# Capstone project - 4

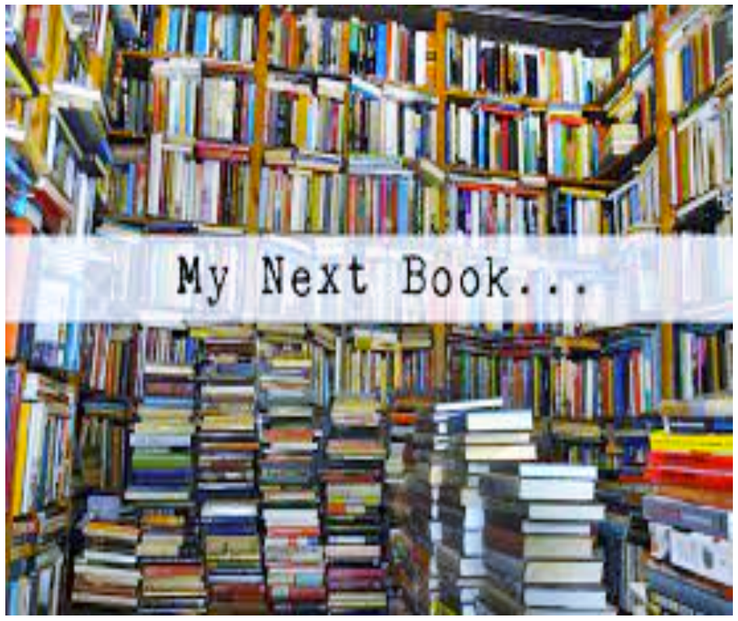
## Book Recommendation System

Individual project
Name:ramji seth
Email:ramjiseth9755@gmail.com

# Content

- Problem statement
- Data Summary
- Analysis of different datasets
- Data Cleaning
- Outlier treatment
- Imputing missing values
- Different Recommendation Model
- Challenges
- Conclusion
- Future Scope

# Problem Statement




During the last few decades,with the rise of Youtube, Amazon, Netflix, and many other such web services, recommender systems have become much more important in our lives in terms of providing highly personalized and relevant content.

The main objective is to create a recommendation system to recommend relevant books to users based on popularity and user interests.

# Data Summary

1. The dataset is comprised of three csv files:: 1) users 2) books 3) ratings

**Users_dataset**

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

➢ Shape of Dataset- (278858,3)

**Ratings_dataset**

- User_ID
- ISBN
- Book-Rating

➢ Shape of Dataset -(1149780,3)

**Books_dataset**

- ISBN (unique for each book)
- Book-Title
- Book-Author
- Year-Of-Publication
- Publisher
- Image-URL-S
- Image-URL-M
- Image-URL-L

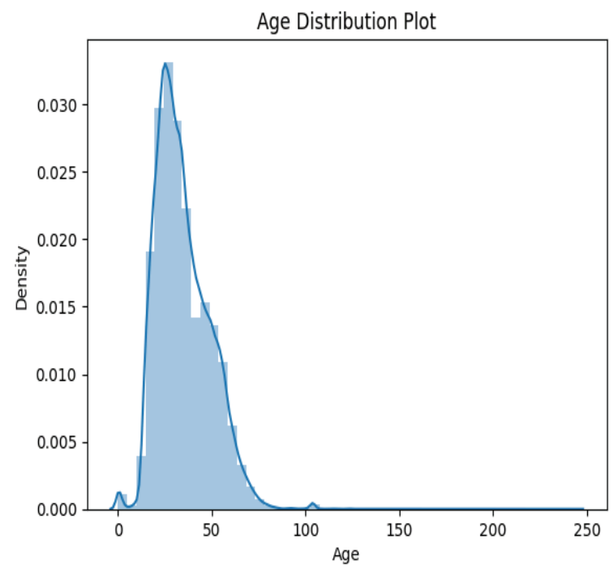
➢ Shape of Dataset - (271360,8)

# Observations from "Users" dataset(Age)

- The Age range given here is from 0 to 250.
- Outliers are in the Age column.

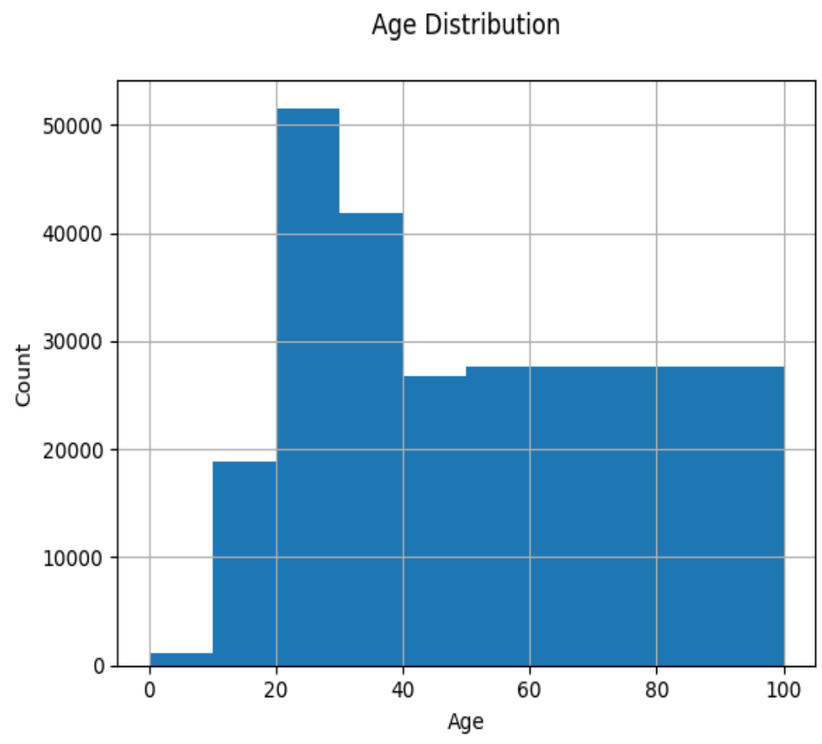
```
sns.distplot(users.Age)
plt.title('Age Distribution Plot')
```

```
Text(0.5, 1.0, 'Age Distribution Plot')
```



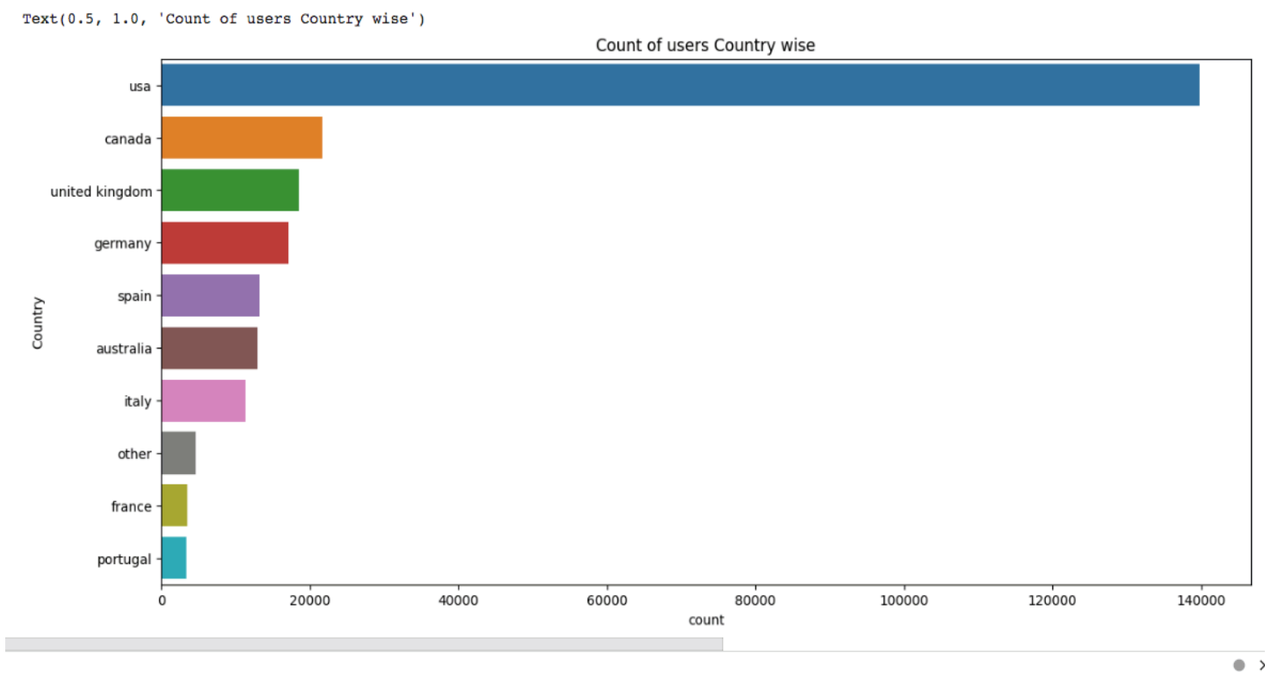
## Here is the AGE DISTRIBUTION graph and we found that-

- The Age range distribution is right skewed
- Most active readers lie in age group 20-40



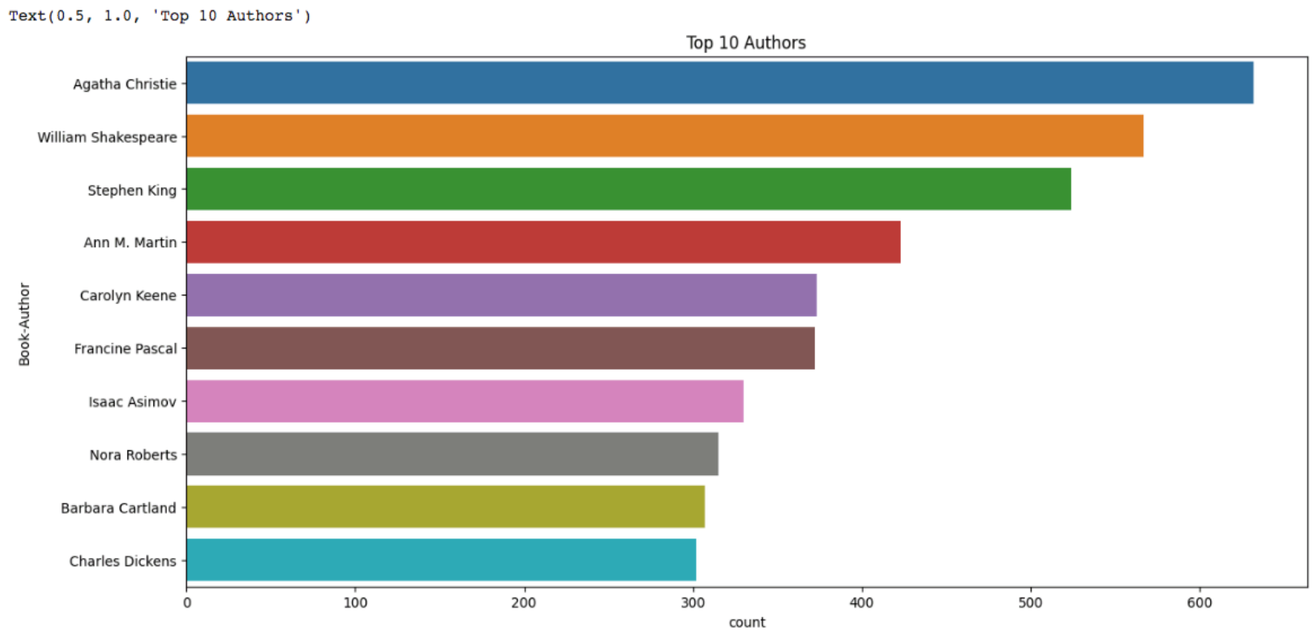
## Here we are Splitting Location column and analyzing country and found that-

- The most active readers are from USA.



# Observations from “books” dataset(Authors)

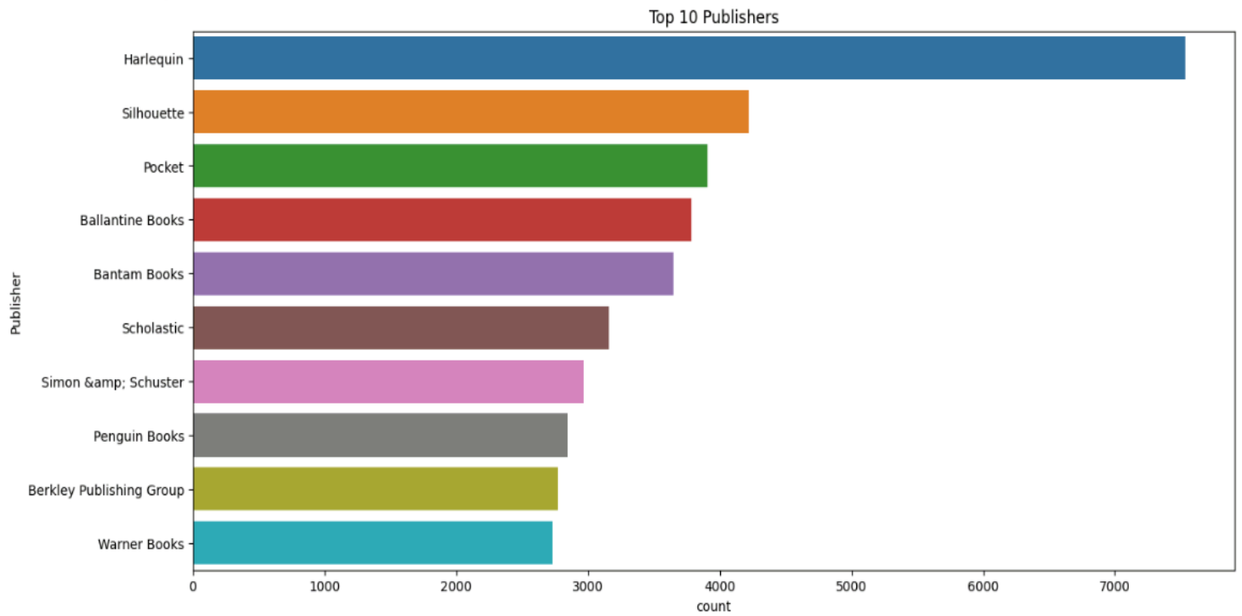
- Agatha Christie wrote highest number of books in our given dataset



# Observations from “books” dataset(Publishers)

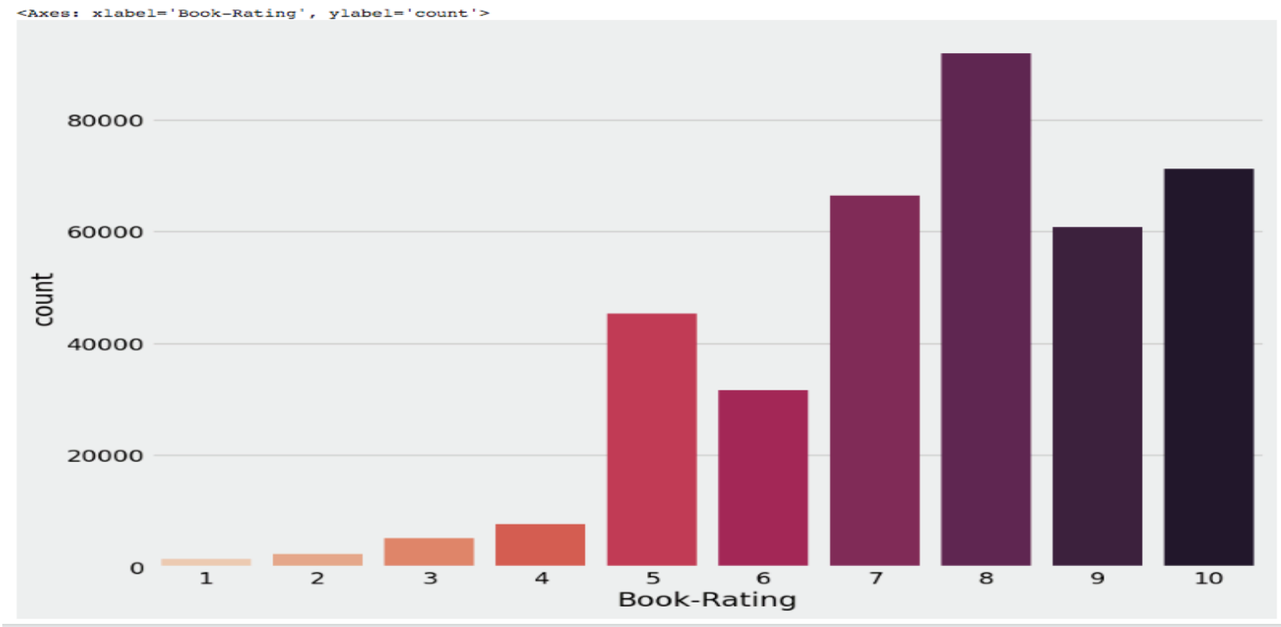
- Harlequin published highest number of books in our given dataset.

```
Text(0.5, 1.0, 'Top 10 Publishers')
```



# Observations from "Ratings" dataset

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



# Data Cleaning from “users” dataset

## 1. Null Value Imputation

Age column has 40% of missing values

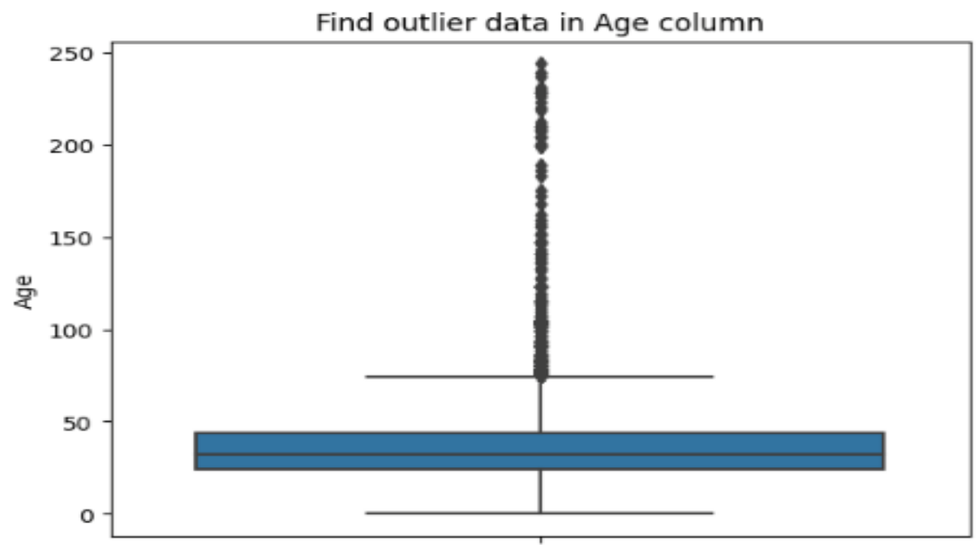
| | index | Missing Values | % of Total Values | Data_type |
|---|---|---|---|---|
| 0 | Age | 110762 | 39.72 | float64 |
| 1 | User-ID | 0 | 0.00 | int64 |
| 2 | Location | 0 | 0.00 | object |

## Imputing missing values

As we know that the outliers are in Age column and

Age has positive Skewness (right tail) so we can use median to fill Nan values

```
Text(0.5, 1.0, 'Find outlier data in Age column')
```



## Data Cleaning from “books” dataset

**1. Null Value Imputation:**

```
books_df.isnull().sum()
```

```
ISBN                    0
Book-Title              0
Book-Author             1
Year-Of-Publication     0
Publisher               2
Image-URL-S             0
Image-URL-M             0
Image-URL-L             3
dtype: int64
```

## Replacing *strings* by *int* values

| | ISBN | Book-Title | Book-Author | Year-Of-Publication | |
|---|---|---|---|---|---|
| **209538** | 078946697X | DK Readers: Creating the X-Men, How It All Beg... | 2000 | DK Publishing Inc | ht |
| **221678** | 0789466953 | DK Readers: Creating the X-Men, How Comic Book... | 2000 | DK Publishing Inc | h |

# Different Models

## 1) Popularity Based Recommendation

Book weighted average formula:

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

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.

# Different Models

| | 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 |

# Different Models

2) Model based collaborative filtering

**SVD**

```
test_rmse    1.602006
test_mae     1.239913
fit_time     2.518859
test_time    0.697736
dtype: float64
```

**NMF**

```
test_rmse    2.621385
test_mae     2.238251
fit_time     6.553807
test_time    0.452583
dtype: float64
```

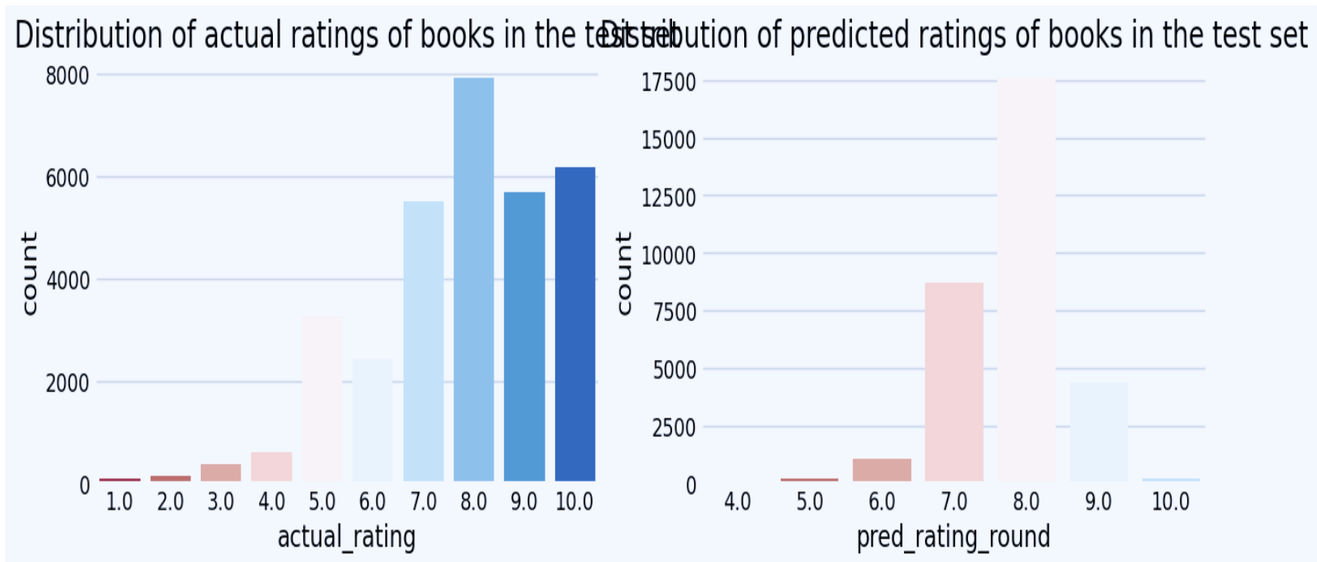
# Different Models

## SVD Model Results

| | user_id | isbn | actual_rating | pred_rating | impossible | pred_rating_round | abs_err |
|---|---|---|---|---|---|---|---|
| **31844** | 121470 | 0743230213 | 4.0 | 7.378298 | False | 7.0 | 3.378298 |
| **12681** | 120908 | 0452281903 | 8.0 | 8.261429 | False | 8.0 | 0.261429 |
| **30945** | 16488 | 0671724738 | 7.0 | 7.769623 | False | 8.0 | 0.769623 |
| **30892** | 163570 | 0394571029 | 9.0 | 8.730977 | False | 9.0 | 0.269023 |
| **2199** | 243930 | 1401301061 | 10.0 | 7.306892 | False | 7.0 | 2.693108 |

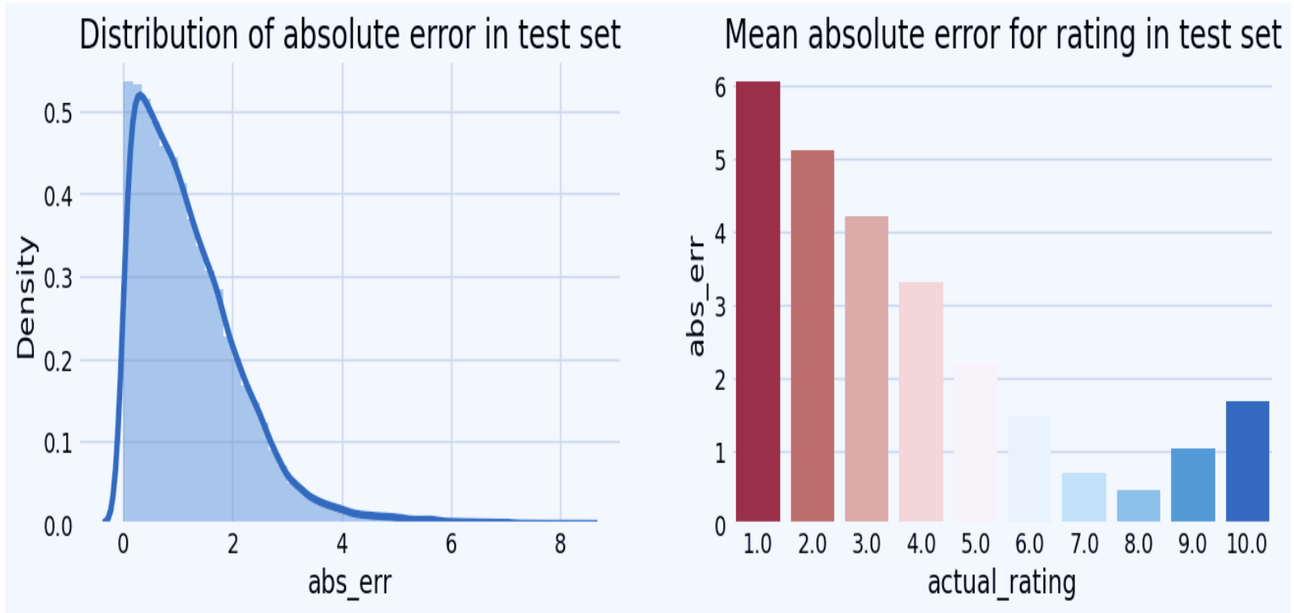
# Different Models

## SVD Model Results



# Different Models

## SVD Model Results



# Collaborative Filtering-(Item-Item based)

### 3.) Collaborative Filtering-(Item-Item based)

- **Cosine Similarity**
- **Nearest Neighbour**

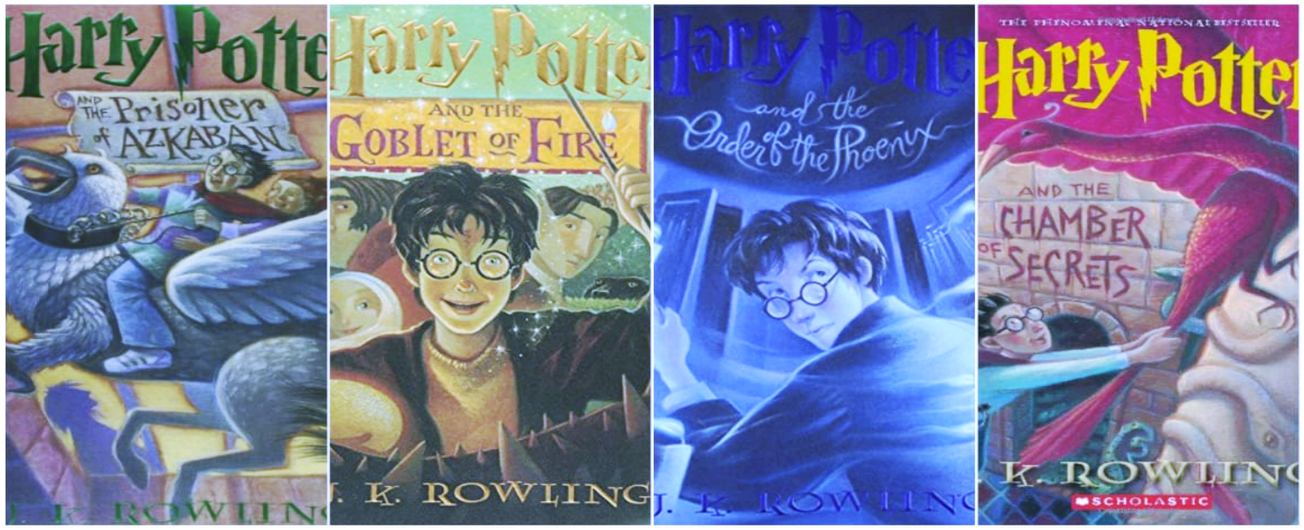
```
Recommendations for Valhalla Rising (Dirk Pitt Adventures (Paperback)):

1: The President's Daughter, with distance of 0.8941085151275194:
2: Cybernation (Tom Clancy's Net Force, No. 6), with distance of 0.916775927165431:
3: Open Season, with distance of 0.9194460840751736:
4: Blackout, with distance of 0.9226373674880946:
5: Tom Clancy's Op-Center: Line of Control (Tom Clancy's Op Center (Paperback)), with distance of 0.9267344495560187:
```

# Different Models

SVD & Correlation

**Recommendations for Harry Potter and the Sorcerer's Stone(Book 1)**



# Different Models

4) Collaborative Filtering-(User-Item-based)

```
Enter User ID from above list for book recommendation  69078
Recommendation for User-ID =  69078
         ISBN                                          Book-Title  recStrength
0  0446310786                              To Kill a Mockingbird        0.843
1  0345370775                                     Jurassic Park        0.798
2  0345361792                          A Prayer for Owen Meany         0.703
3  0316769487                        The Catcher in the Rye        0.657
4  0440214041                             The Pelican Brief        0.646
5  0312966970          Four To Score (A Stephanie Plum Novel)        0.641
6  0440211727                                 A Time to Kill        0.622
7  044021145X                                     The Firm        0.615
8  0060928336  Divine Secrets of the Ya-Ya Sisterhood: A Novel        0.598
9  0553572997                                 The Alienist        0.586
```

# Conclusion

● In EDA, the Top-10 most rated books were essentially novels. Books like '**The Lovely Bone**' and '**The Secret Life of Bees**' were very well perceived.

● Majority of the readers were of the age bracket 20-35 and most of them came from North American and European countries namely USA, Canada, UK, Germany and Spain.

● 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.

● Author with the most books was Agatha Christie, William Shakespeare and Stephen King.

● For modelling, it was observed that for model based collaborative filtering SVD technique worked way better than NMF with lower Mean Absolute Error (MAE) .

## Conclusion

A recommendation system helps an organization to create loyal customers. The recommendation system today are very powerful that they can handle the new customer too who has visited the site for the first time. They recommend the products which are currently trending or highly rated and they can also recommend the products which bring maximum profit to the company.

# Challenges

- Handling of sparsity was a major challenge as well since the user interactions were Not present for the majority of the books.

- Understanding the metric for evaluation was a challenge as well.

- Since the data consisted of text data, data cleaning was a major challenge in features like Location etc..

- Decision making on missing value imputations and outlier treatment was quite challenging as well.

# Future Scope

- 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