Personalized Movie Recommendation System

Milestone: Model Performance Evaluation and Interpretation

Group 82

Sai Revanth Reddy Boda
Bharath Raju Palla

+1 (781)-600-5040 (Sai Revanth Reddy Boda)
+1 (857)-396-8447 (Bharath Raju Palla)

boda.s@northeastern.edu
palla.b@northeastern.edu

Percentage of Effort Contributed by Student 1:_____50%_____

Percentage of Effort Contributed by Student 2:_____50%_____

Signature of Student 1: ____Sai Revanth Reddy Boda_____

Signature of Student 2: ____Bharath Raju Palla_____

Submission Date: _____7th April 2023_____

Problem Setting	3
Problem Definition	3
Data Sources	3
Data Description	3
Data Exploration Figure 1: Released for every year Figure 2: Top 10 movies with average rating Figure 3: Top 10 movies with highest volume of ratings Figure 4: Genres vs count of movies with respective genre Figure 5: Least 10 movies above average with volume of ratings Figure 6: Least 10 movies above average with average of ratings	4
Data Mining Tasks	7
Data Mining Models/Methods	8
Multidimensional Scaling Figure 7: Spread of ratings in a lower dimension after MDS Figure 8: Spread of ratings in a lower dimension after rating encoding Figure 9: Performance across different K values with results Figure 10: ROC Curve of SVC applied on encoded data Figure 11: Performance across different K values with results of different user	9
Performance Evaluation Table 1: Performance across different K values with an algorithm that uses Pearson correlation Table 2: Performance across different K values with an algorithm that uses Cosine similarity Table 3: Performance across different K values with an algorithm that uses NCFS correlation Table 4: Performance across different K values with an algorithm that uses Hybrid prediction Figure 12: Performance graph for Hybrid model Figure 13: Performance graph for Cosine model Figure 14: Performance graph for NCFS model Figure 15: Performance graph for Pearson model	15
Project Results Table 5: Recommendations given by model using Pearson correlation Table 6: Recommendations given by model using Pearson Similarity Table 7: Recommendations given by model using NCFS correlation Table 8: Recommendations given by Hybrid model	16
Impact of the project outcomes	18
References	18

Problem Setting:

A recommendation system is a systematic algorithm that learns to suggest items or content to users based on their preferences and behaviour. The objective of a recommendation system is to personalize the user experience by providing relevant and meaningful suggestions. Recommendation systems are generally used in various sectors including e-commerce, entertainment, news, and social media to refine users' interests which will individualize depicted content and improve engagement.

Problem Definition:

The primary objective of this project is to investigate the various filtering techniques used for recommendation systems and create an algorithm that is able to make a personalized movie recommendation to a user based on the viewing history, ratings, and genre preferences of a user.

Data Source:

The Movie recommendation dataset has been taken from Grouplens, an open-source repository for academic and development datasets. Dataset link: https://grouplens.org/datasets/movielens/latest/

Data Description:

The dataset contains two different files (. Csv's) which are used for further study are:

1. Movies.csv

   The dataset consists of 9743 rows and 3 columns describing the movie's title, a unique id (movieId), and respective genres.

2. Ratings.csv

   This dataset consists of 100837 rows and 4 columns which describe each user's history like what movies are watched and provided ratings and genres included. The Columns are named as userId, movieId, rating, and timestamp.

- Genres are a pipe-separated list with 19 unique genres.
- Users are provided with ids to anonymize user information.
- Ratings are made on an out-of-5 scale, with half-star increments (0.5 – 5.0).
- Timestamps represent seconds since midnight of January 1, 1970, UTC.

Data Exploration:

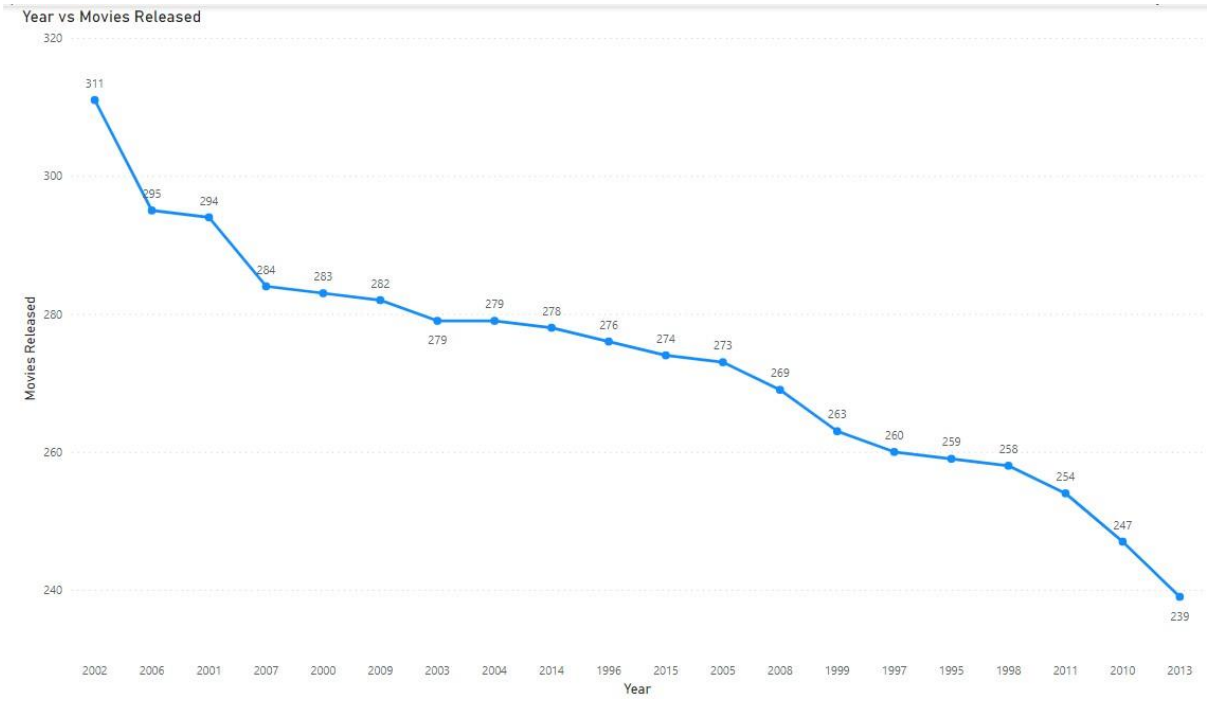


Figure 1: Released for every year

From the above graph, we can analyze the number of movies released in the corresponding year in the x-axis for the top 20 years.

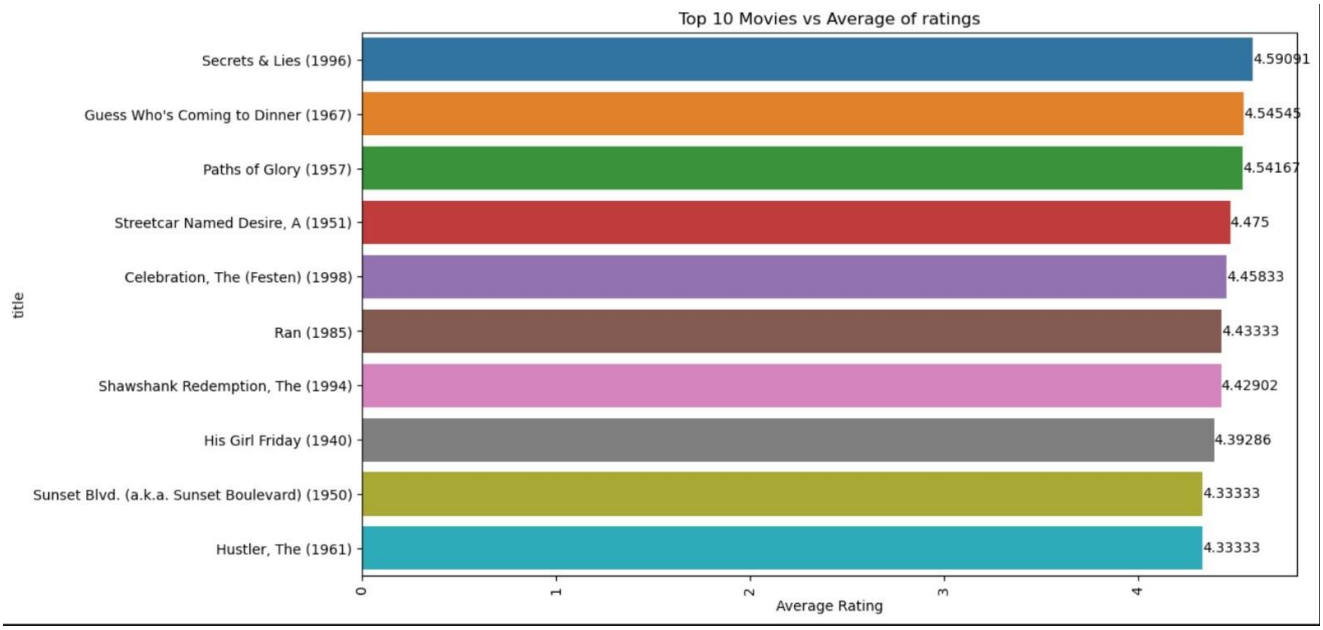


Figure 2: Top 10 movies with average rating

From the above graph, we can examine the average rating received for a movie for the top 10 average ratings.

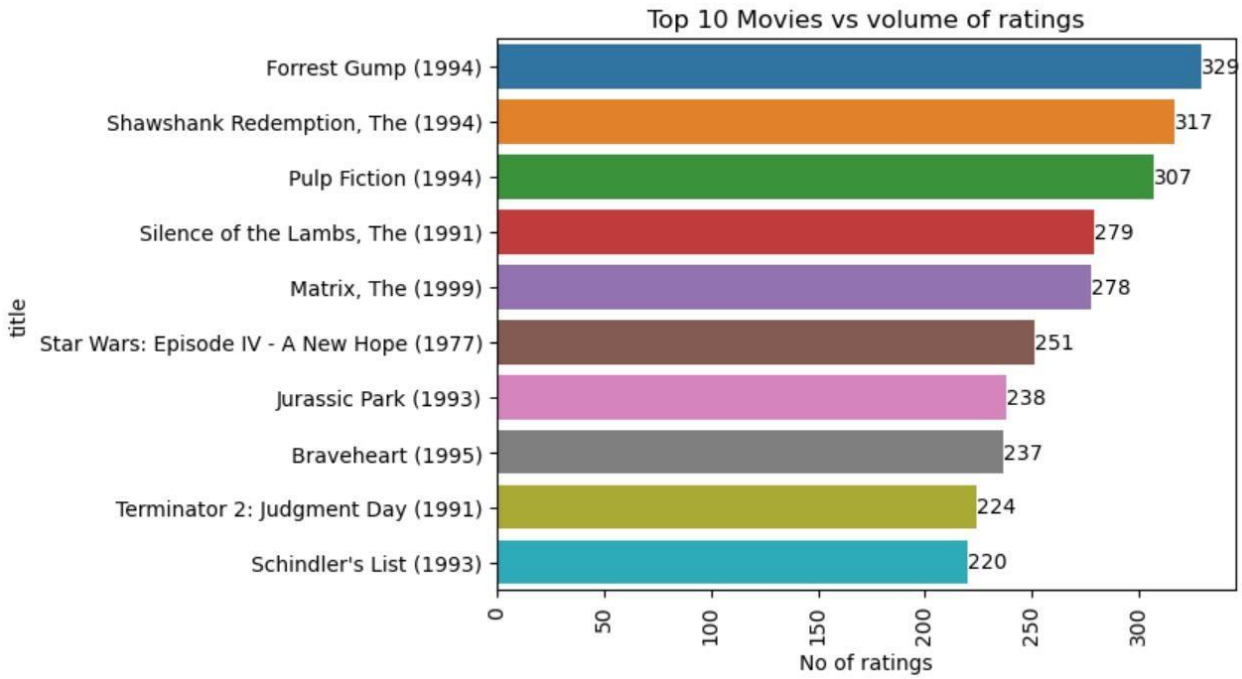


Figure 3: Top 10 movies with highest volume of ratings

From the above graph, we can examine the Number of ratings received for a movie. Displays the top 10 movies.

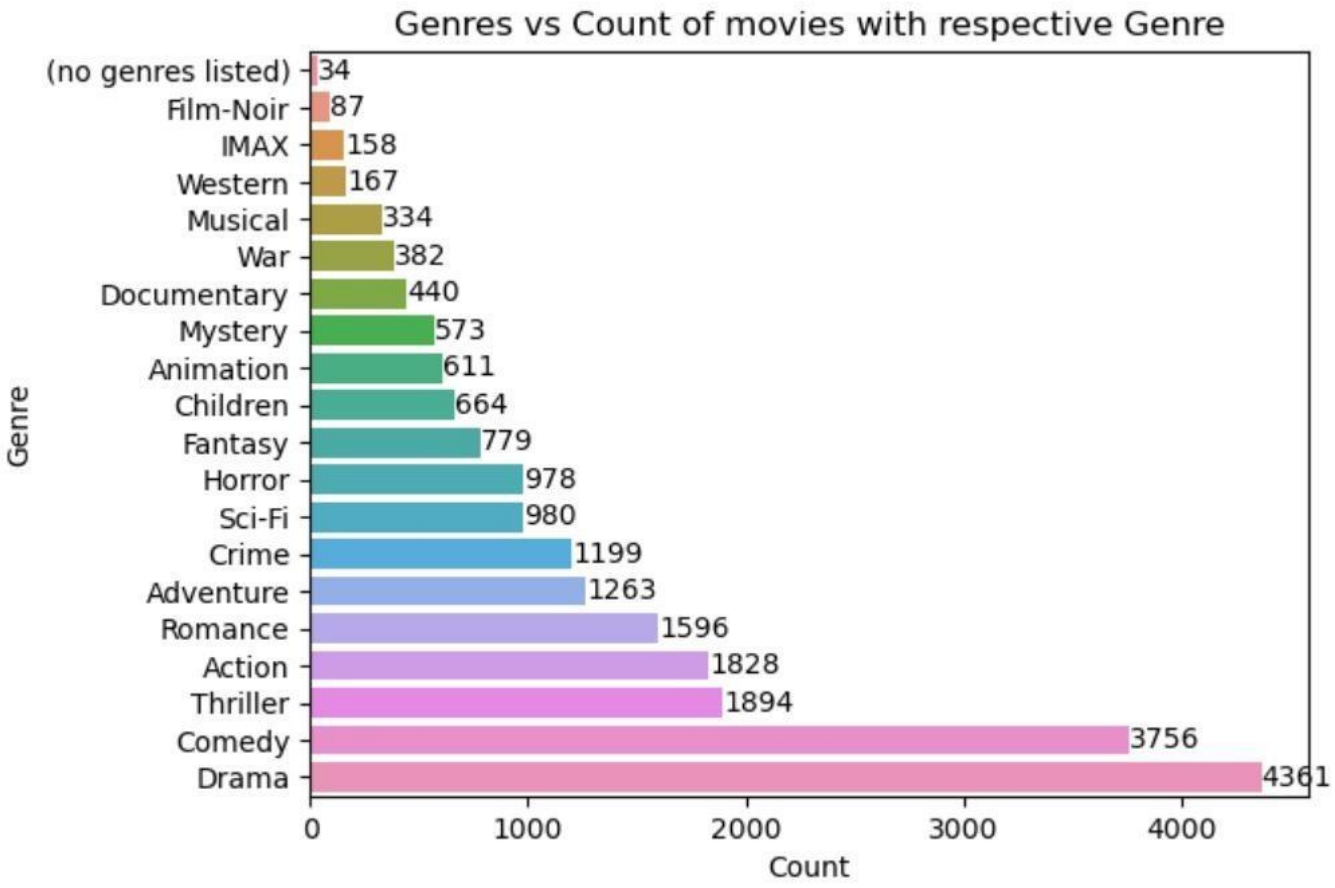


Figure 4: Genres vs count of movies with respective genre

From the above graph, we can examine the genre contribution for all movies.

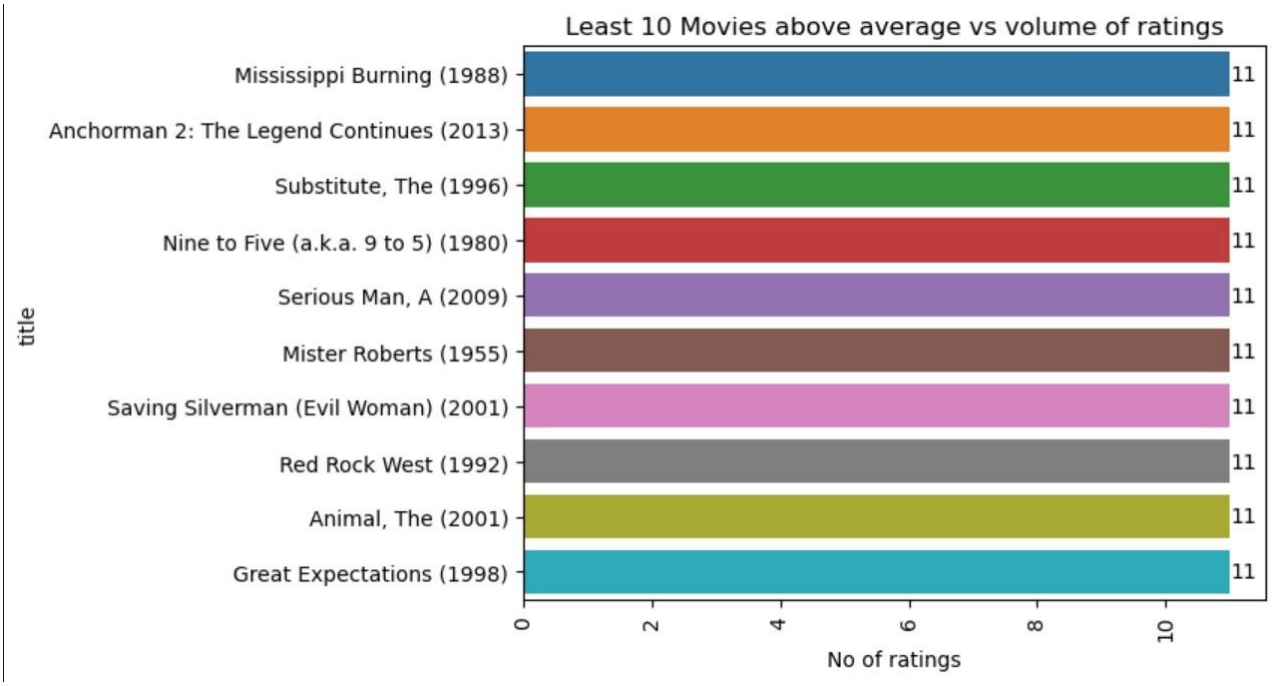


Figure 5: Least 10 movies above average with volume of ratings

From the above graph, we can examine the 10 movies recorded with the condition of an above-average number of people rating a movie (i.e; We only consider movies with a number of critiques greater than the average number of people rating a movie).

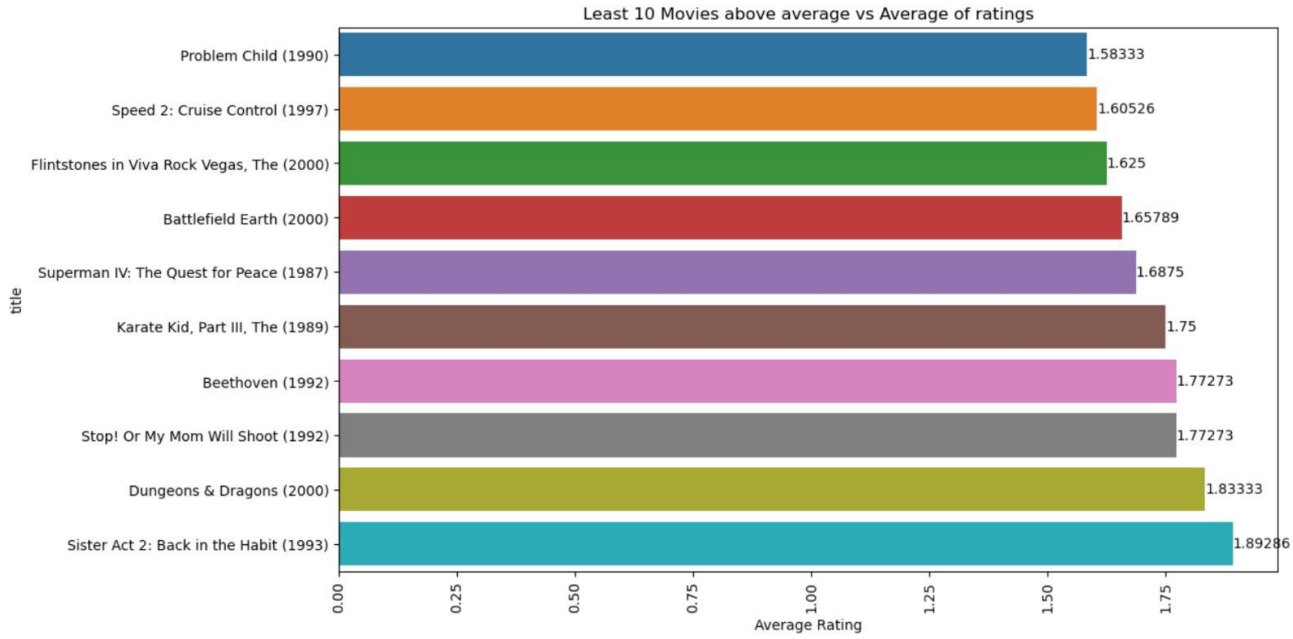


Figure 6: Least 10 movies above average with average of ratings

From the above graph, we can examine the 10 least-rated movies recorded with the condition of an above-average number of people rating a movie (i.e; We only consider movies with a number of critiques greater than the average number of people rating a movie).

Descriptive Statistics

- Our data contains 100836 ratings spread across 9742 movies. These data were given from the perspective of 610 users. Each user has rated multiple movies and with a minimum of 20 movies and a maximum of 2698 movies rated by User 414. The movies are spread across 22 unique genres which spread across all the movies. Among the movies present the genre “Drama” is famous among the users, and similarly “Film-noir” is least favored.
- We can observe from the above visualization that our dataset has more concentration of movies from the 90s than from the latest years.
- From the average ratings computed, we can see the top 10 placed well-liked movies by users among which the movie “Secrets and Lies” is the highest and “Forest Gump” movie has the most no of views given the assumption that all views have rated and none vice-versa.
- “The problem Child” is the least in ratings with a qualifying no of Critiques condition applied.

Data Mining Tasks:

There are several models that we are going to use for a movie recommendation system based on genres and user ratings.

1. **Content-based filtering:** This model uses the user’s past movie ratings and genres to recommend similar movies. It builds a user profile based on the genres and ratings of the movies that the user has watched and then recommends movies that have similar genres.

   We used KNN with L2-Norm to calculate the genre weightage per each user and recommend the movies and along with KNN we used Support Vector Machines (SVMs) to predict the relevance of a movie to a user based on the genres of the movie and the user’s ratings for similar movies.

2. **Collaborative filtering:** This model uses the ratings of other users who have similar movie preferences as the current user to recommend movies. It can be further classified into two types of user-based collaborative filtering and item-based collaborative filtering.

   User-User Collaborative Filtering: Here we find look alike users based on similarity and recommend movies which first user’s look-alike has chosen in past.

   It is used for efficiently computing the required information by forming user-rating matrix and performing similarity analysis on target user to predict their behaviour.

3. **Hybrid models:** These models combine multiple recommendation algorithms to improve the overall recommendation performance. A hybrid model is ensemble collaborative filtering and content-based filtering to get the best of both worlds.

Data Mining Models/Methods:

1. **Content-based filtering:**

   A common method for creating recommendation systems is content-based filtering, particularly around movie suggestions. The fundamental tenet of content-based filtering is that viewers will prefer films with comparable material to those they have previously appreciated.

   The primary idea behind content-based filtering is that before using this information to generate user suggestions, it is important to assess the characteristics and content of objects, such as movies. Here, when producing movie suggestions, we took the genre of a movie into account before recommending more movies with related subjects.

   The degree of resemblance between two films can be assessed using a variety of methodologies, including cosine similarity, Euclidean distance, and Jaccard similarity. The algorithm can identify the movies that have the most in common by employing these techniques. In our case we used Euclidean distance and measured similarity between movies.

   Once the algorithm has determined which movies are comparable to those that a user has previously liked, it can produce recommendations by suggesting those comparable films to the user. This strategy assumes that viewers will have a recurring affinity for particular types of content, such as genres or actors, and that films with comparable content will appeal to those people more frequently.

   One benefit of content-based filtering is that it can still produce recommendations for users who have not previously rated many movies. This is so that the system can determine the user's preferences from the content and other characteristics of the movies rather than just relying on explicit ratings.

   But one of the drawbacks of content-based filtering is that it can only suggest movies that are comparable to those that a user has previously liked or rated. Because of this, it might not be able to suggest movies that go against the user's typical preferences. In addition, since content-based filtering relies on finding correlations to previously liked movies, it may be ineffective at proposing items that are new or unfamiliar to a user.

   We used K-Nearest Neighbours (KNN) to predict the rating of a user based on the genre predictors. The idea behind this approach is to find the K movies that are most like the ones that the user has rated, based on the genres they belong to, and then use the ratings of these movies to predict the rating of the user for a new movie.

Multidimensional Scaling

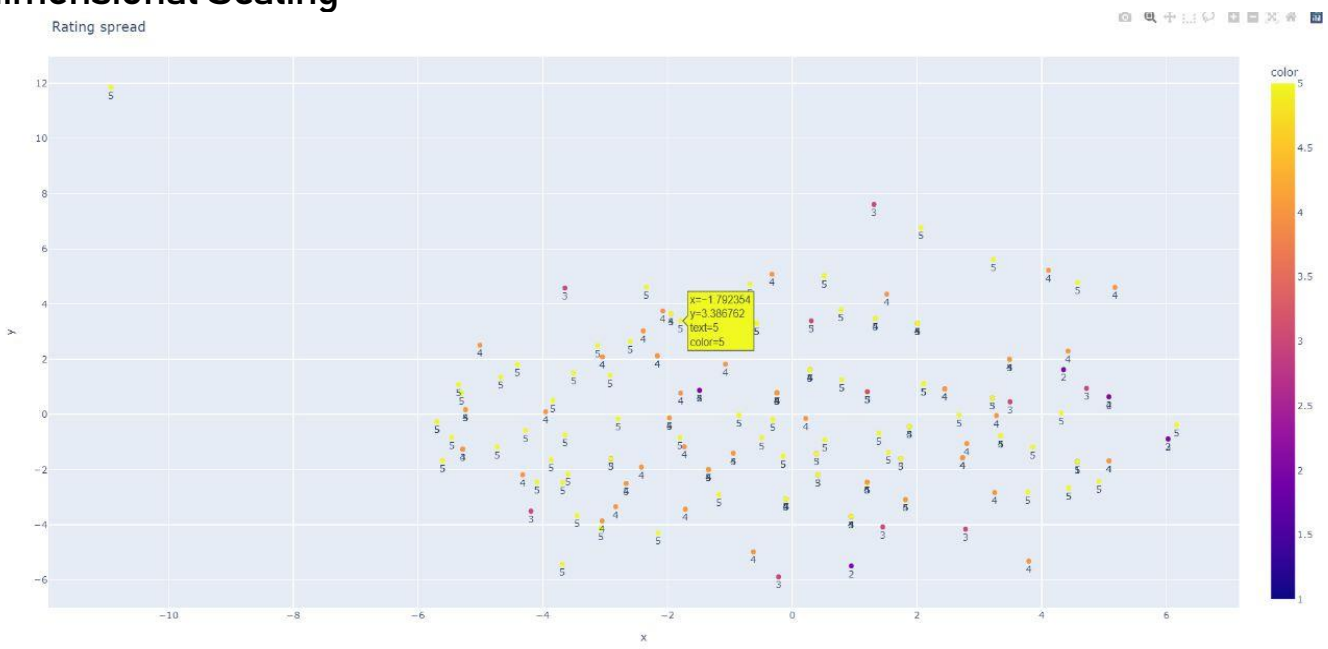


Figure 7: Spread of ratings in a lower dimension after MDS

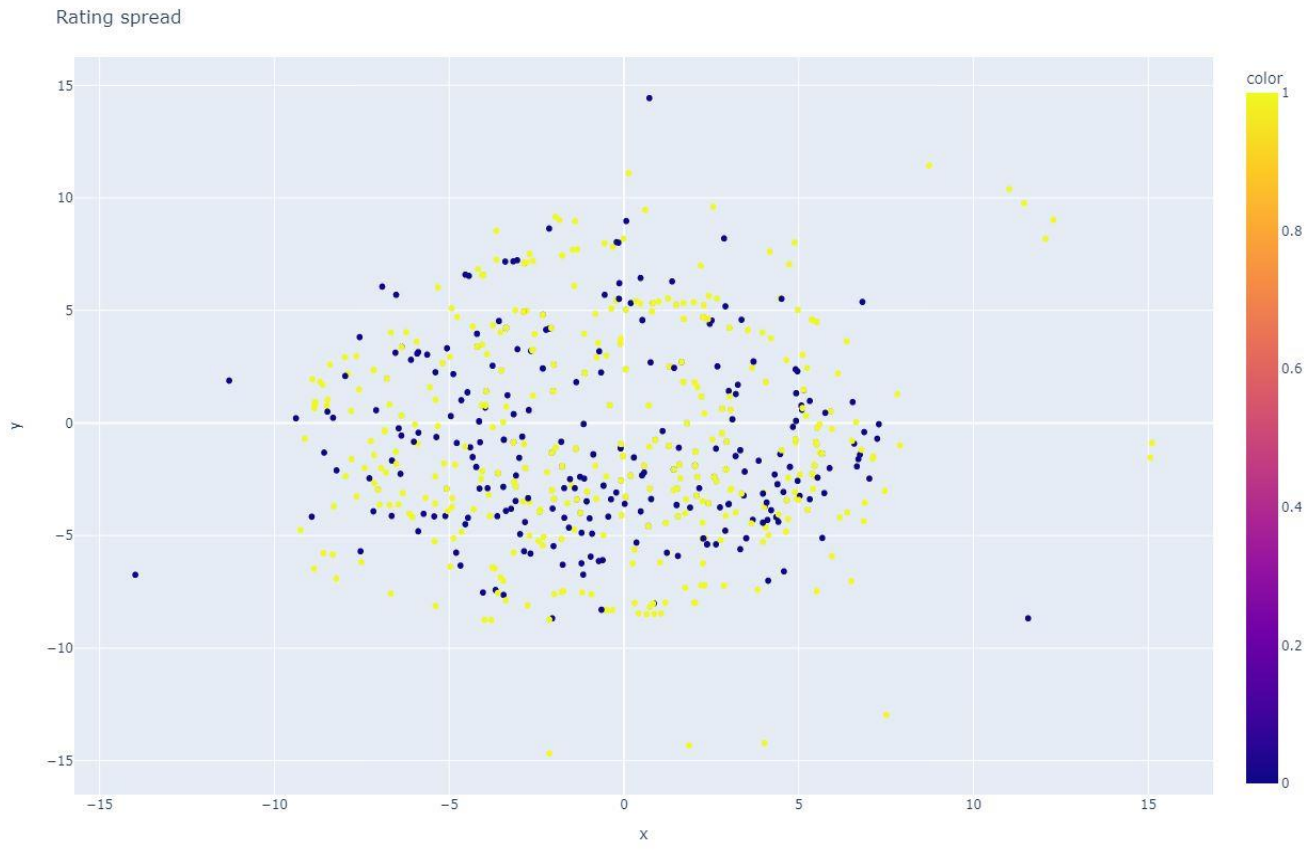


Figure 8: Spread of ratings in a lower dimension after rating encoding

We applied Multidimensional scaling (MDS) before KNN as it is a technique used to visualize high dimensional data in lower dimensions, typically 2 or 3. It can be used to analyse the similarity structure of the data and to identify groups or clusters of similar items.

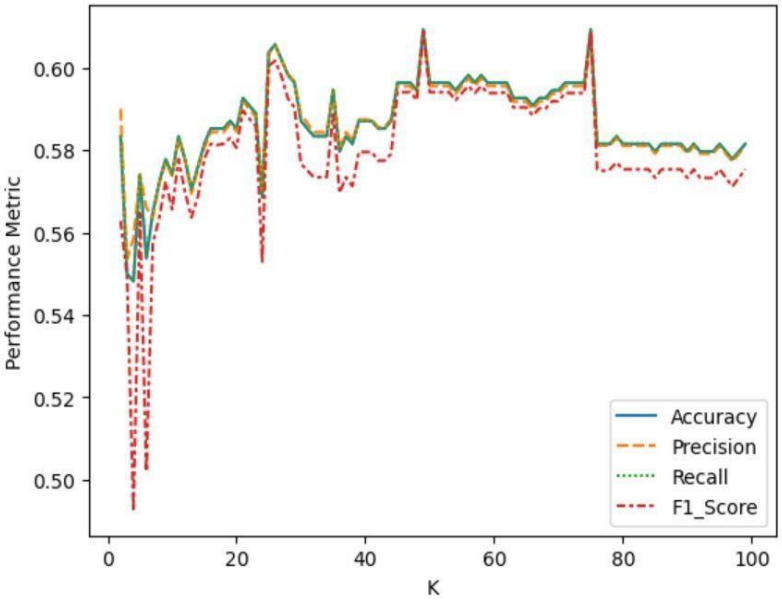
In the context of movie recommendation, MDS can be used to visualize the similarity between movies based on their genre and ratings. This can be useful for understanding the structure of the data and identifying potential patterns that may be useful for making recommendations.
From the clustering, we can observe that from above we can say that with only genre matrix it's not preferable to predict whether a user will like a movie or not, so we went on ahead and applied KNN with varied k values and SVC algorithm, computed the different classification metrics and ROCs to test our hypothesis.

Below illustrates the KNN model performance over different K values and from that we pick a good k value to get (N) number of predictions (movie recommendations) for a specific user.
We also used SVC to classify the user preferences and observed results not great compared to that of KNN.

But logically we are not expecting too much of a predictive ability from models that are being applied to genre vectors and predicting ratings, because it doesn't take into consideration the user interaction and similarity values.

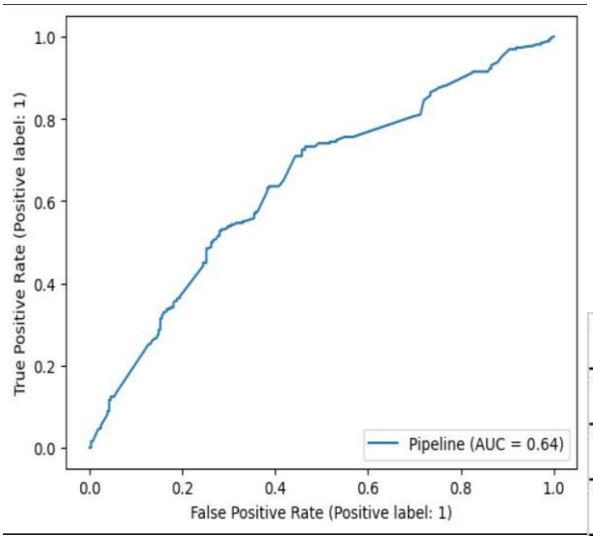
Hence, we proceed to Collaborative Filtering which considers user-user interaction and theoretically should produce better results than content-based filtering algorithms.

Below are the results from KNN and SVC for different user's



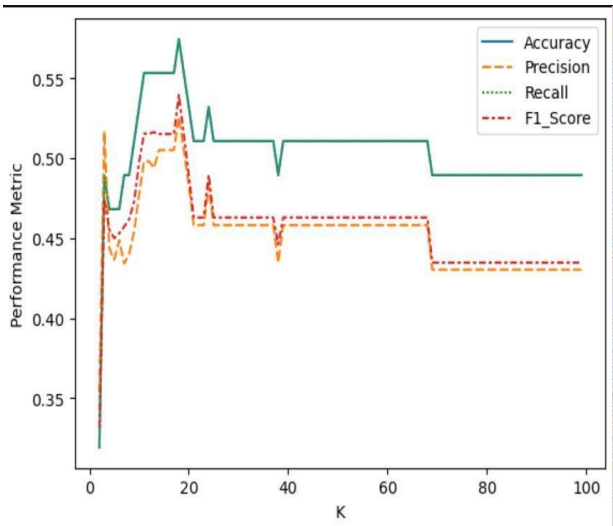
	Accuracy	Precision	Recall	F1 Score
49	0.60259	0.608729	0.609259	0.608741
75	0.609259	0.608671	0.609259	0.608612
26	0.605556	0.605436	0.605556	0.601598
25	0.603704	0.603264	0.603704	0.600459
27	0.601852	0.601607	0.601852	0.597857
...	...	...	...	...
7	0.564815	0.563599	0.564815	0.55733
24	0.568519	0.569825	0.568519	0.552956
3	0.55	0.553458	0.55	0.549406
6	0.553704	0.366033	0.553704	0.502056

Figure 9: Performance across different K values with results



Accuracy	0.872340425553191
Precision	0.760977818017202
Recall	0.872340425531914
F1_Score	0.812862669245648

Figure 10: ROC Curve of SVC applied on encoded data



K	Accuracy	Precision	Recall	F1_Score
18	0.574468	0.525777	0.574468	0.539412
13	0.553191	0.493823	0.553191	0.516154
11	0.553191	0.497954	0.553191	0.515299
12	0.553191	0.497954	0.553191	0.515299
17	0.553191	0.504997	0.553191	0.51504
...	...	...	...	..
72	0.489362	0.43026	0.489362	0.434697
71	0.489362	0.43026	0.489362	0.434697
70	489362	0.43026	0.489362	0.434697
69	489362	0.43026	0.489362	0.434697
2	0.319149	0.353546	0.319149	0.331389

Figure 11: Performance across different K values with results of different user

The low AUC value indicates that the classifier struggles to distinguish between positive and negative instances, indicating poor discriminative power. Higher AUC values generally indicate better model performance, with a lower limit of 0.5 representing random guessing. Our AUC value of 0.68 is below the desired level, which aligns with our earlier hypothesis that predicting user preferences based on genre matrix alone may be challenging.

Overall, while the KNN classifier demonstrates moderate performance, additional features and algorithms may be necessary to improve its accuracy in predicting user preferences based on genre information alone. The low AUC value suggests that the model's performance could be improved with further exploration and refinement.

2. Collaborative filtering:

Collaborative filtering's fundamental idea is that users will be more likely to enjoy films that other users who's like their tastes.

Collaborative filtering's main goal is to find user similarities based on their prior actions, such as the movies they have viewed and rated. The algorithm then makes recommendations for users based on the behaviour of other users who behave similarly to them using these commonalities.

General algorithm of user based collaborative filtering:

1. Identify users who are most similar to the user of interest (u)
    a. Comparing the preference of user u to the preference of all the other users (v)

2. Among the items that the user has not yet experienced or watched, recommend the ones that are most preferred by the neighbours of u.

We identify similarities between user's using different similarity indexes.

1. Pearson Similarity

$$Pearson\_Sim(u,v) = \frac{\sum_{p\in I_u\cap I_v}(r_{u,p}-\mu_u)\cdot(r_{v,p}-\mu_v)}{\sqrt{\sum_{p\in I_u\cap I_v}(r_{u,p}-\mu_u)^2}\cdot\sqrt{\sum_{p\in I_u\cap I_v}(r_{v,p}-\mu_v)^2}}$$

Where $\mathrm{r_{u,p}}$ represents the rating of active user u for item p and μ_u is the average rating given by user u. The value of Pearson_Sim (u, v) ranges between −1 and 1. The value 1 indicates the most similarity between users and −1 shows the difference between two users.

2. Cosine Similarity

$$Cos\_Sim(u,v) = \frac{\vec{r}_u\cdot\vec{r}_v}{||\vec{r}_u||\cdot||\vec{r}_v||}$$

$\vec{\mathrm{r}}_\mathrm{u}$ and $\vec{\mathrm{r}}_\mathrm{v}$ is the average rating value of user u and v respectively. $\mathrm{r_{u,p}}$ and $\mathrm{r_{v,p}}$ denotes the rating of item p by user u and v respectively. $\vec{\mathrm{r}}_\mathrm{u}$ and $\vec{\mathrm{r}}_\mathrm{v}$ is the vector of the user u and v rated respectively. The magnitude of vector is represented as $||\cdot||$.

3. URP Similarity

$$URP_{Sim(r_{u,p},r_{v,p})} = 1 - \frac{1}{1+exp(-|\mu_u-\mu_v|\cdot|\sigma_u-\sigma_v|)}$$

where μ_u and μ_v is the mean rating of user u and v respectively and σ_u and σ_v represents the standard variance of user u and v.

$$\sigma_u = \sqrt{\sum_{p \in I_u} \frac{(r_{u,p} - \overline{r_u})^2}{|I_u|}} \qquad \mu_u = \sum_{p \in I_u} \frac{r_{u,p}}{|I_u|}$$

$URP_{Sim(r_{u,p}, r_{v,p})}$ measures the preference of each user. Diferent users have diferent rating preferences. Some users prefer to give high ratings. On the other hand, some users tend to rate low value. In order to reflect this behaviour preference, here we use the mean and variance of the rating to model the user preference.

4. Modified Jaccard Similarity

$$JPSS_{Sim(u,v)} = PSS_{Sim(u,v)}.Jaccard'_{Sim(u,v)}$$

$$PSS\ (r_{u,p}, r_{v,p}) = \text{Proximity}(r_{u,p}, r_{v,p}) \times \text{Significance}(r_{u,p}, r_{v,p}) \times \text{Singularity}(r_{u,p}, r_{v,p})$$

$$Jaccard'_{Sim(u,v)} = \frac{|I_u \cap I_v|}{|I_u| \times |I_v|}$$

$|\mathrm{I_u} \cap \mathrm{I_v}|$ represents the number of similar items that users u and v have seen, and $|\mathrm{I_u}|$ indicates the number of items that the active user (u) and $|\mathrm{I_v}|$ indicates the number of items the other user (v) has rated. The $\mathrm{PSS}_{Sim(u,v)}$ measure is composed of three factors of similarities.

The Proximity factor calculates the absolute difference between two ratings. The second factor is Significance. We assume that the ratings are more significant if two ratings are more distant from the median rating. The third factor is called the Singularity. This factor represents how two ratings are different from other ratings The Proximity, Significance and Singularity are calculated through.

$$\text{Proximity}(r_{u,p}, r_{v,p}) = 1 - \frac{1}{1 + exp(-|r_{u,p} - r_{v,p}|)}$$

$$\text{Significance}(r_{u,p}, r_{v,p}) = 1 - \frac{1}{1 + exp(-|r_{u,p} - r_{med}| \cdot |r_{v,p} - r_{med}|)}$$

$$\text{Singularity}(r_{u,p}, r_{v,p}) = 1 - \frac{1}{1 + exp\left(-\left|\frac{r_{u,p} + r_{v,p}}{2} - \mu_p\right|\right)}$$

Where $\mathrm{r_{u,p}}$ is the rating of item p by user u and $\mathrm{r_{v,p}}$ is the rating of item p by user v. $\mathrm{r_{med}}$ is the median value in the rating scale.

5. NCFS similarity

$$\text{NCFS\_Sim(u,v)} = JPSS_{Sim(u,v)}.URP_{Sim(u,v)}$$

NCFS is a new user similarity measure to improve the recommendation and prediction performance when only few ratings are available for calculating the similarity for all users. This

similarity measure not only considers the local context information of user ratings, but also the global preference of user behaviour. The NHSM can overcome the drawback of the Pearson similarity measure when the rating matrix is sparse.

Prediction

$$NCFS\_Predict(u,p) = \mu_u + \frac{\sum_{j=1}^{m}\left(r_{v_j,p} - \mu_v\right)\cdot NCFS\_Sim(u,v_j)}{\sum_{j=1}^{m}|NCFS\_Sim(u,v_j)|}$$

$$Pearson\_Predict(u,p) = \mu_u + \frac{\sum_{j=1}^{m}\left(r_{v_j,p} - \mu_v\right)\cdot Pearson\_Sim(u,v_j)}{\sum_{j=1}^{m}|Pearson\_Sim(u,v_j)|}$$

$$Hybrid\_Predict(u,p) = \frac{NCFS\_Predict(u,p) + Pearson\_Predict(u,p)}{2}$$

The hybrid system calculates these ratings for all unobserved items by the active user and offers the Top 10 movies.

μ_{u} is the average ratings of active user and m is the number of neighbor users with the active user. $\mathrm{NCFS\_Sim}(\mathrm{u},\mathrm{v_j})$ and $\mathrm{Pearson\_Sim}(\mathrm{u},\mathrm{v_j})$ are the similarity degree of the active user u with the user $\mathrm{v_j}$. $\mathrm{r_{v_j,p}}$ is the rating of item p by user v and μ_{v} is the average ratings of user v.

There is possible that a certain similarity measure cannot predict a value, and in this condition, most systems use the average of the active users' ratings for a target item. Therefore, by combining these two similarity measures as a new method, if the system is not able to predict the amount of user preferences by k-nearest users of a certain similarity measure, the system can use the other k-nearest users from another similarity measure to predict them. $\mathrm{Hybrid\_Predict}(\mathrm{u},\mathrm{p})$ is a formula for combining user preferences.

Where u is an active user and p is an item. The NCFS_Predict (u, p) and the Pearson_Predict (u, p) are the predictions of item rating p. The output of the NCFS_Predict (u, p) and the Pearson_Predict(u, p) is between 1 and 5, and if each of the similarity measures is not able to predict the value of user preference, the NCFS_Predict (u, p) and the Pearson_Predict (u, p) return the average ratings of the active user. Finally, the hybrid_Predict (u, p) uses similarity degree to predict the user preferences accurately.

Performance Evaluation

Pearson

k	R2_score	MAE	RMSE
2	-0.0607	0.680995	0.84425
3	0.123996	0.657828	0.767233
4	0.247121	0.596395	0.711274
5	0.277989	0.565395	0.69654
6	0.293162	0.568817	0.689182
7	0.300553	0.572706	0.68557
8	0.321266	0.569199	0.675342
9	0.31185	0.566529	0.680011
10	0.314762	0.566773	0.67857
11	0.30418	0.564245	0.68379
12	0.323845	0.553819	0.674058
13	0.32691	0.553878	0.672529
14	0.316549	0.556372	0.677685
15	0.30876	0.561099	0.681536
16	0.321523	0.552196	0.675214
17	0.31615	0.5546	0.677883
18	0.313515	0.55474	0.679188
19	0.329546	0.548163	0.67121
20	0.340334	0.545483	0.665788
21	0.339346	0.546414	0.666287
22	0.342134	0.544729	0.664879
23	0.342387	0.546512	0.664751
24	0.349103	0.543693	0.661348
25	0.354028	0.541124	0.658842

Table 1: Performance across different K values with an algorithm that uses Pearson correlation

Cosine

k	R2_score	MAE	RMSE
2	0.066458	0.663768	0.792029
3	0.184449	0.618824	0.740286
4	0.120404	0.664573	0.768804
5	0.18665	0.60674	0.739287
6	0.195682	0.597603	0.73517
7	0.247822	0.580766	0.710942
8	0.250813	0.577481	0.709528
9	0.245135	0.579882	0.712211
10	0.276438	0.564559	0.697288
11	0.310261	0.538336	0.680795
12	0.317557	0.536749	0.677185
13	0.332737	0.527085	0.669611
14	0.334062	0.525669	0.668946
15	0.332364	0.528098	0.669798
16	0.339859	0.517663	0.666028
17	0.331292	0.521854	0.670336
18	0.335373	0.517316	0.668287
19	0.32828	0.524184	0.671844
20	0.321608	0.528504	0.675172
21	0.316536	0.524783	0.677691
22	0.317086	0.525319	0.677419
23	0.324666	0.524043	0.673649
24	0.323677	0.521369	0.674142
25	0.316671	0.526556	0.677625

Table 2: Performance across different K values with an algorithm that uses Cosine similarity

NCFS

k	R2_score	MAE	RMSE
2	0.123396	0.616227	0.767495
3	0.129706	0.611699	0.764728
4	0.127934	0.613118	0.765506
5	0.128074	0.61301	0.765445
6	0.128002	0.613065	0.765476
7	0.128025	0.613048	0.765466
8	0.128025	0.613048	0.765466
9	0.128025	0.613048	0.765466
10	0.128025	0.613048	0.765466
11	0.128025	0.613048	0.765466
12	0.128025	0.613048	0.765466
13	0.128025	0.613048	0.765466
14	0.128025	0.613048	0.765466
15	0.128025	0.613048	0.765466
16	0.128025	0.613048	0.765466
17	0.128025	0.613048	0.765466
18	0.128025	0.613048	0.765466
19	0.128025	0.613048	0.765466
20	0.128025	0.613048	0.765466
21	0.128025	0.613048	0.765466
22	0.128025	0.613048	0.765466
23	0.128025	0.613048	0.765466
24	0.128025	0.613048	0.765466
25	0.128025	0.613048	0.765466

Table 3: Performance across different K values with an algorithm that uses NCFS correlation

Hybrid

k	R2_score	MAE	RMSE
2	0.288722[illegible]	0.691343[illegible]	0.577910
3	0.368732[illegible]	0.651300[illegible]	0.548824
4	0.369403[illegible]	0.650954[illegible]	0.548845
5	0.353731[illegible]	0.658993[illegible]	0.558000
6	0.367697[illegible]	0.651833[illegible]	0.554101
7	0.368797[illegible]	0.651266[illegible]	0.552555
8	0.385692[illegible]	0.642491[illegible]	0.540726
9	0.368677[illegible]	0.651328[illegible]	0.549941
10	0.364709[illegible]	0.653372[illegible]	0.553738
11	0.360981[illegible]	0.655286[illegible]	0.552897
12	0.370495[illegible]	0.650390[illegible]	0.545106
13	0.378554[illegible]	0.646213[illegible]	0.543162
14	0.376107[illegible]	0.647484[illegible]	0.544993
15	0.374564[illegible]	0.648285[illegible]	0.543436
16	0.379713[illegible]	0.645610[illegible]	0.539239
17	0.379812[illegible]	0.645559[illegible]	0.537801
18	0.378522[illegible]	0.646230[illegible]	0.540968
19	0.384397[illegible]	0.643168[illegible]	0.536271
20	0.387602[illegible]	0.641491[illegible]	0.533741
21	0.388427[illegible]	0.641059[illegible]	0.533982
22	0.388674[illegible]	0.640930[illegible]	0.534365
23	0.388100[illegible]	0.641231[illegible]	0.536823
24	0.390545[illegible]	0.639948[illegible]	0.535604
25	0.392328[illegible]	0.639011[illegible]	0.534012

Table 4: Performance across different K values with an algorithm that uses Hybrid prediction

Hybrid

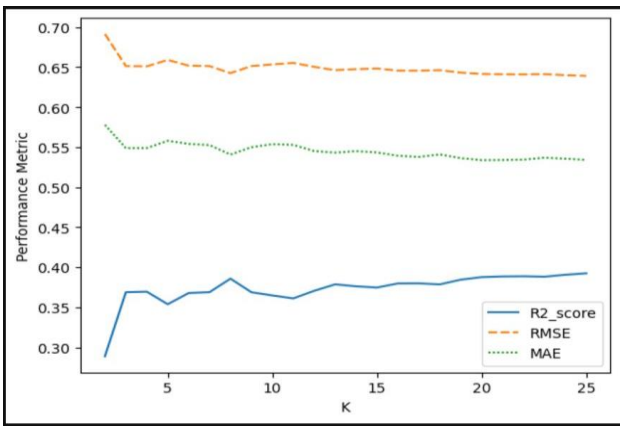


Figure 12: Performance graph for Hybrid model

Cosine

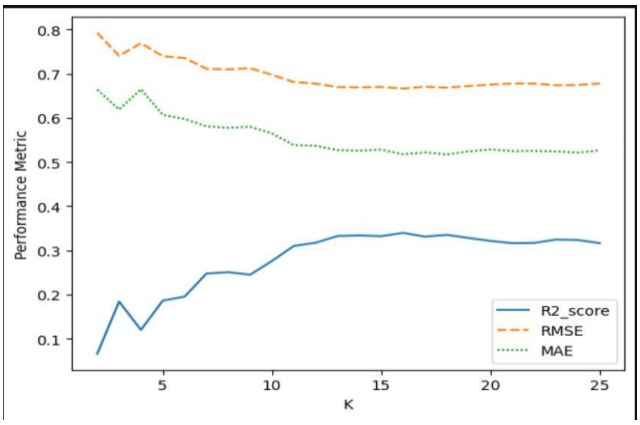


Figure 13: Performance graph for Cosine model

NCFS

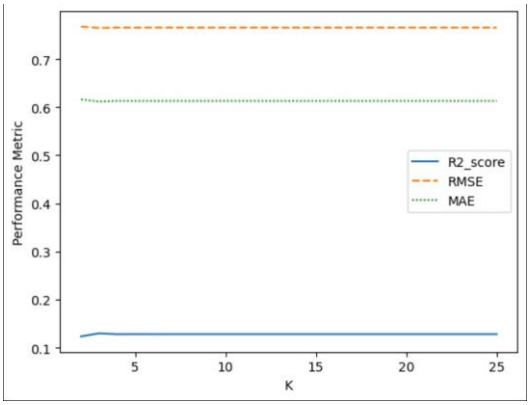


Figure 14: Performance graph for NCFS model

Pearson

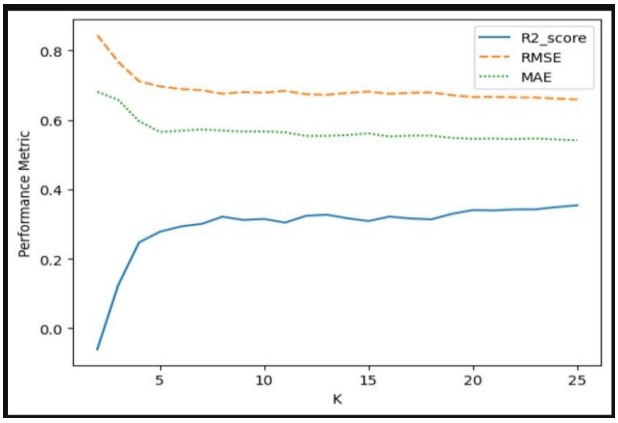


Figure 15: Performance graph for Pearson model

Project Results

Pearson

movieId	Predicted Rating	title
99	4.804944	Heidi Fleiss: Hollywood Madam (1995)
148	4.650781	Awfully Big Adventure, An (1995)
213	4.106285	Burnt by the Sun (Utomlyonnye solntsem) (1994)
246	4.189916	Hoop Dreams (1994)
29	3.856115	City of Lost Children, The (Cité des enfants p
187	4.47322	Party Girl (1995)
40	4.276777	Cry, the Beloved Country (1995)
58	3.745074	Postman, The (Postino, Il) (1994)
272	3.721237	Madness of King George, The (1994)
53	4.410432	Lamerica (1994)

Table 5: Recommendations given by model using Pearson correlation

Cosine

movieId	Predicted Rating	Title
99	5.000000	Heidi Fleiss: Hollywood Madam (1995)
148	5.000000	Awfully Big Adventure, An (1995)
53	5.000000	Lamerica (1994)
246	4.601392	Hoop Dreams (1994)
123	4.384624	Chungking Express (Chung Hing sam lam) (1994)
40	4.282417	Cry, the Beloved Country (1995)
28	4.224232	Persuasion (1995)
29	4.201077	City of Lost Children, The (Cité des enfants p...
199	4.121869	Umbrellas of Cherbourg, The (Parapluies de Che...
187	4.115994	Party Girl (1995)

Table 6: Recommendations given by model using Pearson Similarity

NCFS

movieId	Predicted Rating	Title
85	4.884004	Angels and Insects (1995)
99	4.884004	Heidi Fleiss: Hollywood Madam (1995)
53	4.884004	Lamerica (1994)
194	4.703400	Smoke (1995)
148	4.650781	Awfully Big Adventure, An (1995)
40	4.521789	Cry, the Beloved Country (1995)
28	4.493001	Persuasion (1995)
199	4.493001	Umbrellas of Cherbourg, The (Parapluies de Che...
283	4.305108	New Jersey Drive (1995)
70	4.249907	From Dusk Till Dawn (1996)

Table 7: Recommendations given by model using NCFS correlation

Hybrid

movieId	Hybrid Rating	Title
99	4.804944	Heidi Fleiss: Hollywood Madam (1995)
148	4.650781	Awfully Big Adventure, An (1995)
213	4.106285	Burnt by the Sun (Utomlyonnye solntsem) (1994)
246	4.189916	Hoop Dreams (1994)
29	3.856115	City of Lost Children, The (Cité des enfants p...
187	4.047322	Party Girl (1995)
40	4.276777	Cry, the Beloved Country (1995)
58	3.745074	Postman, The (Postino, Il) (1994)
272	3.721237	Madness of King George, The (1994)
53	4.410432	Lamerica (1994)

Table 8: Recommendations given by Hybrid model

Impact of Project Outcome

A movie recommendation system has several potential outcomes that can significantly impact the movie industry. Firstly, it can greatly enhance the user experience by providing personalized recommendations based on their viewing history, ratings, and preferences, leading to increased user satisfaction and engagement. Secondly, if integrated into a subscription-based streaming service, it can increase user retention and generate more revenue for the service provider. Thirdly, it can assist users in discovering new and lesser-known content, resulting in increased consumption and visibility for content creators. Lastly, the system's data collection can provide valuable insights into user behavior, aiding content creators and service providers in making better-informed decisions regarding content acquisition, production, and marketing. Overall, a movie recommendation system can have a significant impact on user experience, revenue generation, and data insights.

References

[1] A new user similarity model to improve the accuracy of collaborative filtering Haifeng Liu, Zheng Hu, Ahmad Mian, Hui Tian, Xuzhen Zhu