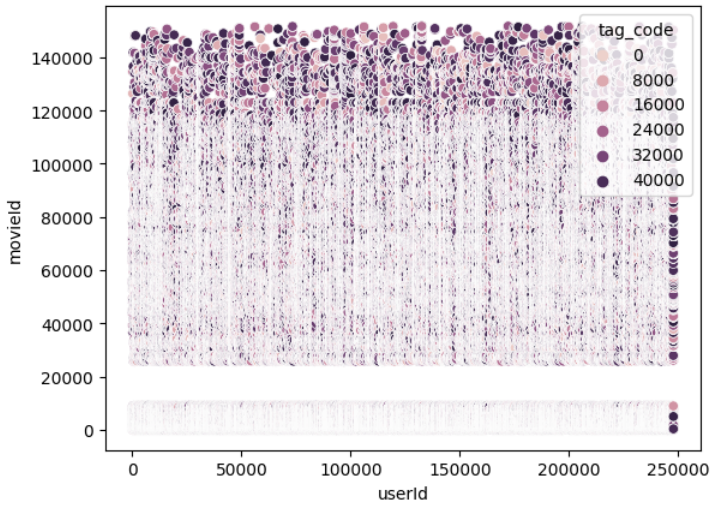
Movie Recommendation Machine Learning Model

Team :

Ankur Yadav B21CS011
Adarsh Raj Shrivastava B21AI003

Given Datasets(link):

- Link dataset: This file contains metadata about movies, including the movie ID, the IMDB ID, and the TMDB ID.
- Movies dataset: This file contains description of movies including movie ID, movie title and its genre.
- Ratings dataset: This file contains four columns: user ID, movie ID, rating, timestamp. It provides the user-item interaction data
- Tags dataset: This file contains tags/comments given by users to movies.



The resulting plot shows the distribution of tags across different users and movies.

Then Preprocessed the data in which we checked for the Null values and handled them using fillna attribute and SimpleImputer. Normalization of Data was not required as we had to build a recommendation system.

Popularity-Based :

It is a type of recommendation system which works on the principle of popularity which in this case is the ratings. This system checks about the movies which are highly rated and are most popular among the users and directly recommend those.
This system was implemented in the form of a Class named ‘popularity_based’, the class contains -:

- '_init_' constructor to give the datasets used for recommendation.
- 'movie_r' function which takes the movie title as input for finding its genres, then finds all the movies along their IDs with respect to the genres.
- 'fit' function which takes 'n' number of movies to recommend as input, calls the above function to get movie IDs and finds their respective ratings, then it sorts the movies according to their rating in descending order so that movie with highest rating is recommended first, then it selects the top n movies and returns the result.

For giving input as 'Toy Story (1995)' and n = 10 the following movies were recommended-

```
Top 10 Recommended Movies are ->
['Miranda (1985)',
 'Transistor Love Story (Monrak Transistor) (2001)',
 'Miss Castaway and the Island Girls (2004)',
 'Deux mondes, Les (2007)',
 'Sterile Cuckoo, The (1969)',
 'Tortoise and the Hare, The (1935)',
 'We Are the Best! (Vi är bäst!) (2013)',
 'Jewtopia (2012)',
 'Elämältä kaiken sain',
 'Those Calloways (1965)']
```

Advantages :

- It is good for new users who do not have any interaction history with the user.
- There is no need for the user's historical data.
- They are very simple to implement

Disadvantages :

- The results are not personalized.
- Does not consider user context i.e. recommends movies by rating majorly.
- The system would recommend the same sort of products/movies which are solely based upon popularity relative to every other user

Collaborative Filtering :

Method 1 :

This method makes use of ratings and movies dataset to recommend movies. In this user will search for a movie and it will recommend n movies based on this movie recommendation model. It is based on item based collaborative filtering.

Implementation :

1) First reduce the big ratings data into a small dataset as it can't be handled by colab. The original dataset contains about 2.28 crore which is reduced to about 20 lakh data points.
Reduced data :

Reduced Rating Dataset :

	userId	movieId	rating	timestamp
0	1	169	2.5	1204927694
1	1	2471	3.0	1204927438
2	1	48516	5.0	1204927435
3	2	2571	3.5	1436165433
4	2	109487	4.0	1436165496
...	...	...	...	...
1999995	21530	541	3.5	1175969344
1999996	21530	551	4.0	1439178392
1999997	21530	553	4.0	1175919775
1999998	21530	586	1.5	1174203696
1999999	21530	587	1.5	1174198350

[2000000 rows x 4 columns]

2) Now, a pivot data was created using movieId as row and userId as column and rating given by the user to a particular movie. This makes the data easy to grasp by the model.
The Pivot data created :

userId	movieId	1	2	3	4	5	6	7	8	9	...	21521
0	1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0
1	2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0
2	3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0
3	4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0
4	5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...
20044	151507	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0
20045	151547	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0
20046	151593	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0
20047	151657	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0
20048	151661	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0

3) Then Created the csr_matrix to reduce the sparsity of pivot data as most values are 0 in pivot data.
Few samples in csr_data :

```
CSR data :
   (0, 14)       4.0
  (0, 16)        5.0
  (0, 22)        5.0
  (0, 27)        4.0
  (0, 33)        3.0
  (0, 36)        4.5
  (0, 39)        4.0
  (0, 46)        4.0
  (0, 49)        4.0
  (0, 55)        3.0
  (0, 56)        2.0
  (0, 57)        4.0
```

4) Then I extracted the Movie ID corresponding to the user movie name input and searched that ID in the pivot data.
5) Then I fit the csr_data in the KNN model distance matrix between all pairs of data points in the training set and store them in memory for later use.
6) Then find the indices of the movies with the least distance. Then returns the datarame of recommended movies title and corresponding distance.

Results :
User Input : ‘Toy Story’
Output :

	Title	Distance
1	Star Wars: Episode IV - A New Hope (1977)	0.453206
2	Independence Day (a.k.a. ID4) (1996)	0.457937
3	Back to the Future (1985)	0.482287
4	Toy Story 2 (1999)	0.483394
5	Mission: Impossible (1996)	0.484676
6	Willy Wonka & the Chocolate Factory (1971)	0.489792
7	Jurassic Park (1993)	0.491190
8	Forrest Gump (1994)	0.491286
9	Star Wars: Episode VI - Return of the Jedi (1983)	0.502984
10	Lion King, The (1994)	0.504587
11	Aladdin (1992)	0.512340
12	Twelve Monkeys (a.k.a. 12 Monkeys) (1995)	0.513619
13	Groundhog Day (1993)	0.514719
14	Twister (1996)	0.517257
15	Men in Black (a.k.a. MIB) (1997)	0.517335

I think these are pretty good results as for ’Toy Story’ it is recommending similar movies with the same interest : Adventure, Animation, Children Film and Fantasy.

Method 2 :

This method makes use of ratings and movies dataset to make recommendations. It uses Pearson correlation coefficient between the user ratings of the input movie and all other movies.

Step wise step implementation:

1. During initialisation, all the required datasets are passed in the function.Then, we merge ratings and movies dataset to form a single dataset.
2. Then dropped the unnecessary columns like timestamp and genre in this case.
3. Now, a pivot table was created using user id as row, movie names as column and values as rating given by the user to a particular movie. It would look like

title	'Round Midnight (1986)	'Til There Was You (1997)	'Twas the Night Before Christmas (1974)	'burbs, The (1989)	(500) Days of Summer (2009)	*batteries not included (1987)	+1 (2013)	...And Justice for All (1979)	10 (1979)	10 Items or Less (2006)	...	Zorro, the Gay Blade (1981)	Zulu (1964)	[REC] (2007)	[REC]² (2009)	eXistenZ (1999)
userId																
1	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
2	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
3	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
4	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
5	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

5 rows × 9482 columns

4. Now we find a correlation matrix with a method as Person coefficient.
5. Then using this correlation matrix we find the most correlated movies and return the top 10 movies from the list.
6. Outcomes using this method for “Toy Story (1995)” with rating 5 are:

Top 10 recommended movies are(with correlation value):
Toy Story (1995) 2.5
Lion King, The (1994) 1.1110902661186373
Toy Story 2 (1999) 1.1082703136840713
Mission: Impossible (1996) 1.0604615648715883
Groundhog Day (1993) 1.0362646711370025
Aladdin (1992) 0.9820566614349058
Independence Day (a.k.a. ID4) (1996) 0.9816008144164033
Back to the Future (1985) 0.9766680419748562
Men in Black (a.k.a. MIB) (1997) 0.9526952561363445
Monsters, Inc. (2001) 0.9484369504565936

Item Based Collaborative Filtering :

Advantages :

- Personalized recommendations: The method is based on the user's rating history, so it can provide personalized recommendations that are tailored to the user's individual preferences.
- No need for explicit feature engineering: The method does not require any feature engineering or domain knowledge, as it relies solely on the user ratings to generate recommendations.

Disadvantages :

- Lack of diversity: Collaborative filtering can suffer from lack of diversity, as it tends to recommend similar items to those that a user has already rated highly. This can lead to a "filter bubble" effect where the user is only exposed to a narrow range of recommendations.
- Scalability issues: Collaborative filtering can become computationally expensive for large datasets, especially when the number of users and items is large.

Contributions:

We conducted extensive research on different movie recommendation techniques and encountered several challenges in handling the large dataset. The movie recommendation dataset we worked with contained millions of ratings and tags provided by users, making it challenging to load and process the data using traditional techniques. To overcome this challenge we used sparse matrix representation. One of the methods we used for movie recommendation was a simple system based on popularity, where the most popular movies were recommended to users. We also used collaborative filtering, where we predicted user ratings based on their similarity with other users and recommended movies based on these predictions.

Adarsh Raj Shrivastava (B21AI003):

- Implemented collaborative filtering method 2 (using correlation matrix)

Ankur (B21CS011):

- Implemented collaborative filtering method 1 (using kth nearest neighbor)