimum votes required to be listed in the chart;

R is the average rating of the book; and

C is the mean vote across the whole report.

# These are our top books on the basis of formula base-weighted ratings.

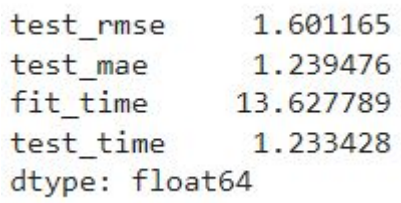


|  | Book-Title | Total_No_Of_Users_Rated | Avg_Rating | Score |
|---|---|---|---|---|
| 0 | Harry Potter and the Goblet of Fire (Book 4) | 137 | 9.262774 | 8.741835 |
| 1 | Harry Potter and the Sorcerer's Stone (Harry Potter (Paperback)) | 313 | 8.939297 | 8.716469 |
| 2 | Harry Potter and the Order of the Phoenix (Book 5) | 206 | 9.033981 | 8.700403 |
| 3 | To Kill a Mockingbird | 214 | 8.943925 | 8.640679 |
| 4 | Harry Potter and the Prisoner of Azkaban (Book 3) | 133 | 9.082707 | 8.609690 |
| 5 | The Return of the King (The Lord of the Rings, Part 3) | 77 | 9.402597 | 8.596517 |
| 6 | Harry Potter and the Prisoner of Azkaban (Book 3) | 141 | 9.035461 | 8.595653 |
| 7 | Harry Potter and the Sorcerer's Stone (Book 1) | 119 | 8.983193 | 8.508791 |
| 8 | Harry Potter and the Chamber of Secrets (Book 2) | 189 | 8.783069 | 8.490549 |
| 9 | Harry Potter and the Chamber of Secrets (Book 2) | 126 | 8.920635 | 8.484783 |
| 10 | The Two Towers (The Lord of the Rings, Part 2) | 83 | 9.120482 | 8.470128 |
| 11 | Harry Potter and the Goblet of Fire (Book 4) | 110 | 8.954545 | 8.466143 |
| 12 | The Fellowship of the Ring (The Lord of the Rings, Part 1) | 131 | 8.839695 | 8.441584 |
| 13 | The Hobbit : The Enchanting Prelude to The Lord of the Rings | 161 | 8.739130 | 8.422706 |
| 14 | Ender's Game (Ender Wiggins Saga (Paperback)) | 117 | 8.837607 | 8.409441 |
| 15 | Tuesdays with Morrie: An Old Man, a Young Man, and Life's Greatest Lesson | 200 | 8.615000 | 8.375412 |
| 16 | Charlotte's Web (Trophy Newbery) | 68 | 9.073529 | 8.372037 |
| 17 | Dune (Remembering Tomorrow) | 75 | 8.973333 | 8.353301 |
| 18 | A Prayer for Owen Meany | 181 | 8.607735 | 8.351465 |
| 19 | Fahrenheit 451 | 164 | 8.628049 | 8.346969 |

# 2:- Model Based Collaborative Filtering Recommender

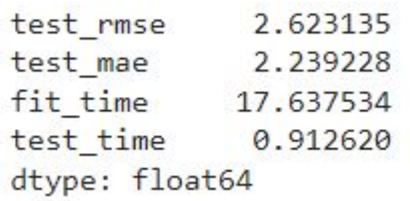
- The goal of the recommender system is to predict user preference for a set of items based on the past experience
- Collaborative filtering is a technique used by websites like Amazon, YouTube, and Netflix. It filters out items that a user might like on the basis of reactions of similar users.
- Model based approach involves building machine learning algorithms to predict user's ratings
- Singular Value Decomposition (SVD) and Non-negative Matrix Factorization (NMF) are matrix factorization techniques used for dimensionality reduction. Surprise package provides implementation of those algorithms.

SVD



```
test_rmse      1.601165
test_mae       1.239476
fit_time      13.627789
test_time      1.233428
dtype: float64
```

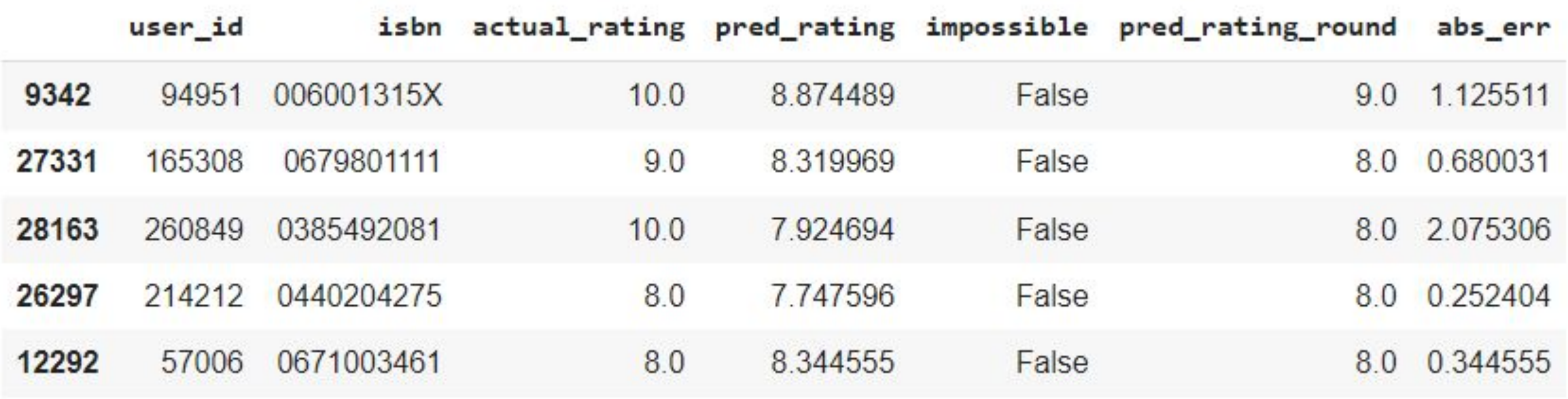
NMF



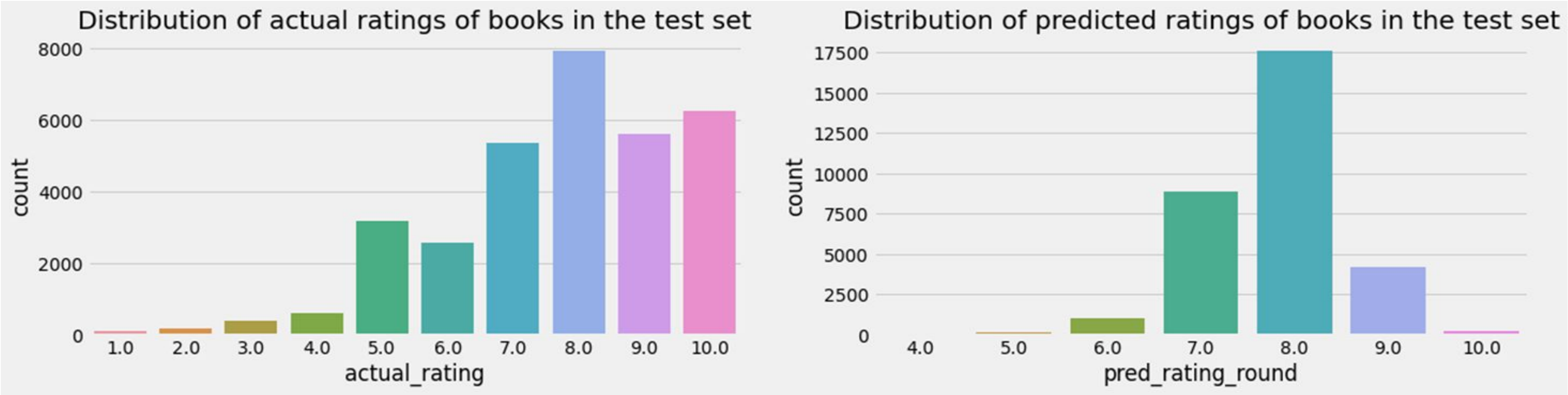
```
test_rmse      2.623135
test_mae       2.239228
fit_time      17.637534
test_time      0.912620
dtype: float64
```

It's clear that for the given dataset much better results can be obtained with SVD approach - both in terms of accuracy and training / testing time.

# SVD Model Results :



| | user_id | isbn | actual_rating | pred_rating | impossible | pred_rating_round | abs_err |
|---|---|---|---|---|---|---|---|
| **9342** | 94951 | 006001315X | 10.0 | 8.874489 | False | 9.0 | 1.125511 |
| **27331** | 165308 | 0679801111 | 9.0 | 8.319969 | False | 8.0 | 0.680031 |
| **28163** | 260849 | 0385492081 | 10.0 | 7.924694 | False | 8.0 | 2.075306 |
| **26297** | 214212 | 0440204275 | 8.0 | 7.747596 | False | 8.0 | 0.252404 |
| **12292** | 57006 | 0671003461 | 8.0 | 8.344555 | False | 8.0 | 0.344555 |



Distribution of actual ratings of books in the test set

Distribution of predicted ratings of books in the test set

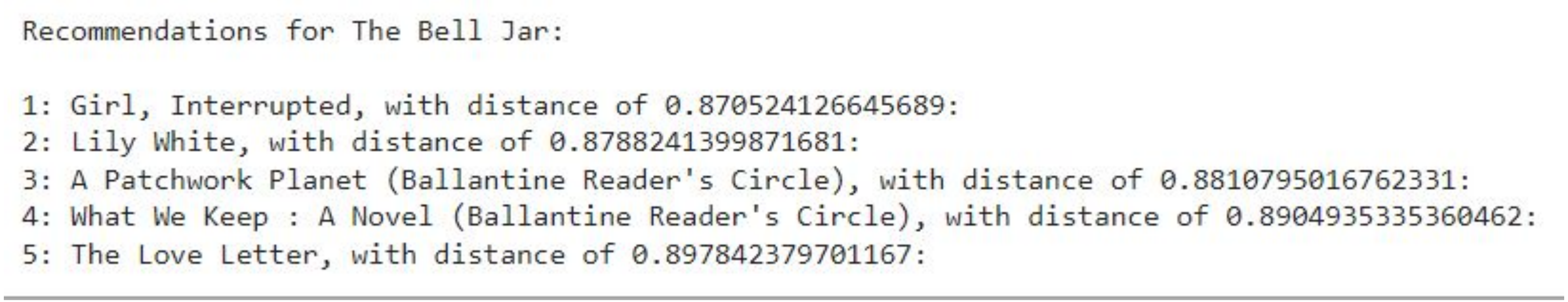
# Observations

- According to the distribution of actual ratings of books in the test set, the biggest part of users give positive scores - between 7 and 10.
- The mode equals 8 but count of ratings 7, 9, 10 is also noticeable.
- The distribution of predicted ratings in the test set is visibly different.
- It shows that the recommender system is not perfect and it cannot reflect the real distribution of book ratings.

# 3:- Memory Based Collaborative Filtering Recommender

- ## Collaborative Filtering (Item-Item based)

A KNN model, with cosine similarity as a metric for measuring the distance between different ratings, was used to provide recommendations



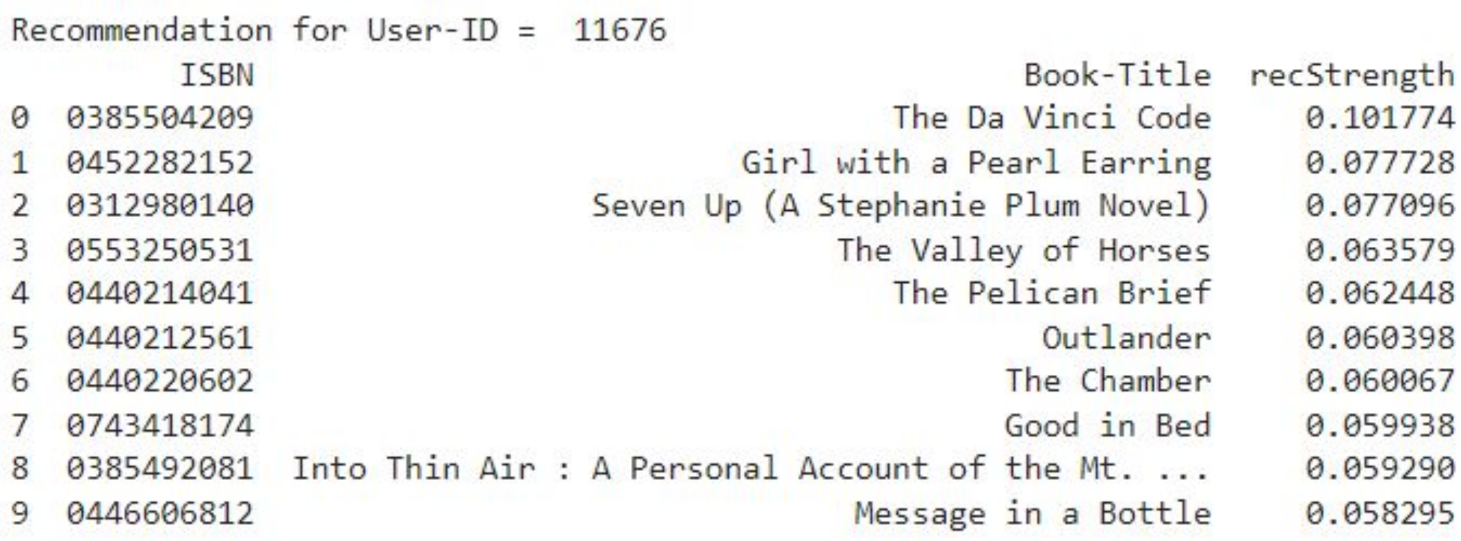
```
Recommendations for The Bell Jar:

1: Girl, Interrupted, with distance of 0.870524126645689:
2: Lily White, with distance of 0.8788241399871681:
3: A Patchwork Planet (Ballantine Reader's Circle), with distance of 0.8810795016762331:
4: What We Keep : A Novel (Ballantine Reader's Circle), with distance of 0.8904935335360462:
5: The Love Letter, with distance of 0.897842379701167:
```

We can see, that the recommended books, are quite similar in genre to the selected item

## 3:- Memory Based Collaborative Filtering Recommender\

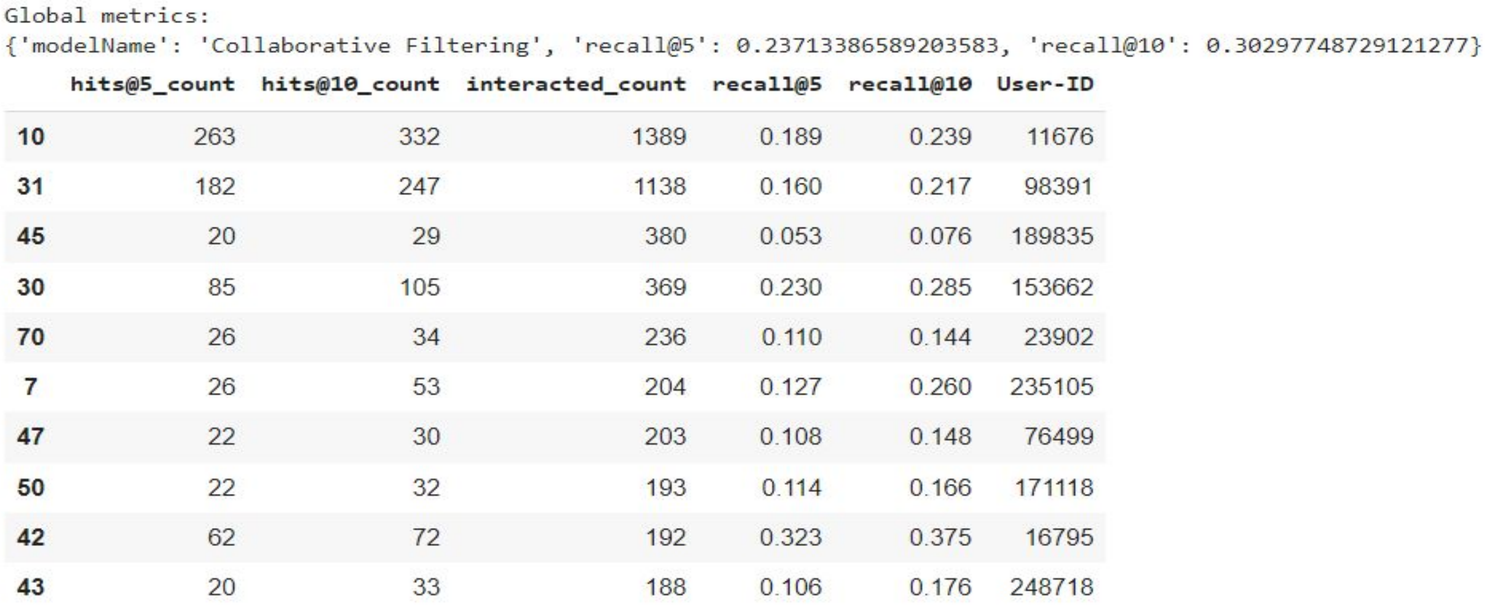
- Collaborative Filtering (User-Item based)



```
Recommendation for User-ID =  11676
         ISBN                                       Book-Title  recStrength
0  0385504209                                The Da Vinci Code     0.101774
1  0452282152                        Girl with a Pearl Earring     0.077728
2  0312980140                Seven Up (A Stephanie Plum Novel)     0.077096
3  0553250531                             The Valley of Horses     0.063579
4  0440214041                                The Pelican Brief     0.062448
5  0440212561                                        Outlander     0.060398
6  0440220602                                      The Chamber     0.060067
7  0743418174                                      Good in Bed     0.059938
8  0385492081  Into Thin Air : A Personal Account of the Mt. ...     0.059290
9  0446606812                              Message in a Bottle     0.058295
```

# Model Evaluation

In Recommender Systems, there are a set metrics commonly used for evaluation. We choose to work with Top-N accuracy metrics, which evaluates the accuracy of the top recommendations provided to a user, comparing to the items the user has actually interacted in test set.



```
Global metrics:
{'modelName': 'Collaborative Filtering', 'recall@5': 0.23713386589203583, 'recall@10': 0.30297748729121277}
```

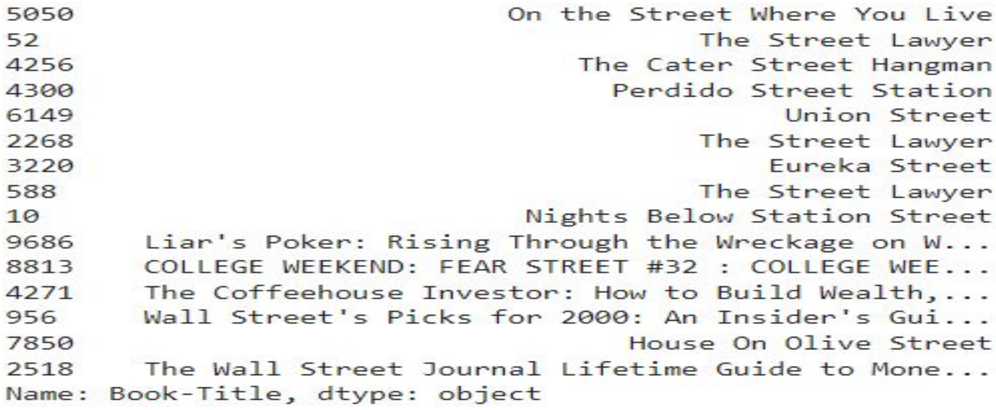
| | hits@5_count | hits@10_count | interacted_count | recall@5 | recall@10 | User-ID |
|---|---|---|---|---|---|---|
| **10** | 263 | 332 | 1389 | 0.189 | 0.239 | 11676 |
| **31** | 182 | 247 | 1138 | 0.160 | 0.217 | 98391 |
| **45** | 20 | 29 | 380 | 0.053 | 0.076 | 189835 |
| **30** | 85 | 105 | 369 | 0.230 | 0.285 | 153662 |
| **70** | 26 | 34 | 236 | 0.110 | 0.144 | 23902 |
| **7** | 26 | 53 | 204 | 0.127 | 0.260 | 235105 |
| **47** | 22 | 30 | 203 | 0.108 | 0.148 | 76499 |
| **50** | 22 | 32 | 193 | 0.114 | 0.166 | 171118 |
| **42** | 62 | 72 | 192 | 0.323 | 0.375 | 16795 |
| **43** | 20 | 33 | 188 | 0.106 | 0.176 | 248718 |

As we can see that our recom-system work fine and gives 0.23 recall@5 which is fine enough.

# 4:- Content Based Filtering Recommender

- For Content Based Book Recommendation we have to use NLP techniques like Keyword extraction.
- Keyword extraction is automatic detection of terms that best describe the subject of a document.
- For keyword extraction we tried both of the following,
    - **Countvectorizer**
    - **Tf-Idf Vectorizer**

## a. Content-Based Recommendation on the basis of Book-Title( **with count-vectorizer**)

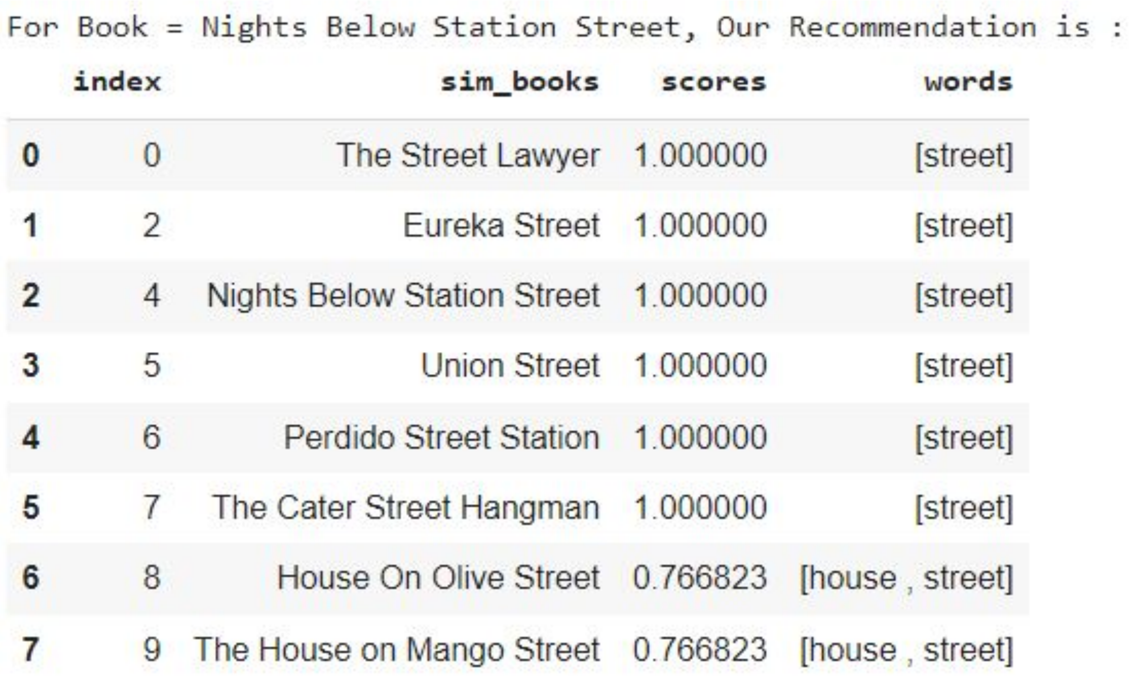


```
5050                     On the Street Where You Live
52                                   The Street Lawyer
4256                          The Cater Street Hangman
4300                            Perdido Street Station
6149                                      Union Street
2268                                 The Street Lawyer
3220                                    Eureka Street
588                                  The Street Lawyer
10                         Nights Below Station Street
9686     Liar's Poker: Rising Through the Wreckage on W...
8813     COLLEGE WEEKEND: FEAR STREET #32 : COLLEGE WEE...
4271     The Coffeehouse Investor: How to Build Wealth,...
956      Wall Street's Picks for 2000: An Insider's Gui...
7850                             House On Olive Street
2518     The Wall Street Journal Lifetime Guide to Mone...
Name: Book-Title, dtype: object
```

As we can see all the books with similar to 'Street' will be recommended by this recommender.

# 4:- Content Based Filtering Recommender

## b. Content-Based Recommendation on the basis of Book-Title (**with tfidf-vectorizer**)

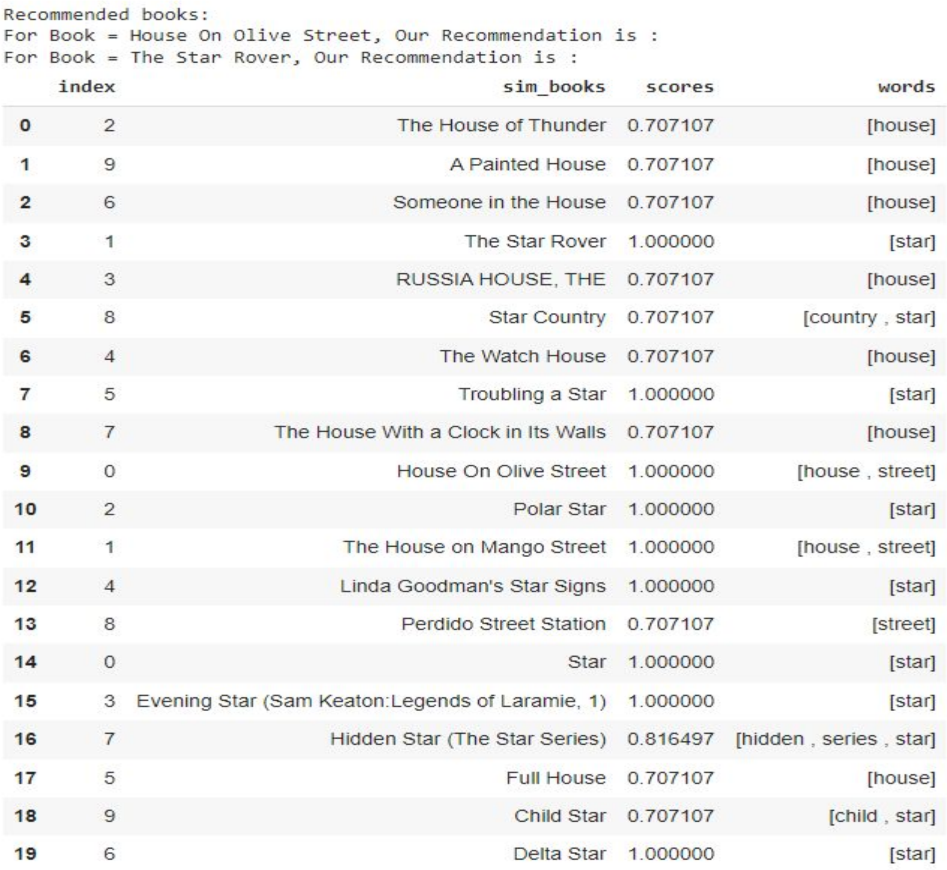


```
For Book = Nights Below Station Street, Our Recommendation is :
```

| | index | sim_books | scores | words |
|---|---|---|---|---|
| **0** | 0 | The Street Lawyer | 1.000000 | [street] |
| **1** | 2 | Eureka Street | 1.000000 | [street] |
| **2** | 4 | Nights Below Station Street | 1.000000 | [street] |
| **3** | 5 | Union Street | 1.000000 | [street] |
| **4** | 6 | Perdido Street Station | 1.000000 | [street] |
| **5** | 7 | The Cater Street Hangman | 1.000000 | [street] |
| **6** | 8 | House On Olive Street | 0.766823 | [house , street] |
| **7** | 9 | The House on Mango Street | 0.766823 | [house , street] |

# 4:- Content Based Filtering Recommender

## c. Content-Based Recommendation on the basis of Book-Purchase history list



```
Recommended books:
For Book = House On Olive Street, Our Recommendation is :
For Book = The Star Rover, Our Recommendation is :
```

| | index | sim_books | scores | words |
|---|---|---|---|---|
| **0** | 2 | The House of Thunder | 0.707107 | [house] |
| **1** | 9 | A Painted House | 0.707107 | [house] |
| **2** | 6 | Someone in the House | 0.707107 | [house] |
| **3** | 1 | The Star Rover | 1.000000 | [star] |
| **4** | 3 | RUSSIA HOUSE, THE | 0.707107 | [house] |
| **5** | 8 | Star Country | 0.707107 | [country , star] |
| **6** | 4 | The Watch House | 0.707107 | [house] |
| **7** | 5 | Troubling a Star | 1.000000 | [star] |
| **8** | 7 | The House With a Clock in Its Walls | 0.707107 | [house] |
| **9** | 0 | House On Olive Street | 1.000000 | [house , street] |
| **10** | 2 | Polar Star | 1.000000 | [star] |
| **11** | 1 | The House on Mango Street | 1.000000 | [house , street] |
| **12** | 4 | Linda Goodman's Star Signs | 1.000000 | [star] |
| **13** | 8 | Perdido Street Station | 0.707107 | [street] |
| **14** | 0 | Star | 1.000000 | [star] |
| **15** | 3 | Evening Star (Sam Keaton:Legends of Laramie, 1) | 1.000000 | [star] |
| **16** | 7 | Hidden Star (The Star Series) | 0.816497 | [hidden , series , star] |
| **17** | 5 | Full House | 0.707107 | [house] |
| **18** | 9 | Child Star | 0.707107 | [child , star] |
| **19** | 6 | Delta Star | 1.000000 | [star] |

# Conclusion

Building a model to recommend another books is extremely beneficial to the company because it can increase their sales via recommend relevant books to their customers and optimise its business model and revenue accordingly.

- For modelling, it was observed that for model based collaborative filtering SVD technique worked way better than NMF with lower Mean Absolute Error (MAE) .
- Amongst the memory based approach, item-item CF performed better than user-item CF because of lower computation.
- Content-based recommendation on the basis of Tags are also doing good in terms of results.

## Key points :

- Customers of age between 20 to 30 are more likely to buy books.
- Customers who are in USA are more likely to buy books than others.
- Our overall top selling authors are Agatha Cristie, William Shakespeare and Stephen King.
- If we look at the ratings distribution, most of the books have high ratings with maximum books being rated 8. Ratings below 5 are few in number.
- Our overall top selling publishers are Harlequin, Silhouette and Pocket.
- Our overall top selling books are The Lovely Bones: A Novel, Wild Animus and The Da Vinci Code, The Red Tent (Bestselling Backlist). .

# Improvements :

- By using a marketing and advertising approach, we can reduce the age-gap.
- We can clearly see that we have a larger number of buyer within USA, therefore we can easily recommend books to them on the basis of location and use this strategy for our campaign.
- We nearly make 10 recommender system from which we can select Best according to our Business-needs.
- We can push those type of books to publish which are similar to our top-selling books and recommend them to our Users.

# Future Work

We can recommend books to our customers on basis of genres also but we have no information on that so we have to record books genres also for better recommendation.

- We can also record Date-time of our users when they buy book, By using that we can recommend our top books, authors, publication on monthly basis.
- Given more information regarding the books dataset, namely features like Genre, Description etc, we could implement a content-filtering based recommendation system and compare the results with the existing collaborative-filtering based system.
- We would like to explore various clustering approaches for clustering the users based on Age, Location etc., and then implement voting algorithms to recommend items to the user depending on the cluster into which it belongs.

THANK YOU