A Comparative Study in Movie Recommendation System

Sharanya Banerjee
1219403952
sbaner67@asu.edu

Hari Murugan Ravindran
1222324688
hravind1@asu.edu

Pooja Bihani
1219789363
pbihani@asu.edu

Arushi Gaur
1219396022
agaur17@asu.edu

Swarali Vinayak Chine
1222583687
schine@asu.edu

Radhika Kulkarani
1222165776
rkulka16@asu.edu

***Abstract*—In this Project, We have chosen to build a Movie Recommendation System which will recommend user movies based on their preference. It has to be built with extreme precision on which movie users will watch next so that they can be retained on the platform for some more time. It will be based on two of the most popular algorithms, namely Matrix Factorization and KNN. We are doing thorough and detailed research on these algorithms and modeling the above-proposed System. We are also doing a comparative study on these systems to conclude which algorithm delivers the best recommendation to the user. The System is built to provide the utmost positive user experience. Finally, we have also evaluated and presented a comparison of the results delivered by these algorithms based on some common metrics like Precision, Root Mean Squared Error and Recall.**

***Index Terms*—Recommendation System, Matrix Factorization, KNN, Comparative Study.**

I. Introduction

OTT (over-the-top) is a means of providing television and film content over the internet at the request and to suit the requirements of the individual consumer. In 2022, revenue from OTT Video is expected to reach US $281.70 billion. By 2026, the number of users in this industry is estimated to reach 3,930.3 million. By 2026, user penetration will have increased to 49.9% from 43.1% in 2022. In Our Past decade, we explored new content only through word of mouth or out of our research and interest. But in this digital age, the users are provided with an abundance of content to explore. Psychology says that when a person is provided with so many choices it is the same as not being provided with an option. Users will find it difficult to choose a movie or series to watch. Thus to ease users' experience and guide users with some form of aid Recommendation systems were built. These Systems at the beginning will ask users to provide it with some type of content that they have already watched and liked very much. From that content, the recommendation system will adapt and recommend users with different kinds of content that users might like. There are many kinds of algorithms these recommendation systems may use to provide recommendations to the users. They are broadly classified into two major categories: Content-Based Systems and Collaborative Filtering. Content-Based Systems make use of a feature of a particular item and recommend similar items. On the other hand, Collaborative Filtering makes use of the actions of users to recommend similar items. In our system, we have used two algorithms: Matrix Factorization and KNN. We have also finally evaluated them using common metrics.

II. Problem Statement

Innovation is needed when there arises friction in user experience. In the domain of content delivery companies, they are generating an immense amount of content to accommodate users who all have unique kinds of tastes. It is as much a curse for the user as it is a blessing to them. Now they are in a situation of content overload. But for providing the ultimate efficiency in the System there is another important task of delivering the right kind of content to the right user. The recommendation System comes to the content provider's aid in this situation. It adapts itself based on the user's preference and changes its structure. Earlier users who are fond of one kind of content need to search a lot of resources to find a content of similar nature. Nowadays recommendation systems will help users to find similar kinds of content in a wide variety of languages. These systems are employed in different kinds of domains such as videos, music, products and dresses, etc. These kinds of recommendation systems will not only be helpful for users but also for the companies to earn more by providing suitable content for users they need so they are retained in the platform for the maximum time which results in maximum revenue. The next major problem is which machine learning algorithm to choose for the recommendation system. Since the system can be modeled either using features of the particular item or using the actions of the user. So in this project, we are going to do a comparative study of two algorithms and choose the effective one out of them.

III. Related Works

A. Recommendation System in E-Commerce Websites: A Graph-Based Approached

[1]. In this paper, they discussed the possible impact recommendation system could bring on the e-commerce business. In this, we can use the recommendation system in three major types namely content-based, collaborative and hybrid. It states that a content-based system will provide content based on

user preference like it suggests novels based on the chosen genre. Also if it is collaborative it recommends content based on similar users' approaches. It also talks in-depth about the limitations we have in the current systems. Here they also did a comparative study on some of the famous e-commerce sites like Amazon, eBay, Flipkart, Snapdeal and Paytm. They finally conclude that systems which make use of the graph-based approach were found to be effective.

B. News Recommendation Based on Collaborative Semantic Topic Models and Recommendation Adjustment

[2]. We are receiving so much news every day. But we are interested in following news on certain kinds of topics. It will be very effective to use a recommendation system in this field. But the major problem with this is that it is difficult to categorize the news. Here they are proposing a collaborative semantic topic model to predict the user's preference with a unique hybrid approach. It combines Matrix Factorization with articles semantic latent topics derived from word embedding and Latent Dirichlet Allocation. This approach provides the user with news content to the user who is most likely to consume it.

C. Application research on personalized recommendation in distance education

[3]. Distance Education has always been the best use of online education. It results in the exponential growth of educational resources. But users are puzzled by the immense amount of educational content. Users will find it difficult to obtain information due to the huge amount of resources. So it is necessary to give personalized recommendations to the learners. It provides users recommendations based on content filtering and collaborative filtering. It discusses the whole framework which is suitable for the recommendation system.

D. Help me find a job: A graph-based approach for job recommendation at a scale

[4]. In this paper, they have discussed the possibility of employing the recommendation system in the job searching domain. Usually, people with good skills find it difficult to approach appropriate companies. Similarly, Recruiters also find it difficult to employ people according to the company's requirements. All these existing systems will focus only on the content of the resumes and the job descriptions but they suffer from rigidity and lack of implicit relations that uncover user behavior. Also, they don't address the scalability of the real world and the problem of cold start. It addresses the problem by harnessing the power of deep learning.

IV. Dataset Description

We are going to use MovieLens 20M Dataset for our Movie Recommendation System. This dataset describes 5-star rating and free-text tagging activity from MovieLens, a movie recommendation service. It contains 20000263 ratings and 465564 tag applications across 27278 movies. These data were created by 138493 users between January 09, 1995 and March 31, 2015. Users were selected at random for inclusion. All selected users had rated at least 20 movies and each user was represented by an id, and no other information was provided. The data are contained in six files, genome-scores.csv, genome-tags.csv, links.csv, movies.csv, ratings.csv and tags.csv.

- The tag.csv mainly contains tags suitable for movies by Users. The columns in tag.csv are userId, movieId, tag and timestamp.
- The rating.csv contains ratings for movies by Users. The ratings range from 0.5 to 5 with 0.5 increments. The columns in rating.csv are userId, movieId, rating and timestamp.
- The movie.csv mainly contains movie titles with suitable genres for those movies. The columns in movie.csv are movieId, title and genres.
- The link.csv contains identifiers in an external movie database like IMDB. The columns in link.csv are movieId, imdbId and tmbdId.
- The genome_scores.csv contains movie-tags relevance data. The columns in genome_scores.csv are movieId, tagId and relevance.
- The genome_tags.csv contains tag descriptions. The columns in genome_tags.csv are tagId and tag.

For the preprocessing steps, we took a subset of the total data, as the training with 20 million records was taking too much time. We primarily trained our model using the movie lens data with hundred thousand records and one million records. We checked for null/missing values in the movie's dataset. Finally we analyzed the dataset and removed the timestamp column as it was leading to fewer RMSE values.

V. System Architecture and Algorithms

Our movie recommendation system architecture consists of three main components as shown in Figure 1 . First, the data is preprocessed and fed to our training algorithm, which uses collaborative filtering techniques to come up with the prediction of ratings for the movies without a rating. After the prediction, the top 10 movies with the best ratings for each user are passed onto the User Interface. Finally, the user interface takes as input a userid and gives the recommendation for the user. In the training phase, we also measure the performance of the models using Precision, Recall and RMSE measures. We are storing the recommendations using the different algorithms in files beforehand so that the frontend can parse the files whenever needed to increase efficiency and reduce latency in large scale systems. [7]

For faster processing, we are storing images the recommendations from the two algorithms in files for the users so that system is fast and efficient. When we request the recommendations from UI, the data is retrieved from these files. Also, to make it visually appealing, we have added the image posters in the directory to parse.

Collaborative filtering methods based on users use information from other users to give a prediction for a particular user, for example, to give recommendations for a particular

user, we can use information from users similar to them. We use the following collaborative filtering algorithms for our recommendation system.

For splitting the data into training and testing data, we use the K-Fold technique. K-Fold is a validation technique in which we split the data into k-subsets and the holdout method is repeated k-times where each of the k subsets are used as test set and other k-1 subsets are used for the training purpose. Then the average error from all these k trials is computed , which is more reliable as compared to other methods.

Software Details: Google Collaborator: To write python code of the back-end. Front-end: HTML, CSS, Javascript Back-end: Python 3.x Python Libraries:

1) google colab
2) scikit-learn
3) scikit-surprise
4) numpy
5) collections
6) flask
7) pandas

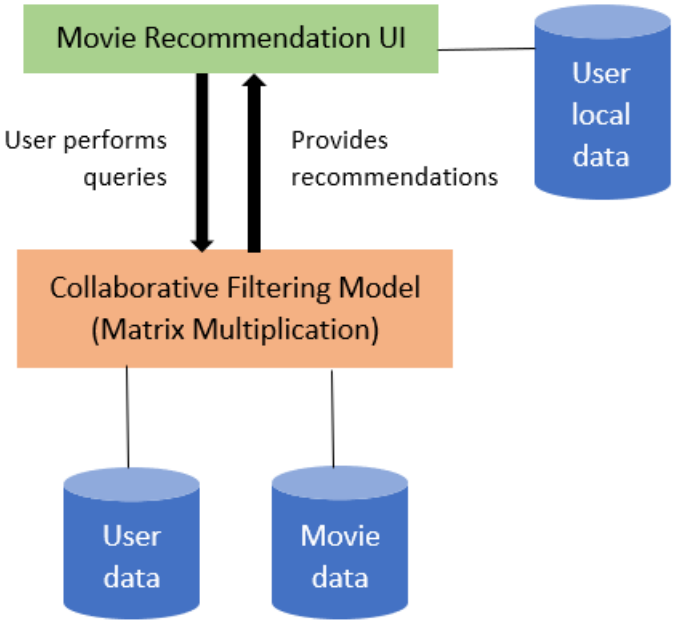

Fig. 1. System Architecture

A. Matrix Factorization

Matrix factorization is a class of collaborative filtering algorithms [5] used in recommender systems. They work by decomposing the user-item interaction matrix into the product of two lower dimension rectangular matrices. The user-item interaction matrix is decomposed into the product of two lower dimensionality rectangular matrices by matrix factorization algorithms. Users and items are the two main components of a recommender system. Assume there are m users and n items. The purpose of our recommendation system is to create a m x n matrix (also known as the utility matrix) that contains each user-item pair's rating (or preference). By taking this approach we can get a representation of the latent features and thus improve the recommendations. We tried many different types of algorithms and libraries but we found that the matrix factorization methods provided by the surprise library performed the best. For Dimensionality Reduction, we used Singular Value Decomposition (SVD) as it performed better (in terms of accuracy and computation time) than the others.

SVD: The user-item interaction matrix is factored into a row matrix and a column matrix, with the row having user info and the column with the item info or vice versa. The predicted rating is given by $r_{ui} = q_i^T p_u$ (with q and p being the matrices and u and i being user and item respectively) and to estimate the unknown ratings the equation is minimized using techniques such as stochastic gradient descent.

Cold Start Problem: In the beginning when the users are new to the platform, the system has no prior knowledge about the likes and dislikes of the user and is not able to recommend movies. Hence, the system is unable to develop any relationships between people and movies. That is, the system experiences a cold start problem. There are multiple techniques to resolve this issue. When a new user joins, the system can make a few random suggestions and ask the us In this, r to rate the movies. Based on the ratings, the system can then infer knowledge and utilize this data to present more suggestions. Another popular way of handling this issue is to use matrix factorization. The user and item interactions are depicted as the product of two matrices. In the first matrix, each user is assigned a row, and each movie is assigned a column in the second matrix; this set is called as latent factor. Latent factors are rows or columns associated with a specific person or user. There are no connected latent factors when a new item is introduced. Each movie has particular qualities associated with it. A function can be created that estimates the relevant movie latent components given the features. We can make a reasonable recommendation using feature mapping.

B. KNN Based Recommendation

KNN is K nearest neighbor clustering algorithm which is a collaborative filtering algorithm. KNN basically uses the database in which the data is clustered to infer knowledge. KNN uses only k as a parameter and does not make any assumptions on the underlying data and uses user similarity. [6] In this method to get the prediction of the ratings, a similarity measure such as cosine similarity is used to get similar users and then the ratings of unknown movies of a particular user are predicted using ratings given by the K nearest users.

Curse of Dimensionality: As the number of data points and number of dimensions increase, all the points start being equidistant from each other. We need to trade-off some level of accuracy for a substantial improvement in speed.

VI. Evaluation

A. Evaluation metrics

The evaluation is done based on the following parameters such as Root Mean Square Root(RMSE), Recall, Precision.

RMSE is the standard deviation of the residuals (prediction errors).

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} (Predicted_i - Actual_i)^2}{N}}$$

Fig. 2. RMSE

Recall refers to the percentage of total relevant result correctly classified by the algorithm.

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative}$$

Fig. 3. Recall

Precision is percentage of relevant results returned by the algorithm.

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive}$$

Fig. 4. Precision

Evaluation metrics for matrix factorization

```
Split: 1
RMSE: 0.8638
Precision: 0.8782240437158482
Recall: 0.2244459989576019
Split: 2
RMSE: 0.8609
Precision: 0.8681992337164764
Recall: 0.21949023038168547
Split: 3
RMSE: 0.8678
Precision: 0.8641256830601106
Recall: 0.22658506324906658
Split: 4
RMSE: 0.8760
Precision: 0.8625136612021874
Recall: 0.21586291664020027
```

Fig. 5. Evaluation metrics for Matrix Factorization

B. Results

From the table for output comparison as shown in Figure 4, we can see some differences in metric values between the two algorithms. Figures 2 and 3 show the outputs for Matrix Factorization and KNN algorithms respectively. We can infer from the table that matrix factorization gives us lower error values as compared to KNN based recommendation. Matrix factorization also gives a higher precision and recall as compared to KNN based recommendation. Hence, we can say that matrix factorization performs a better job in returning the

Evaluation metrics for KNN based

```
Split: 1
Computing the msd similarity matrix...
Done computing similarity matrix.
RMSE: 0.9423
Precision: 0.7746174863387992
Recall: 0.2592057220657712
Split: 1
Computing the msd similarity matrix...
Done computing similarity matrix.
RMSE: 0.9572
Precision: 0.7821038251366131
Recall: 0.255056290433384
Split: 1
Computing the msd similarity matrix...
Done computing similarity matrix.
RMSE: 0.9521
Precision: 0.7763387978142084
Recall: 0.26901841502822776
Split: 1
Computing the msd similarity matrix...
Done computing similarity matrix.
RMSE: 0.9513
Precision: 0.7696174863387986
Recall: 0.27133657271129136
```

Fig. 6. Evaluation metrics for KNN based

Parameters	Matrix Factorization	KNN
Precision	0.869945355191	0.866038251
RMSE	0.8640	0.8762
Recall	0.2311431390146	0.218207391

Fig. 7. Evaluation results

expected outcomes when compared to KNN. Hence, looking at the results based on evaluation we can argue that matrix factorization is better suited for recommendation applications.

VII. User Interface Design

On the home screen, the user has to enter a user id. There are 2 files present in the backend, one for matrix factorization results and the other for KNN results. The movies for the given user ids are selected from these files and presented in two column fashion. The posters of the movies are displayed. The recommendation system fetches 10 similar movies for the user using the pre-trained model. The similar movies are displayed on the output page. Figures 5, 6 and 7 are screenshots from the User Interface.

In the Fig 7, we are displaying two columns of the recommendations to show the movie recommended by matrix factorization algorithm and KNN based recommendation so that we can compare them simultaneously.

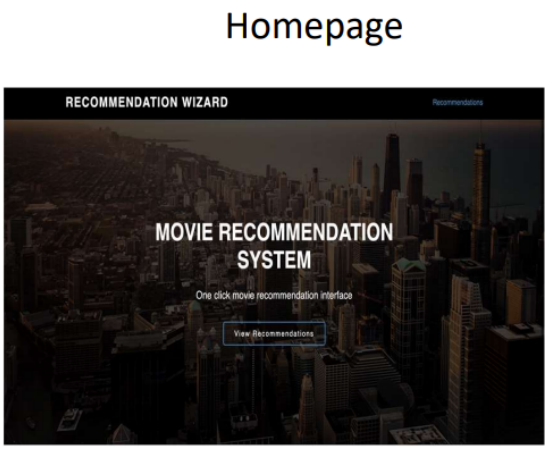

Fig. 8. Homepage

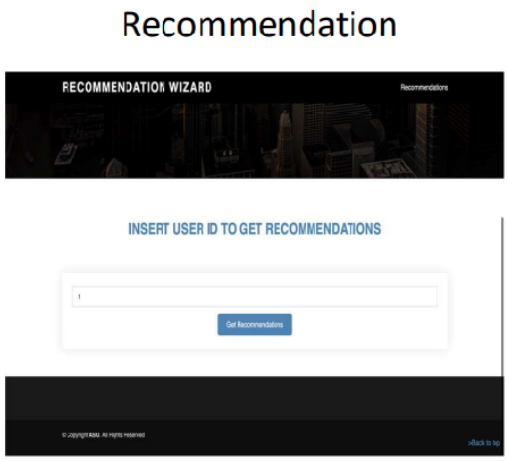

Fig. 9. Recommendation page

Fig. 10. Output page

VIII. Division of work and contributions

Swarali Chine : Worked on initial research of the given problem statement. Studied the given research papers to get a better understanding of the problem and performed literature review of collaborative filtering methods.Also worked on the literature survey on the topic and designing of the system.Analyzed various algorithms based on parameters like Root Mean Square Error, Mean Absolute Error,Root Mean Square Error. Went through various existing systems and algorithms to choose the right algorithm for the recommendation system.Helped evaluate the collaborative filtering algorithms viz Matrix Factorization and KNN and draw important insights from them.Did a thorough analysis and evaluation of the algorithms and came on the conclusion.Also, worked on proposal and final report documentation and proposal and demo presentations.

Hari Murugan Ravindran: Have worked on the initial research of the problem statement. Did literature survey on the topic we chose to analyse and evaluate all the factors which exist in the current Systems. I have worked on the preprocessing of the dataset. Focussed more on understanding the dataset, cleaning it and how to make it useful for the system. Have worked on coming up with the design of the system. Have Considered many algorithms to choose which would be suitable for the project to do a comparative study and arrived at Matrix Factorization and KNN. Helped with the Implementation and refactoring of the code and was responsible for the final test of the system. Helped in the Project Proposal, Presentation and Final Report. Finally did a thorough analysis of the comparative study done on the Recommendation system and examined whether we have arrived at the point which we initially planned to reach.

Radhika Kulkarni: As part of this team, I began working on this project by doing a literature review on Recommender systems. I researched the various applications of it in the real world and various algorithms used. I also reviewed the reference papers given to us and analysed the approaches presented in it. Furthermore, I looked at the various datasets available and explored more on MovieLens Dataset specifically. As a team, we decided to implement and present a comparison between Matrix Factorization and KNN algorithms. It was imperative of me to study these algorithms thoroughly before moving to the implementation of the project. I was more thoroughly involved in the implementation, coding and debugging of the system. For doing this, I was involved in discussions with other team members in the regular team meetings where we analysed our progress on the project and also checked our understanding. I was also involved in preparing the presentation for the demo along with other members of our group.

Pooja Bihani: I worked on the initial research of the given problem statement and performed literature survey to analyse the existing systems. I also did a research on content-based and collaborative filtering algorithms and contributed by identifying the algorithms that can have the best performance and

accuracy for our recommendation system. Explored datasets which would be suitable for our usecase. Helped with the refactoring and testing of the code. Aided in the evaluation of collaborative filtering algorithms such as Matrix Factorization and KNN, as well as the extraction of key ideas. Also, worked on project proposal and final report documentation.

Sharanya Banerjee: I contributed towards the project by designing the recommendation engine architecture and communicating the requirements to the team. I also played a role in testing the platform by simulating various scenarios and edge cases and exerting our system to them. Along with that, I prepared the proposal presentation and demo presentations and delivered both presentations on their respective days to the panel and the professor.

Arushi Gaur: Movie recommendation is a famous problem nowadays with multiple solutions available, each with advantages and disadvantages. I read about the numerous existing algorithms to understand their working and their pros and cons. This allowed me to decide which algorithm could better serve the project's aim and how to implement them according to the requirement. I read research papers about multiple algorithms of collaborative filtering and content-based algorithms. After finalizing the algorithms, I started working on implementing them algorithms and creating an application to recommend movies. The application displays movie recommendations from two algorithms simultaneously to provide a better comparison. After the final code changes, fixes and testing, I moved ahead with presentation preparation for the project.

IX. Conclusions

Recommendation system has become more and more important because of the information overload. For a content-based recommender system specifically, we attempt to find a new way to improve the accuracy of the representative of the movie. In our report we have broadly discussed the related works. We have used the collaborative filtering algorithms for our recommendation system such as Matrix factorization and KNN based recommendation. In the end of the project, we use KNN and various metrics to evaluate the improvement of the new approach.

References

[1] S. Shaikh, S. Rathi and P. Janrao, "Recommendation System in E-Commerce Websites: A Graph Based Approached," 2017 IEEE 7th International Advance Computing Conference (IACC), 2017, pp. 931-934, doi: 10.1109/IACC.2017.0189.

[2] Y. Liao, J. Lu and D. Liu, "News Recommendation Based on Collaborative Semantic Topic Models and Recommendation Adjustment," 2019 International Conference on Machine Learning and Cybernetics (ICMLC), 2019, pp. 1-6, doi: 10.1109/ICMLC48188.2019.8949259.

[3] P. R. Y. Jiang, H. Zhan and Q. Zhuang, "Application research on personalized recommendation in distance education," 2010 International Conference on Computer Application and System Modeling (ICCASM 2010), 2010, pp. V13-357-V13-360, doi: 10.1109/ICCASM.2010.5622798.

[4] W. Shalaby et al., "Help me find a job: A graph-based approach for job recommendation at scale," 2017 IEEE International Conference on Big Data (Big Data), 2017, pp. 1544-1553, doi: 10.1109/BigData.2017.8258088.

[5] J. Yu et al., "Collaborative Filtering Recommendation with Fluctuations of User' Preference," 2021 IEEE International Conference on Information Communication and Software Engineering (ICICSE), 2021, pp. 222-226, doi: 10.1109/ICICSE52190.2021.9404120.

[6] Y. Kato and K. Yamamoto, "Development of Sightseeing Spot Recommendation System Considering Users' Visit frequency," 2020 Joint 11th International Conference on Soft Computing and Intelligent Systems and 21st International Symposium on Advanced Intelligent Systems (SCIS-ISIS), 2020, pp. 1-8, doi: 10.1109/SCISISIS50064.2020.9322776.

[7] K. Shah, A. Salunke, S. Dongare and K. Antala, "Recommender systems: An overview of different approaches to recommendations," 2017 International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS), 2017, pp. 1-4, doi: 10.1109/ICIIECS.2017.8276172.