Movie Recommendation System

IMT2021042-Raghunadh
IMT2021035-Kushal Dasari
IMT2021034-Prachoday

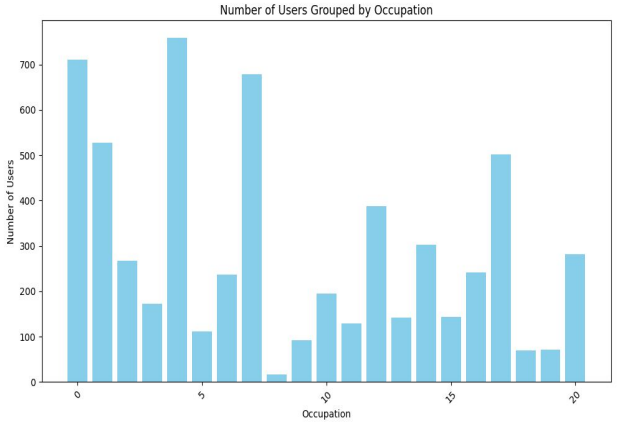
Problem Statement

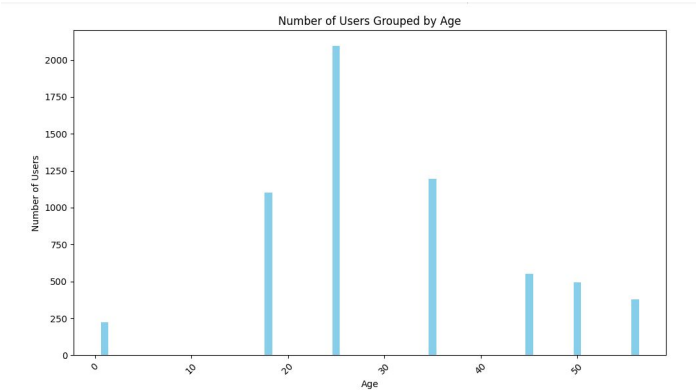
Design a Movie Recommendation System using the given ratings of Movies.

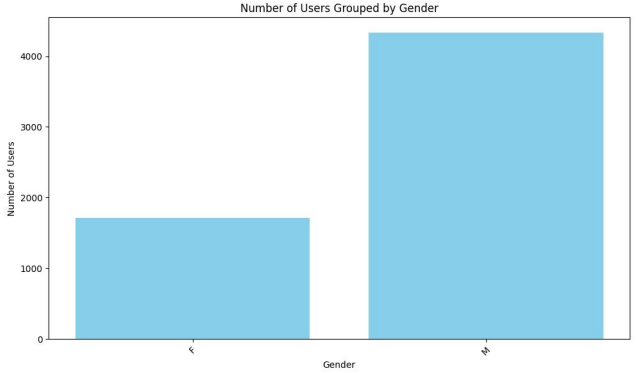
The dataset given to us has the users table which consists of information about 6040 users and 3883 movies and has the ratings given by users to all movies.

Our objective is to predict top 5 movies for a user, based on the ratings of that user for movies he/she have taken.

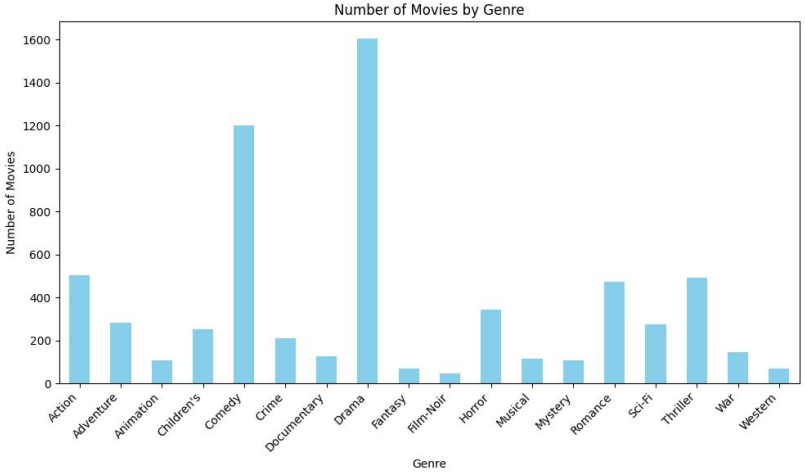
EDA

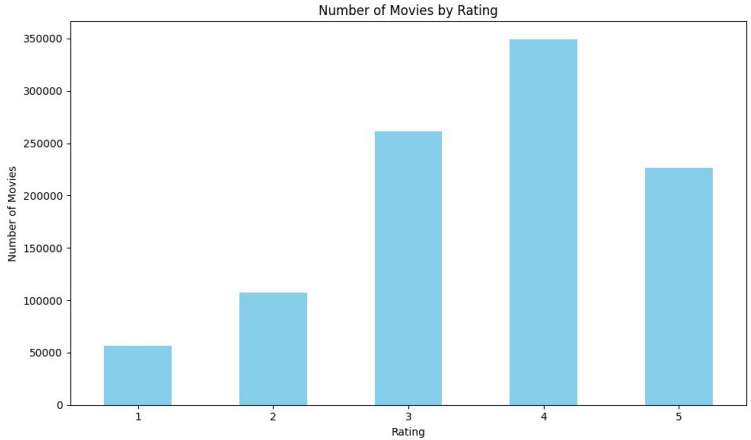

Number of Users Grouped by Occupation
Number of Users
Occupation
0
100
200
300
400
500
600
700
0
5
10
15
20


Number of Users Grouped by Age
Number of Users
Age
0
250
500
750
1000
1250
1500
1750
2000
0
10
20
30
40
50

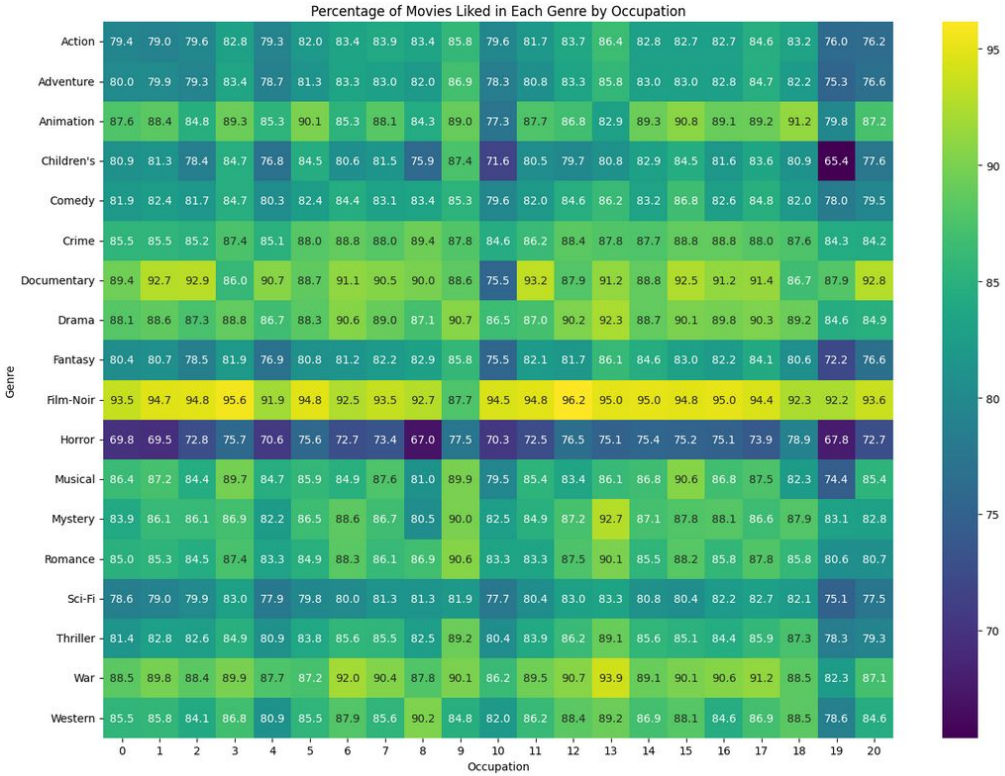

Number of Users Grouped by Gender
Number of Users
Gender
0
1000
2000
3000
4000
F
M

Movies data summary

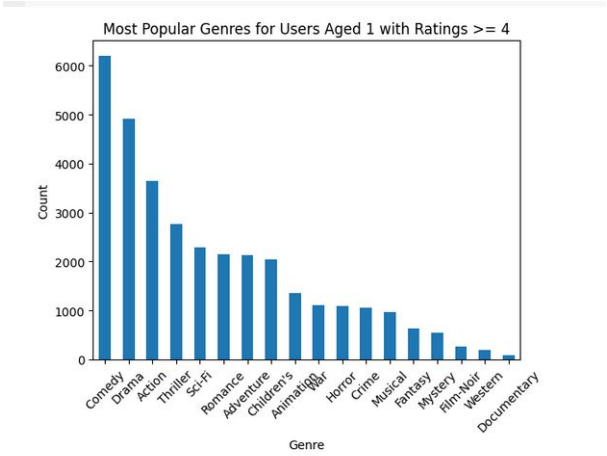


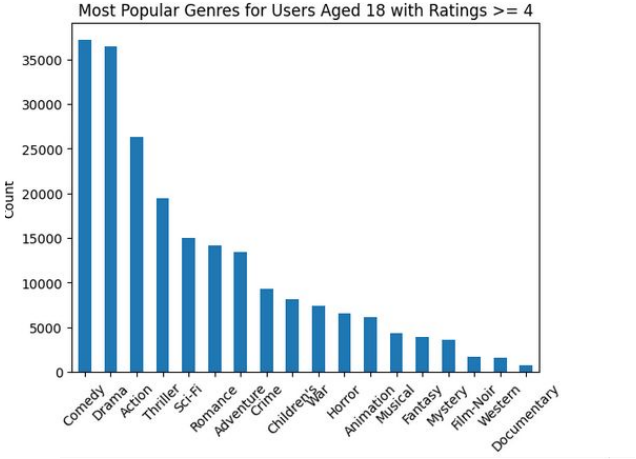


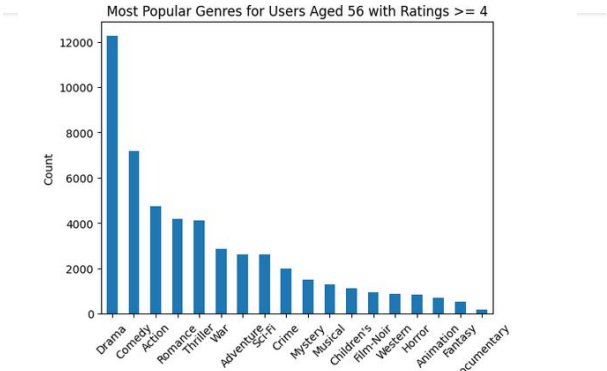
Percentage of liked movies based on occupation

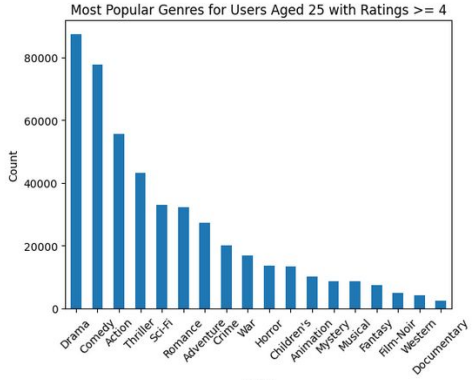


Movies liked by different age groups

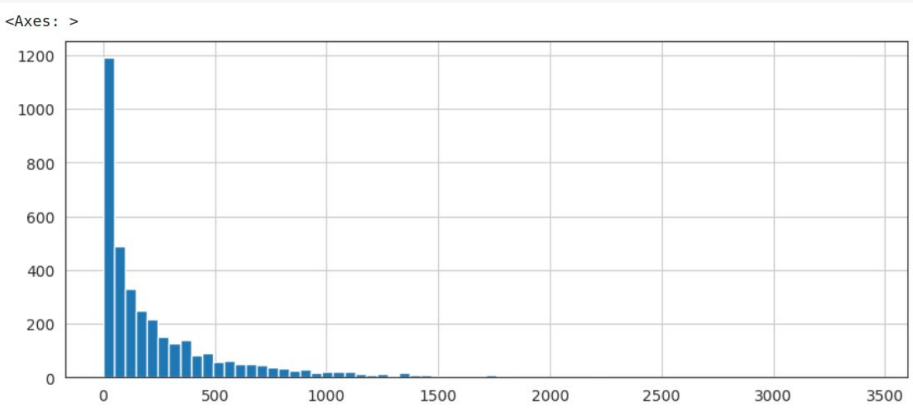




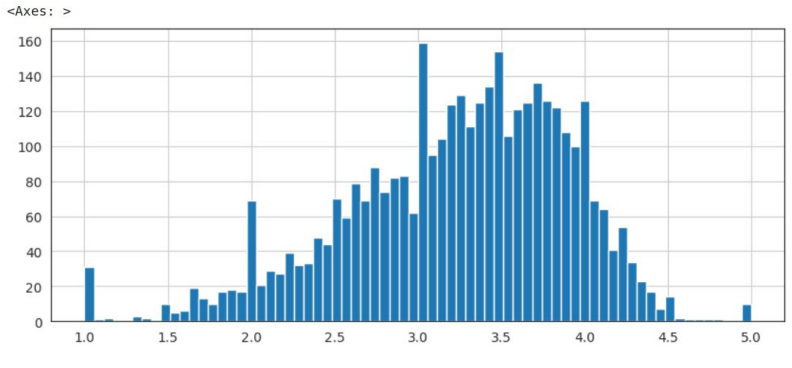




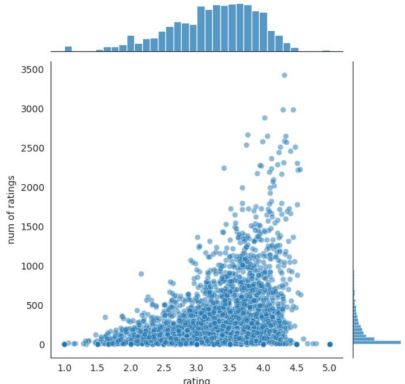
Movies vs ratings



Movies vs number of ratings



Movies vs average ratings



Inferences from data

- We observe that users of same age have similar interests
- Users of age group 1 like comedy genre the most
- Users of age group 56 like drama genre the most

Components used in our project

K Means

- We did K Means plus plus implementation where we initialize centroids randomly and such that they are far away
- We used an elbow plot for deciding the number of clusters

SVD

- We used power iteration method to SVD
- We compare nth and n-1th iteration’s eigen vectors and check whether they below the certain threshold/tolerance (chosen 1e-6).

Methods used

We used the following methods for building our movie recommendation system

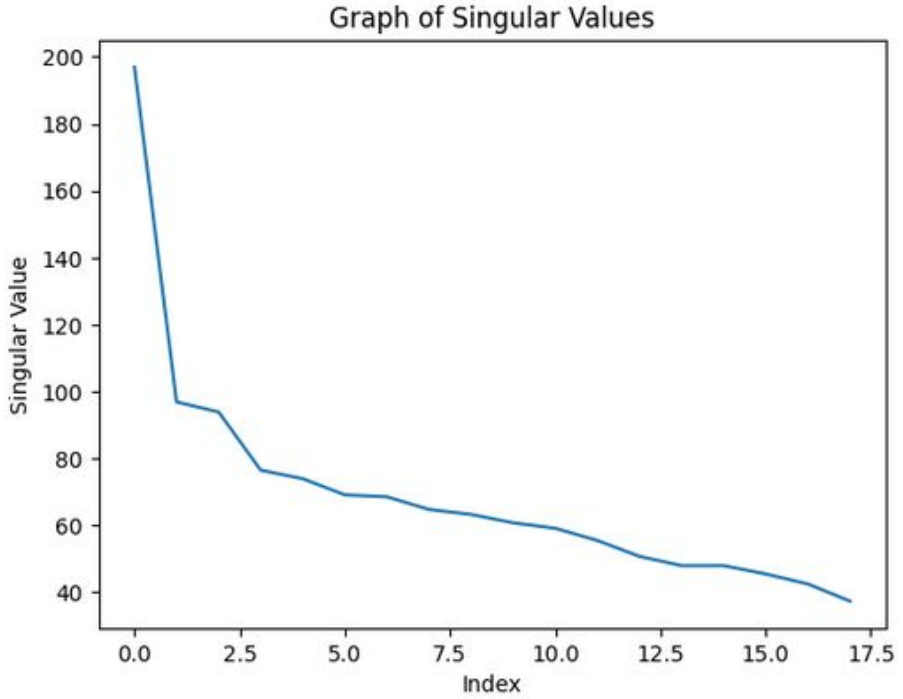
- Collaborative Filtering (User based and Item based)
- Performing SVD and then K-Means

SVD and K Means

- We first created a user genre matrix using user and movie tables of the dataset
- We filled the null values in the matrix by 2.5 because there are very less null values
- We did standardisation to each row to remove any sort of user bias
- We applied truncated SVD to the matrix
- Divided users into clusters with the user vectors obtained

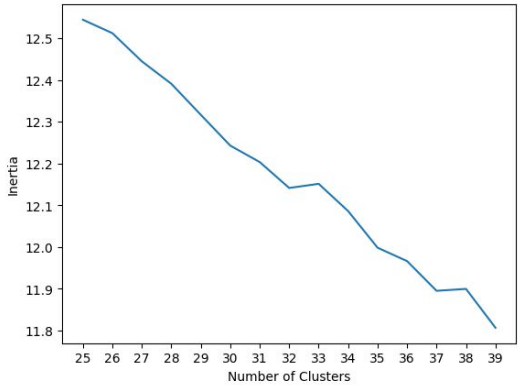
SVD to the user genre matrix

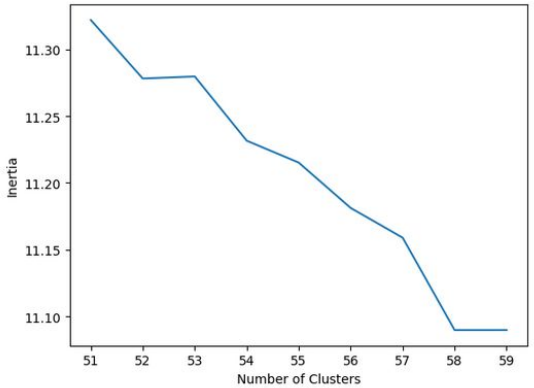
This is the plot of singular values we obtained for the user-genre matrix.

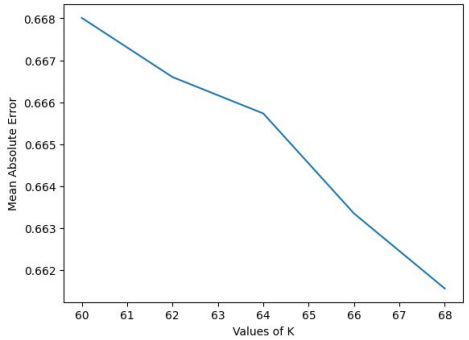


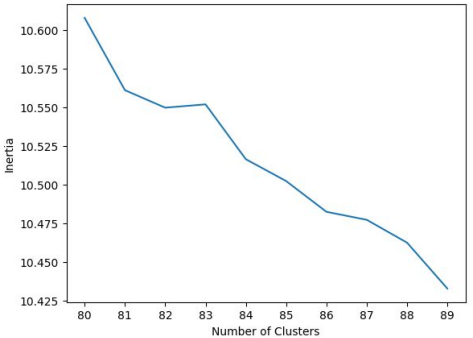
Deciding the value of k using elbow plot

From the elbow plots obtained to us, we decided the keep the value of k to be 55

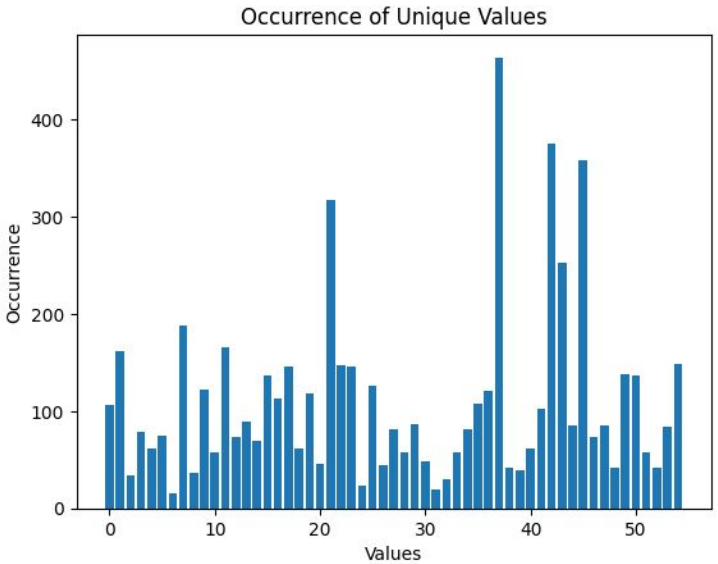








This is the number of users in each cluster after applying K Means



Deciding the number of eigen values

- We used mse to do this
- We predicted the rating of movies for which we know and calculated mse by removing eigen values and performed reduced svd
- As errors are nearly same and eigen values are not too small, we decided not to remove any

Errors obtained after removing 0 to 5 eigen values

```
1.0411323945398692
1.198071897581552
0.989879968615881
1.0891818496921792
1.1280021347139546
```

Predicting top 5 movies

- We will first took the user vector and predicted the cluster it belongs to using the centroids obtained after doing K Means
- We calculated the average of ratings of movies rated by the users in the cluster to which he belongs to.
- We assigned a weight to each rating as

  (number of reviews movie got)/number of users in the cluster)

- This is because if only one user in the cluster has rated it, it would have high rating
- We found the top 5 movies by calculating the rating of each movie watched by users in the cluster and sorting the obtained rating in descending order

Approach 2(user-user)

We perform user based collaborative filtering and Item collaborative filtering.

● We performed the following steps on the data:

1.create a matrix that has the user ids on one access and the movie id on another axis.Each cell will then consist of the rating the user gave to that movie. Note there will be a lot on Nan values, because most people have not seen most of the movies.

2.Then we compute the mean value for every user for handling bias.

3. Then we compute similarity between users using various correlation functions(Pearson correlation,spearman correlation,jaccard similarity,manhattan correlation), and store it in a list.

4. Identify items rated by similar users for the user in question:

5. Calculated weighted average scores.

4. Rank them.

Novelty Incorporation: Age Group Similarity

- In our recommendation system, we introduce the concept of age group similarity as a novel factor influencing user collaborative filtering. The rationale behind this addition stems from the observation that individuals within similar age groups tend to share common preferences and interests in terms of the content they consume, particularly in the context of movies.

```
# Calculate age proximity similarity
age_similarity = 1 / (1 + np.abs(age1 - age2))

# Combine similarity and age proximity with weights
similarity_with_age = 0.85 * correlation + 0.15 * age_similarity
```

Predicting top 5 movies

- If new users enters and he/she watches some movies based on this information we need to predict top 5 movies.

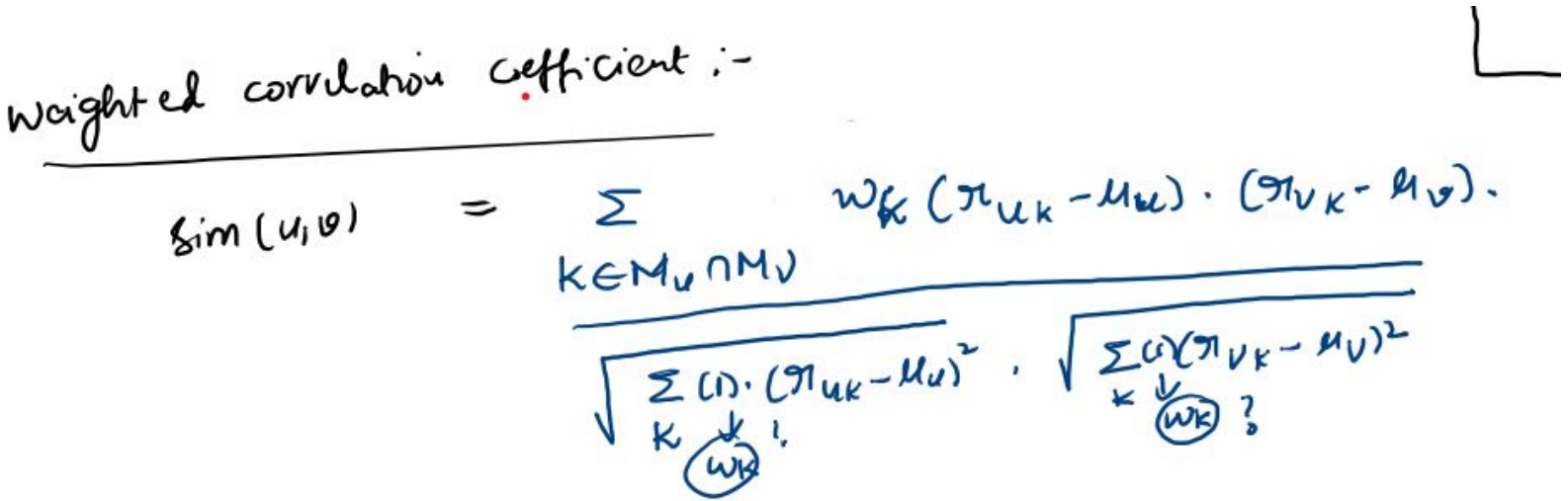
We performed the following steps on the data:

1.Create a DataFrame for the new user’s ratings, and concatenate with existing rating dataframe.

2.Then we predict the rating for all the movies in the dataframe, and sort it in descending order and take first 5 movies, These are the top 5 movies that recommend to the user

Functions used for similarity

Pearson correlation:



Prediction function

This is the function used to calculate the rating:

$$\hat{r}_{uk} = \mu_u + \frac{\sum_{v \in U_k} sim(u,v) \cdot (r_{vk} - \mu_v)}{\sum_{v \in U_k} |sim(u,v)|}$$

Approach 2(item-item)

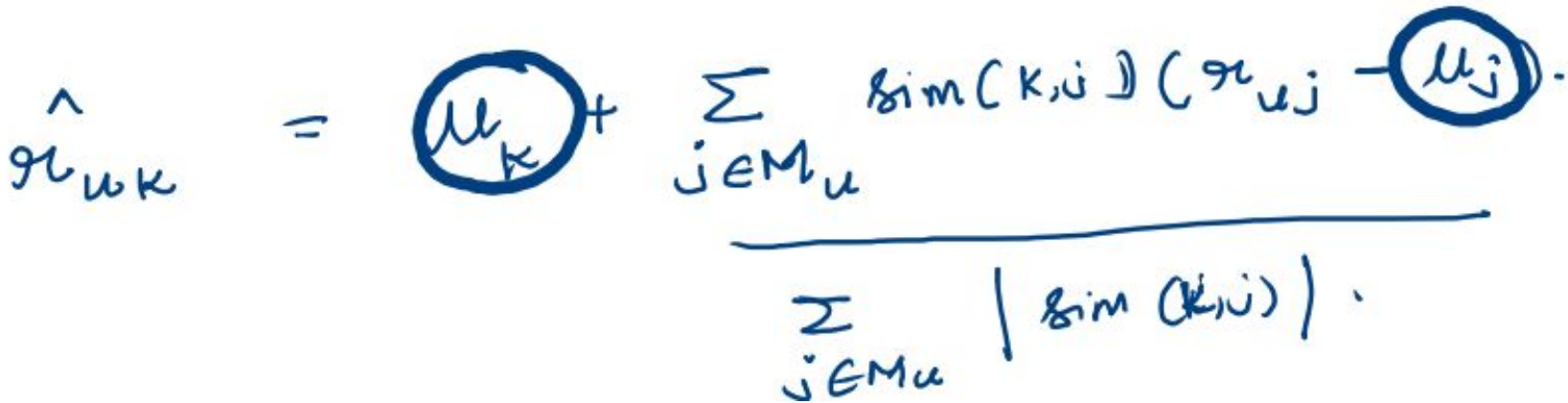
- Same like user-user we just need to reverse it.

  Similarity function used for item based collaborative filtering :

$$Sim(i,j) = \frac{\sum_{u \in U_i \cap U_j} (r_{ui} - \mu_i)(r_{uj} - \mu_j)}{\sqrt{\sum_{u \in U_i \cap U_j} (r_{ui} - \mu_i)^2} \cdot \sqrt{\sum_{u \in U_i \cap U_j} (r_{uj} - \mu_j)^2}}$$

Prediction function

This is the prediction function we used in item item collaborative filtering :



Prediction for new user using collaborative filtering

Test case we gave:

New user id:6041

Movie id’s:2,3,7;

Rating give:4,5,3;

Age:18.

Output:

Top 5 recomended movies:
Across the Sea of Time (1995)
Wonderful, Horrible Life of Leni Riefenstahl, The (Die Macht der Bilder) (1993)
Schindler's List (1993)
Seventh Heaven (Le Septième ciel) (1997)
Stage Fright (1950)

Prediction for new user using K Means and SVD

Test case we gave:

New user id:6041

Movie id's:2,3,7;

Rating give:4,5,3;

Age:18.

Output:

```
2086    2858
454      593
195      260
239      318
405      527
Name: movie_id, dtype: int64
```

```
[79] movies[movies['movie_id'] == 2858]
```

	movie_id	title	genres
2789	2858	American Beauty (1999)	Comedy|Drama

```
[80] movies[movies['movie_id'] == 593]
```

	movie_id	title	genres
589	593	Silence of the Lambs, The (1991)	Drama|Thriller

```
[81] movies[movies['movie_id'] == 260]
```

	movie_id	title	genres
257	260	Star Wars: Episode IV - A New Hope (1977)	Action|Adventure|Fantasy|Sci-Fi

```
[82] movies[movies['movie_id'] == 318]
```

	movie_id	title	genres
315	318	Shawshank Redemption, The (1994)	Drama

```
movies[movies['movie_id'] == 527]
```

	movie_id	title	genres
523	527	Schindler's List (1993)	Drama|War

Conclusion

- We conclude that the recommendatioins by SVD are better.
- SVD plus k-means combines the strengths of matrix factorization (SVD) and clustering (k-means) techniques, allowing for a more comprehensive understanding of user-item interactions.
- This hybrid approach captures both explicit and implicit user preferences, leading to more accurate recommendations