Movie Recommender

Alessandra Tartarotti
Dublin City University
Dublin, Ireland
alessandra.sonegotartarotti2@mail.dcu.ie

Andrey Totev
Dublin City University
Dublin, Ireland
andrey.totev2@mail.dcu.ie

Daniel Higgins
Dublin City University
Dublin, Ireland
daniel.higgins23@mail.dcu.ie

Rajesh Dande
Dublin City University
Dublin, Ireland
rajesh.dande2@mail.dcu.ie

***Abstract*—Movies are one of the main sources of entertainment for people. Movie streaming services are constantly looking for ways to capture their users' attention and keep them engaged by providing highly personalized recommendations. Movie recommender systems are information filtering tools that predict relevant movies that a user might be interested in. In 2006, Netflix released a dataset containing 100 million anonymous movie ratings and challenged the computer science communities to develop a system that could beat the accuracy of its commercial recommendation system, Cinematch™ by 10%. This challenge inspired data enthusiasts around the globe to find the best collaborative filtering (CF) algorithm for predicting user ratings on movies.**

In this work, we employ matrix factorization (MF) to implement a baseline neural network architecture for collaborative filtering. Furthermore, we explore several different methods of supplying movie attributes (year, genre, popularity score) to the recommender and evaluate the performance of the derived hybrid model.

Keywords — Movie Recommendation System, collaborative filtering, matrix factorization

I. Introduction

The personalisation of products and services helps business' gain insights into user preferences so that they can offer tailored experiences [1]. These personalised recommendations then allow users to make advised choices in their day-to-day activities and purchases. A successful recommendation system can significantly improve the revenue of e-commerce companies or facilitate the interaction of users in online communities. This makes recommender systems a central part of websites, such as Netflix, Amazon and LinkedIn. The goal of a recommender system is to generate meaningful recommendations to a collection of users for items or products that might interest them. The design of such recommendation engines depends on the domain and the particular characteristics of the data available [2]. For example, Netflix users will frequently provide ratings on a 1-5 rating scale and these interactions between users and items are recorded. Additionally, the system may have to access item-specific and user-specific profile information such as demographics and product descriptions. Collaborative filtering systems analyse historical interactions alone, while content-based systems are based on profile attributes. Hybrid techniques attempt to combine both of these designs. The class of collaborative filtering algorithms is divided into two sub-categories, called memory based and model based. The memory-based approach uses user rating data to compute the similarity between users or items and model-based approaches try to predict user ratings of unrated items using different data mining techniques [3]. Netflix reports that its recommender system influences users' choice for 80% of watched movies, and the recommender system contributes more than $1B per year to its business [4]. Other studies of recommender systems also consistently report increases of several percentages in page views, time spent on page and revenue [5].

The Netflix Prize [1] was a large-scale data mining competition held by Netflix for the best collaborative filtering algorithm for predicting user ratings on movies, based on a training set of more than 100 million ratings given by over 480,000 users to 17,770 movies. Each training data point consists of a quadruple (user, movie, date, rating) where rating is an integer from 1 to 5. In this paper, we introduce a method to answer the Netflix prize problem using matrix factorization. The end goal of this research is to predict the user ratings based on the Netflix prize problem dataset, and then incorporate external movie features like genres, and popularity from The Movie Database (TMDb) and analyze the impacts on the predictive performance of a selection of neural network models.

The rest of this paper structure is as follows. Section 2 provides a critical analysis of the related work in this area. Section 3 demonstrates our approach for predicting user ratings for movies they have not yet viewed. Section 4 describes the data understanding and data preparation stages. Section 5 illustrates the Modelling stage. Section 6 evaluates the different model variants using RMSE - the official Netflix Prize competition metric. Section 7 summarises our findings and discusses the scope for future work.

II. Related Work

Recommender systems are based on a variety of approaches such as content based [6],[7], collaborative filtering [8,9,10,13,15] and hybrid [11,12]. Moreover, movie recommendation systems are centred on collaborative filtering and clustering to tackle the similarities between the users and items and perform recommendations. In movie recommendation systems the user is asked to rate the movies which the user has already seen, then these ratings are applied to recommend other movies that the user has not yet seen by utilizing collaborative filtering techniques. Memory-based CF [13] searches for the nearest neighbour in the user space for an active user and dynamically recommends movies. The limitations related to this method are computation complexity

[1] https://www.kaggle.com/netflix-inc/netflix-prize-data

and data sparsity. Some authors [14] have reduced the complexity and bottleneck issues by using item-based CF techniques as this method can decrease the time of computation, as well as deliver rationally correct predictions and accuracy. Model-based CF [15] are developed using machine learning algorithms to predict user's ratings of unrated items and can address the issues of sparsity and scalability.

In CF, the information gathered on the behaviour of users is stored in a user model or user profile. This user profile can be created by explicit information from a user, or implicitly, e.g., recording the observed pages or watched videos. In case of a new visitor without any recorded profile, an issue called cold-start problem appears. This also occurs when a new movie is added to the field, since it has not been assigned yet to any user and cannot be recommended to anyone. Content-based approaches solve this issue by calculating the similarity between the new and already stored items [16]. Halder. S. et al. [17] also showed cold-start and sparsity problems can be minimised using movie swarms data mining technique based on genres of movie. Another issue of CF which is relevant to the Netflix Prize is the long tail problem, whereby the distribution includes many values that are far away from the mean value. Reddy M. et al. [18] research shows movies that never got an above-average rating do not have significant contribution in movie recommendations and can be ignored for calculations.

The Netflix Prize competition had 44,014 submissions in total over a three-year period which used a variety of model-based techniques such as Clustering [16], Matrix Factorization [19,20,21] and ensemble methods [8]. The most popular clustering algorithm used was KNN, which is an unsupervised data mining tool that is used for partitioning a given dataset into groups based on a similarity metric. T Hong et al [16] showed the potential of KNN to perform good predictions with RMSE scores of 1.0 on the Netflix Prize set but the accuracy improvement was only marginal using default values and clustering. However, in other studies of KNN [22], it is shown that the RMSE can be reduced significantly by applying weighting and bias techniques. From the model building perspective, matrix factorization (MF) is one of the most impactful models exploited in collaborative filtering. MF decomposes the user-item interaction matrix into the product of two lower dimensionality rectangular matrices and will be used as a key element of this research. G. Takacs et al. [19] proposed several matrix factorization approaches with improved accuracy prediction on the Netflix Prize dataset. The biased regularised simultaneous MF model (BRISMF) performed best with RMSE of 0.8939. The bias extension speeds up the training phase and yields a more accurate model with better generalization performance. In similar research, Paterek et al [20] introduces the use of bias features and reported RMSE results of 0.9070 for their biased regularised MF. Additionally, Koren et al [21] provide a detailed description of their alternating least squares approach to MF using ridge regression to assure non-negative weights and obtained RMSE scores of 0.9167.

III. Process Flow

The methodology used in this project management process is Cross Industry Standard Process for Data Mining (CRISP-DM). The application of using this is explained below through each section. The business objectives of this study is to predict the rating a user would give a movie they have not yet rated. This objective is achieved by applying collaborative filtering methods to the Netflix Prize problem and understanding how different methods affect the performance of the model. For this research, an external movie dataset was integrated to increase the number of features available. The aim is to analyze the impacts of external integrated movie features on the model performance. The working goal is to minimize the RMSE value.

The next step is the data understanding phase, where all the features are analysed with verification of the quality of data and the integration of external features. The Netflix Prize dataset contains 100 million ratings, given by over 480,000 users to 17,770 movies. The features include movieID, ratings (scale 1-5), userID and date of rating.

The original movie dataset from The Netflix Prize contains the attributes movieID, title and year. To extend it with external features, some experiments were done with different movie datasets. The chosen dataset was TMDb, a free and open movie and TV database, as it is the dataset with the most movie titles in common with the original movie dataset.

IV. Data Preprocessing

From the descriptive statistics of the Netflix dataset, it's evident the data is complete and there are no null values or outliers present. During the data preprocessing stage, a new column "weekday" was added to the Netflix dataset for further analysis, but no further changes were made. Since there are very few features present in the Netflix dataset, incorporating additional features was an important aspect of this research.

TMDb provides a free API[2], which has available methods on their movies, tv shows and actors. Firstly, by using this API, through the endpoint /search/movie and specifying the movie title as a parameter, it was possible to retrieve the following attributes: TMDd Id, genre ids, media type (TV show or movie), vote average and popularity[3], which is a metric created by TMDb influenced by the number of votes and views the movie had, the number of users who marked the movie as "favourite", or added it to their "watch list" and release date. As a result, 16531 movies from the Netflix Prize dataset could be found on The Movie Database API.

Secondly, using the identifier of the movie in the TMDb database obtained on the previous step, it was possible to retrieve more attributes such as overview, tagline, crew, cast and genre description through the endpoint /movie/{movie_id} and using TMDb Id as parameter.

In total 93% of the movies were found on TMDb, among them 937 movies have the genre missing completely at Random. In order to make the data easily accepted as input for our model, one-hot encoding was used for the genre data on the augmented movie dataset, resulting in 19 new columns.

[2] https://developers.themoviedb.org/3/getting-started/introduction

[3] https://developers.themoviedb.org/3/getting-started/popularity

A. *Exploratory Data Analysis*

Performing univariate analysis on the ratings variable shows the distribution of ratings in the Netflix training dataset (Fig. 1). The dataset follows a normal distribution which is slightly skewed towards the higher values. Users tend to give ratings of 3 or 4 for most watched movies, with the global average rating being 3.58.

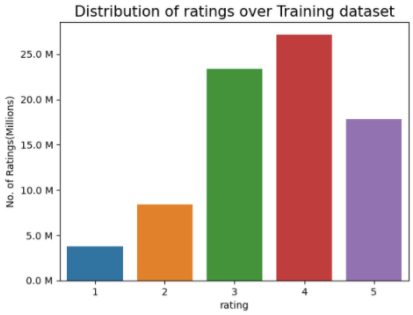

Fig. 1. Bar chart showing the distribution of genres

The average user rated over 200 movies and the average movie was rated by over 5000 users in the training set. However, there is wide variance in the data, with some movies in the training set having as few as 3 ratings and one user rating over 17,000 movies. Fig. 2 shows the distribution of ratings in the training dataset. The ratings per movie are heavily skewed, with some popular movies being rated by hundred thousand of users. However, most of the movies (90%) have thousands of ratings.

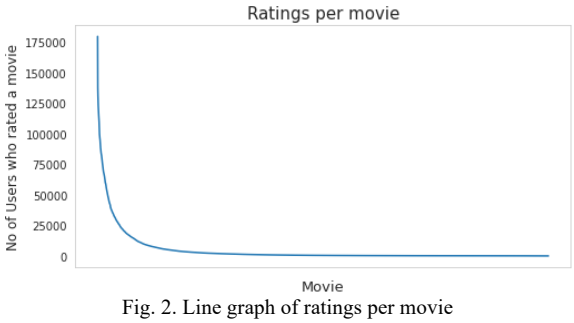

Fig. 2. Line graph of ratings per movie

Among the 19 genres used by TMDb to categorize the movies the predominant genre is drama, followed by comedy, as shown in Fig. 3.

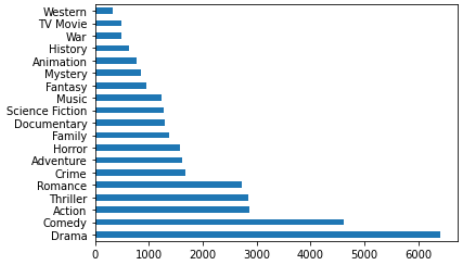

Fig. 3. Bar chart showing the distribution of genres

Analysing the one-hot encoded genre representation in Fig. 4, the highest correlation between genres is 0.3, which happens for Adventure and Action, Animation and Family. We also attempted to cluster the movies based on TMDb "voting score" and "popularity". Using the Elbow Method, we estimated the optimal number of clusters - the graph on the Fig. 5 indicates the number is 3.

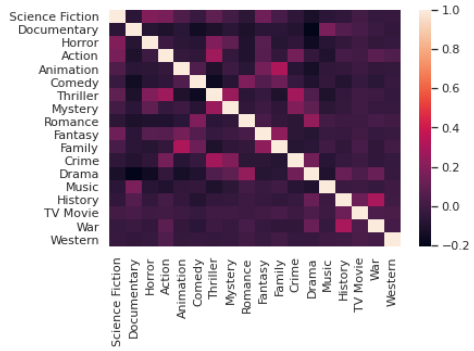

Fig. 4. Heatmap showing correlation between the genres

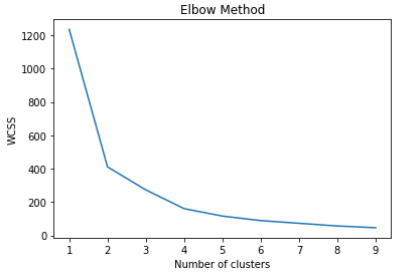

Fig. 5. Line chart representing number of clusters

The algorithm K-Means was applied to the movie dataset using the optimal number of clusters. The Fig 6 shows the low inter-cluster distance.

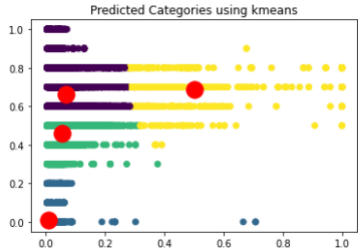

Fig. 6. Plot showing predicted categories of movies based on popularity (X axis) and average vote (Y axis)

B. *Matrices*

For 17,770 movies and 480,189 users in the Netflix dataset, there is a maximum of 17,770*480,189 which is 8,532,958,530 ratings. But the total number of ratings in the Netflix dataset is 100,480,507. The user and movie corresponding utility matrix (Table I) has only 1.18% of total values which makes it very sparse. Creating embeddings and applying matrix factorization is a good approach to handle this problem based on existing literature [19, 20, 21].

TABLE I. SAMPLE RATINGS MATRIX V ON MOVIES BY USERS

	Movie1	Movie2	Movie3	Movie4	Movie5
User1	4	3	3	4	2
User2	3	0	3	0	3
User3	3	3	0	3	2
User4	4	0	2	3	0

We can process this kind of rating matrix through matrix factorization using latent factors to predict the missing ratings. Matrix factorization is an optimal value problem, where we break a matrix into two smaller matrices with smaller dimensions. The vectors in these matrices are commonly known as (vector-space) embeddings. So, for a given n*m matrix V (Table I), the matrix W (n * r) (Table II) and the matrix H (r * m) (Table III) are calculated iteratively by making sure the product of the two matrices is as close to the original matrix as possible (Table IV).

In this example, class1, class2 and class3 are the latent factors.

TABLE II. SAMPLE LATENT FACTOR - USER MATRIX W

	Class1	Class2	Class3
User1	0.5	0.7	0.3
User2	0.2	0.6	0.3
User3	0.3	0.5	0.2
User4	0.1	0.2	0

TABLE III. SAMPLE LATENT FACTOR - MOVIE MATRIX H

	Movie1	Movie2	Movie3	Movie4	Movie5
Class1	0.9	0.5	0	0.4	0.2
Class2	0.4	0.3	0.6	0.5	0.1
Class3	0.2	0.7	0.1	0.6	0.7

TABLE IV. SAMPLE MATRIX PRODUCT OF USER MATRIX AND MOVIE MATRIX

	Movie1	Movie2	Movie3	Movie4	Movie5
User1	4	3	3	4	2
User2	3	**2**	3	**3**	3
User3	3	3	**2**	3	2
User4	4	**3**	2	3	**3**

This kind of latent factor embeddings and matrix factorization approach can be applied to our utility matrix with 8,532,958,530 possible ratings in our dataset and predict the missing ratings. We used 50 dimensions for the user and movie embeddings in our model.

V. MODELING

Matrix Factorization was used to try to build on existing work [23] and compare the performance when incorporating features from the TMDb dataset. A matrix factorization neural network model was used as our baseline model (Fig. 7). The model takes the user ID and the movie ID as input, generates two 50-dimensional vector space embeddings, feeds the dot product of the two vectors into a dense layer and outputs the rating prediction. To speed up the training process, a relatively large batch size (512) was used in combination with an increased learning rate (0.005) to compensate for its impact on the training loss.

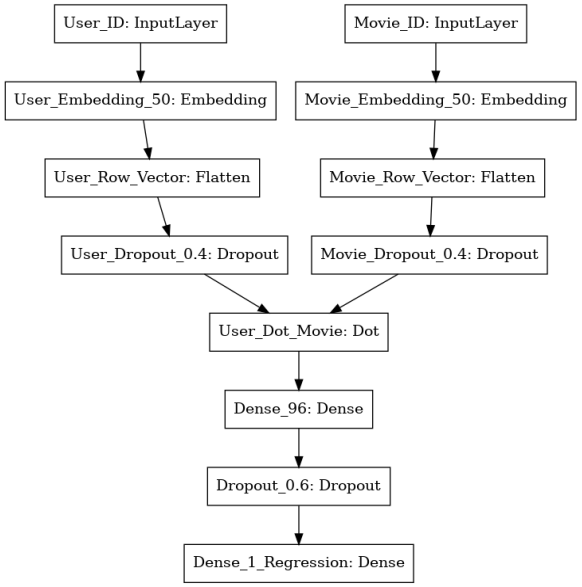

Fig. 7. Baseline model - user ratings with no extra data

To measure the performance impact of movie attributes on the predictions, a number of models were developed which use the movie vector information about genre, year and popularity on the TMDB website.

For the genre attribute, a 19-dimensional one-hot encoded genre representation was input, followed by an 8-dimensional dense layer which is concatenated to the movie embedding (Fig. 8). For the year attribute, an 8-dimensional embedding is generated which is also concatenated to the movie embedding and fed into a 50-dimensional dense layer. The popularity attribute is directly fed into an 8-dimensional dense layer which is then combined with the movie embedding similarly to genre and year (Fig. 9).

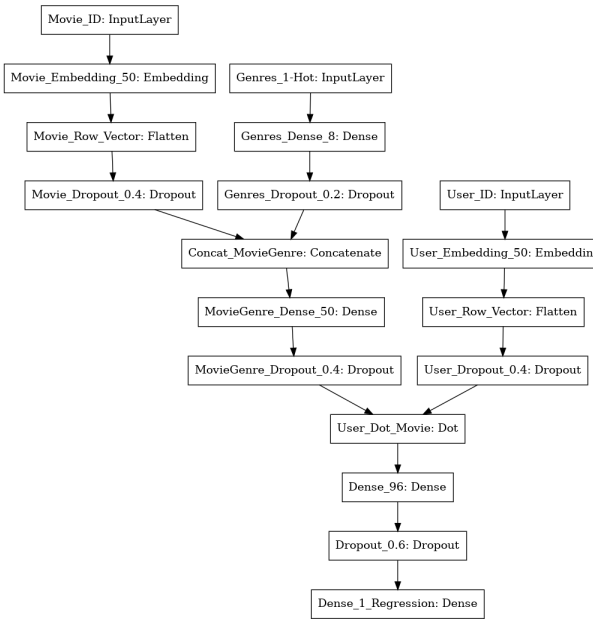

Fig. 8. User ratings (Movie + Genre)

Fig.9. User ratings (Movie + Year, Genre, Popularity)

In the last experiment of infusing the movie embedding with content data, all three attributes were concatenated, before feeding them into a 16-dimensional dense layer which is again combined with the movie embedding.

In the final experiment, a "merge" layer that multiplies the user embedding, the movie embedding, and all content data (year, genre, popularity) (Fig. 10) is created. The multiplication layer feeds into a sequence of two dense layers (128 and 64 elements) before regression. The goal of the experiment is to check whether the previous experiments were not flawed by the fact that the content data was applied only to the movie branch of the model tree.

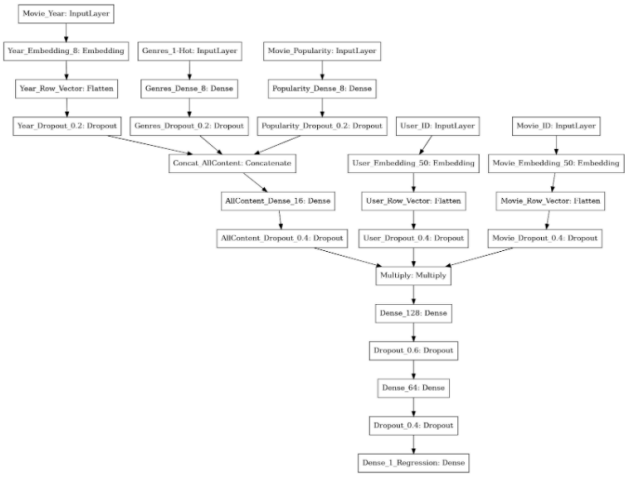

Fig. 10 User ratings (Movie x Year, Genre, Popularity x User).

VI. Evaluation and Results

We achieved RMSE score of 1.1776 with our baseline model for movie rating prediction on the Netflix Prize dataset and evaluated the impact of content attributes on the performance of a collaborative filtering model trained on Netflix Prize dataset.

We also evaluated the model against the MovieLens 100k dataset [24]. Results are listed in the table V.

TABLE V. RESULTS

	Baseline Model (no extra attributes)	***Movie + Extra Attributes***	***Movie x Extra Attributes . x User***
RMSE - Netflix Prize (with TMDB movie attributes)	1.1776	1.1775	1.1775
RMSE - MovieLens	1.2700	1.2697	1.2701

There is no clear winner for the best performing model. In all experiments the model gradually started to overfit the data after training on approximately 70,000 data points. None of the hybrid models was able to yield noticeable performance improvements.

The behaviour of the final architecture was pretty similar to the rest of the models which in a way confirms our conclusions about the (lack of) gain we are getting by applying content information in our models.

VII. Conclusion and Future Work

The experiment results confirmed we could not identify a neural network architecture that can derive any noticeable gains from the content attributes used in this research. Our conclusion is that when vast amounts of training data are available for a user base, it will likely outperform any 'generally available' information about the content.

The future work includes explaining the reason the movie attributes have no effect on the predictions and proposing new ways of enriching the model with content data.

The implementation of this work can be accessed at https://github.com/atotev/ca683-2021-dara.

References

[1] https://www.sitecore.com/knowledge-center/digital-marketing-resources/what-is-personalization-and-how-to-approach-it

[2] Melville, P. and Sindhwani, V., 2010. Recommender systems. Encyclopedia of machine learning, 1, pp.829-838.

[3] https://towardsdatascience.com/various-implementations-of-collaborative-filtering-100385c6dfe0

[4] Gomez-Uribe, C.A. and Hunt, N. 2016. The netflix recommender system: Algorithms, business value, and innovation. ACM Transactions on Management Information Systems (TMIS). 6, 4 (2016), 13.

[5] Belluf, T., Xavier, L. and Giglio, R. 2012. Case Study on the Business Value Impact of Personalized Recommendations on a Large Online Retailer. Proceedings of the Sixth ACM Conference on Recommender Systems (Dublin, Ireland, 2012), 277–280

[6] R. Singla, S. Gupta, A. Gupta and D. K. Vishwakarma, "FLEX: A Content Based Movie Recommender," 2020 International Conference for Emerging Technology (INCET), Belgaum, India, 2020, pp. 1-4. doi: 10.1109/INCET49848.2020.9154163

[7] H. Chen, Y. Wu, M. Hor and C. Tang, "Fully content-based movie recommender system with feature extraction using neural network," 2017 International Conference on Machine Learning and Cybernetics (ICMLC), Ningbo, China, 2017, pp. 504-509. doi: 10.1109/ICMLC.2017.8108968

[8] P. Venil, G. Vinodhini and R. Suban, "Performance Evaluation of Ensemble based Collaborative Filtering Recommender System," 2019 IEEE International Conference on System, Computation, Automation

and Networking (ICSCAN), Pondicherry, India, 2019, pp. 1-5. doi: 10.1109/ICSCAN.2019.8878777

[9] M. Gupta, A. Thakkar, Aashish, V. Gupta and D. P. S. Rathore, "Movie Recommender System Using Collaborative Filtering," 2020 International Conference on Electronics and Sustainable Communication Systems (ICESC), Coimbatore, India, 2020, pp. 415-420. doi: 10.1109/ICESC48915.2020.9155879

[10] A. Pal, P. Parhi and M. Aggarwal, "An improved content based collaborative filtering algorithm for movie recommendations," 2017 Tenth International Conference on Contemporary Computing (IC3), Noida, India, 2017, pp. 1-3. doi: 10.1109/IC3.2017.8284357

[11] K. Funakoshi and T. Ohguro, "A content-based collaborative recommender system with detailed use of evaluations," KES'2000. Fourth International Conference on Knowledge-Based Intelligent Engineering Systems and Allied Technologies. Proceedings (Cat. No.00TH8516), Brighton, UK, 2000, pp. 253-256 vol.1.doi: 10.1109/KES.2000.885805

[12] S. Sharma, A. Sharma, Y. Sharma and M. Bhatia, "Recommender system using hybrid approach," 2016 International Conference on Computing, Communication and Automation (ICCCA), Greater Noida, India, 2016, pp. 219-223. doi: 10.1109/CCAA.2016.7813722

[13] S. Surekha, "A Personalized Recommendation System using Memory-Based Collaborative Filtering Algorithm," 2018 International Conference on Current Trends towards Converging Technologies (ICCTCT), Coimbatore, India, 2018, pp. 1-6. doi: 10.1109/ICCTCT.2018.8551104

[14] P. Su and H. Ye, "An Item Based Collaborative Filtering Recommendation Algorithm Using Rough Set Prediction," 2009 International Joint Conference on Artificial Intelligence, Hainan, China, 2009, pp. 308-311. doi: 10.1109/JCAI.2009.155

[15] T. Chang and W. Hsiao, "Model-based collaborative filtering to handle data reliability and ordinal data scale," 2011 Eighth International Conference on Fuzzy Systems and Knowledge Discovery (FSKD), Shanghai, China, 2011, pp. 2065-2069. doi: 10.1109/FSKD.2011.6019879

[16] http://cs229.stanford.edu/proj2006/HongTsamis-KNNForNetflix.pdf

[17] S. Halder, M. Samiullah, A. M. J. Sarkar and Y. Lee, "Movie swarm: Information mining technique for movie recommendation system," 2012 7th International Conference on Electrical and Computer Engineering, Dhaka, Bangladesh, 2012, pp. 462-465, doi: 10.1109/ICECE.2012.6471587.

[18] M. M. Reddy, R. S. Kanmani and B. Surendiran, "Analysis of Movie Recommendation Systems; with and without considering the low rated movies," 2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE), Vellore, India, 2020, pp. 1-4, doi: 10.1109/ic-ETITE47903.2020.453.

[19] G. Takács, I. Pilászy, B. Németh and D. Tikk, "Investigation of Various Matrix Factorization Methods for Large Recommender Systems," 2008 IEEE International Conference on Data Mining Workshops, Pisa, Italy, 2008, pp. 553-562. doi: 10.1109/ICDMW.2008.86

[20] A. Paterek. Improving regularized singular value decomposition for collaborative filtering. In Proc. of KDD Cup Workshop at SIGKDD'07, 13 ACM Int. Conf. on Knowledge Discovery and Data Mining, pages 39-42, San Jose, CA, USA, 2007.

[21] R. M. Bell and Y. Koren. Scalable Collaborative Filtering with Jointly Derived Neighborhood Interpolation Weights. In Proc of. ICDM, IEEE International Conference on Data Mining, 2007

[22] A. Dai, Y. Xia and Y. Gui, "A self-learning collaborative filtering algorithm in recommendation system," 2017 3rd IEEE International Conference on Computer and Communications (ICCC), Chengdu, China, 2017, pp. 2195-2199, doi: 10.1109/CompComm.2017.8322926.

[23] https://www.kaggle.com/rajmehra03/cf-based-recsys-by-low-rank-matrix-factorization

[24] F. Maxwell Harper and Joseph A. Konstan. 2015. The MovieLens Datasets: History and Context. ACM Transactions on Interactive Intelligent Systems (TiiS) 5, 4, Article 19 (December 2015), 19 pages. DOI=http://dx.doi.org/10.1145/2827872