# CS 328: The Recommender System

Netram Choudary -21110138, Vinay Goud -21110125, Keerthi Ramineni -21110176

Discipline – Computer Science and Engineering

## *Abstract -*

*In today's digital landscape, the abundance of movie content available across streaming platforms presents users with a formidable challenge: sifting through vast libraries to find movies that resonate with their preferences. To address this issue, this project embarks on the development of a recommendation system, integrating machine learning techniques. By analysing user preferences and movie attributes, the recommendation system aims to enhance user satisfaction, foster engagement, and optimize the movie-watching experience. Ultimately, the project seeks to empower users by facilitating seamless access to relevant and engaging content, thereby enriching the digital entertainment landscape.*

***Keywords -***

*Movie recommendation, Content Based, Collaborative filtering, latent factor.*

***Problem Statement-***

*This project undertakes the task of creating a recommendation system designed to alleviate the challenge of navigating extensive movie libraries in digital platforms. By employing of content-based and collaborative filtering techniques, the system aims to provide users with personalized movie suggestions that align with their preferences and interests. Leveraging state-of-the-art machine learning algorithms, including neural networks and matrix factorization, the system endeavors to optimize user satisfaction and engagement. Through the analysis of user interactions and movie attributes, the recommendation system seeks to deliver tailored recommendations that enhance the movie-watching experience. Ultimately, the project aims to bridge the gap between users and relevant movie content, thereby enriching the digital entertainment landscape.*

## Data & Data Preprocessing:

**Movie Lens Dataset:**

### 1. Content-based Filtering :

**For Movie Features:**

- Extract relevant attributes for each movie from the dataset, including the year of release, average rating, and genre information.
- Create a feature vector for each movie, comprising the year of release, average rating as continuous variables, and binary indicators for each genre.
- Set the genre indicators to 1 if the movie belongs to a particular genre and 0 otherwise.

**For User Features:**

- Aggregate user-specific information from the dataset, including the user ID, rating count, and average rating given by the user.
- Calculate the average rating provided by the user for each genre, representing the user's preferences for different genres.
- Compile the user features into a feature vector, incorporating the user ID, rating count, average rating, and genre preference ratings.

User Feature Matrix :

| [user id] | [rating count] | [rating ave] | Action | Adventure | Animation | Children | Comedy | Crime | Documentary | Drama | Fantasy | Horror | Mystery | Romance | Sci-Fi | Thriller |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 2 | 22 | 4.0 | 4.0 | 4.2 | 0.0 | 0.0 | 4.0 | 4.1 | 4.0 | 4.0 | 0.0 | 3.0 | 4.0 | 0.0 | 3.9 | 3.9 |
| 7 | 74 | 3.0 | 3.0 | 3.1 | 3.2 | 3.0 | 3.1 | 3.4 | 0.0 | 3.0 | 3.2 | 2.8 | 3.1 | 2.3 | 2.6 | 3.2 |
| 10 | 86 | 3.4 | 3.7 | 3.6 | 3.9 | 3.8 | 3.3 | 3.5 | 0.0 | 3.3 | 3.8 | 0.0 | 3.0 | 3.5 | 2.4 | 3.9 |
| 15 | 70 | 3.3 | 2.9 | 3.2 | 3.2 | 3.1 | 3.0 | 3.3 | 0.0 | 3.6 | 3.5 | 3.7 | 3.9 | 3.8 | 3.4 | 3.6 |
| 17 | 21 | 4.2 | 4.4 | 4.4 | 4.2 | 3.8 | 3.8 | 4.2 | 3.5 | 4.1 | 4.6 | 0.0 | 4.0 | 4.0 | 4.2 | 4.3 |
| 18 | 229 | 3.6 | 3.5 | 3.6 | 3.8 | 3.8 | 3.4 | 3.8 | 3.5 | 3.9 | 3.7 | 2.8 | 4.0 | 3.8 | 3.6 | 3.7 |

Movie Feature Matrix:

| [movie id] | year | ave rating | Action | Adventure | Animation | Children | Comedy | Crime | Documentary | Drama | Fantasy | Horror | Mystery | Romance | Sci-Fi | Thriller |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 6874 | 2003 | 4.0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 8798 | 2004 | 3.8 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 |
| 46970 | 2006 | 3.2 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 48516 | 2006 | 4.3 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 |
| 58559 | 2008 | 4.2 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

### 2. Collaborative Filtering Preprocessing:

Movie Dataset:

| | movieId | title | genres |
|---|---|---|---|
| 0 | 1 | Toy Story (1995) | Adventure\|Animation\|Children\|Comedy\|Fantasy |
| 1 | 2 | Jumanji (1995) | Adventure\|Children\|Fantasy |
| 2 | 3 | Grumpier Old Men (1995) | Comedy\|Romance |
| 3 | 4 | Waiting to Exhale (1995) | Comedy\|Drama\|Romance |
| 4 | 5 | Father of the Bride Part II (1995) | Comedy |
| 5 | 6 | Heat (1995) | Action\|Crime\|Thriller |
| 6 | 7 | Sabrina (1995) | Comedy\|Romance |
| 7 | 8 | Tom and Huck (1995) | Adventure\|Children |

Using The given rating building the A:
Rows: UserID , Columns: MovieID, Value: Given rating by user to Movie

| movieId | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| userId | | | | | | | | | | |
| 1 | 4.0 | NaN | 4.0 | NaN | NaN | 4.0 | NaN | NaN | NaN | NaN |
| 2 | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN |
| 3 | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN |
| 4 | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN |
| 5 | 4.0 | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN |
| 6 | NaN | 4.0 | 5.0 | 3.0 | 5.0 | 4.0 | 4.0 | 3.0 | NaN | 3.0 |
| 7 | 4.5 | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN |
| 8 | NaN | 4.0 | NaN | NaN | NaN | NaN | NaN | NaN | NaN | 2.0 |
| 9 | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN |
| 10 | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN |

## Techniques :

1) **Content-Based Filtering :** We utilise neural networks for the content-based filtering technique, specifically a multi-layer perceptron architecture implemented using TensorFlow. This approach allows us to recommend items based on the features of the items(movies) and a user's past preferences.

## Neural Network Architecture:

The neural network architecture consists of two separate networks: one for modelling user preferences and another for modelling item attributes. Each network comprises multiple dense layers with rectified linear unit (ReLU) activation functions. We use 256 units in the first hidden layer and 128 units in the second. The output layer consists of the desired number of outputs, which in this case is 32.

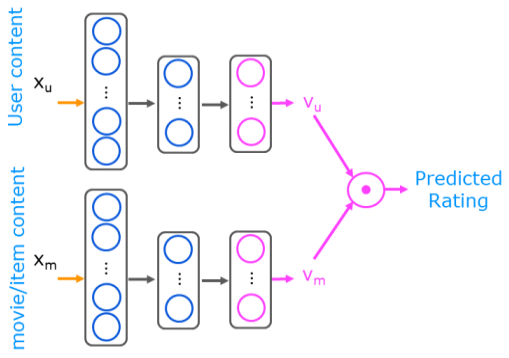


**Image:** Neural Network architecture for content-based filtering

## Model Training :

We train the model to predict movie ratings or preferences based on user input and movie attributes. The cost function used for optimization is the Mean Squared Error (MSE), which measures the average squared difference between predicted and actual ratings. We employ the Adam optimizer with a learning rate of 0.01 to minimize the cost function and update the model parameters.

## Model Architecture :

The model architecture is designed to take user and item features as inputs and output the predicted ratings or preferences. The user input and item input are passed through their respective neural networks to obtain user and item embeddings. These embeddings are then normalized using L2 normalization to ensure consistency in scale. Finally, the dot product of the user and item embeddings is computed to produce the final output, representing the predicted rating or preference for the given user-item pair.

## Model Evaluation :

After training the model, we evaluate its performance using appropriate evaluation metrics such as Mean Squared Error (MSE) or Root Mean Squared Error (RMSE) on a held-out validation dataset.

## Prediction Functions:

We provide functions for making predictions for both existing users and new users. These functions input user and item features and return the predicted ratings or preferences for the given user-item pairs.

By leveraging neural networks and content-based filtering techniques, our recommendation system aims to provide accurate and personalized recommendations to users based on their past preferences and the features of the items.

### a. Recommending top predicted movies to new user:

| y_p | movie id | rating ave | title | genres |
|---|---|---|---|---|
| 4.4 | 54001 | 3.9 | Harry Potter and the Order of the Phoenix (2007) | Adventure\|Drama\|Fantasy |
| 4.3 | 98809 | 3.8 | Hobbit: An Unexpected Journey, The (2012) | Adventure\|Fantasy |
| 4.3 | 137857 | 3.6 | The Jungle Book (2016) | Adventure\|Drama\|Fantasy |
| 4.3 | 81834 | 4 | Harry Potter and the Deathly Hallows: Part 1 (2010) | Action\|Adventure\|Fantasy |
| 4.3 | 8368 | 3.9 | Harry Potter and the Prisoner of Azkaban (2004) | Adventure\|Fantasy |
| 4.3 | 5816 | 3.6 | Harry Potter and the Chamber of Secrets (2002) | Adventure\|Fantasy |
| 4.3 | 69844 | 3.9 | Harry Potter and the Half-Blood Prince (2009) | Adventure\|Fantasy\|Mystery\|Romance |
| 4.2 | 59501 | 3.5 | Chronicles of Narnia: Prince Caspian, The (2008) | Adventure\|Children\|Fantasy |
| 4.2 | 106489 | 3.6 | Hobbit: The Desolation of Smaug, The (2013) | Adventure\|Fantasy |
| 4.2 | 41566 | 3.4 | Chronicles of Narnia: The Lion, the Witch and the Wardrobe, The (2005) | Adventure\|Children\|Fantasy |

**b. Recommending top predicted movies to existing user:**

| y_p | y | user | user genre ave | movie rating ave | movie id | title | genres |
|---|---|---|---|---|---|---|---|
| 4.5 | 5.0 | 2 | [4.0] | 4.3 | 80906 | Inside Job (2010) | Documentary |
| 4.3 | 3.5 | 2 | [4.0,4.0] | 3.9 | 99114 | Django Unchained (2012) | Action\|Drama |
| 4.2 | 4.0 | 2 | [4.0,4.1,4.0,4.0,3.9,3.9] | 4.1 | 79132 | Inception (2010) | Action\|Crime\|Drama\|Mystery\|Sci-Fi\|Thriller |
| 4.2 | 3.5 | 2 | [4.0,4.1,4.0,3.9] | 3.8 | 8798 | Collateral (2004) | Action\|Crime\|Drama\|Thriller |
| 4.2 | 4.5 | 2 | [4.0,4.0] | 4.1 | 68157 | Inglourious Basterds (2009) | Action\|Drama |
| 4.1 | 4.5 | 2 | [4.0,4.1,4.0] | 4.2 | 58559 | Dark Knight, The (2008) | Action\|Crime\|Drama |
| 4.1 | 4.5 | 2 | [4.1,4.0,3.9] | 4.0 | 80489 | Town, The (2010) | Crime\|Drama\|Thriller |
| 4.1 | 4.0 | 2 | [4.0,4.1,3.9] | 4.0 | 6874 | Kill Bill: Vol. 1 (2003) | Action\|Crime\|Thriller |
| 4.0 | 3.5 | 2 | [4.0,4.2,4.1] | 4.0 | 91529 | Dark Knight Rises, The (2012) | Action\|Adventure\|Crime |
| 4.0 | 4.0 | 2 | [4.1,4.0,3.9] | 4.3 | 48516 | Departed, The (2006) | Crime\|Drama\|Thriller |

2) **Collaborative Filtering:** Recommends items based on the preferences of similar users.

## 2.1. SVD (Singular Value Decomposition)

In our recommendation system project, we utilized the inbuilt Singular Value Decomposition (SVD) implementation provided by the NumPy library. Before applying the SVD algorithm, we encountered missing values (NaN) in our dataset, which required careful handling to ensure the accuracy and effectiveness of our model.

**Handling NaN Values**: To address missing values in our dataset, we employed strategies such as replacing NaN values with "zero" or with the mean of other user ratings. This preprocessing step was crucial for ensuring that the SVD algorithm could effectively process the data and generate meaningful recommendations.

**SVD Technique:**

After preprocessing the data, we applied the SVD algorithm provided by the NumPy library to decompose the user-item interaction matrix into three constituent matrices: U, Σ, and $V^T$. However, upon analysis of the results, we observed that the model exhibited high bias, indicating significant deviations from the original ratings in our dataset. The model is to much focusing on those values that are Nan. Nan values are in lot of numbers.

## 2.2. Matrix Factorization – Lower Rank Decomposition.

Matrix factorization is a pivotal technique in collaborative filtering that is pivotal for deriving latent representations of users and items from their interactions. In this study, we present a custom Matrix Factorization algorithm tailored for recommendation systems using TensorFlow, an open-source machine learning framework. This algorithm decomposes the user-item interaction matrix into lower-rank matrices, capturing underlying patterns in user preferences and item characteristics.

## Customized Matrix Factorization Algorithm:

We devised a novel Matrix Factorization algorithm encapsulated in a custom implementation using TensorFlow. The algorithm unfolds as follows:

**Initialization**: We initialize two matrices, W and H represent user factors and item factors, respectively, with random values.

Optimisation Objective: Our optimization objective aims to minimize the difference between the reconstructed user-item interaction matrix

WH and the observed rating matrix A. We adopt the Adam optimizer to minimise this objective efficiently.

Iterative Training: The model iteratively updates the user and item factor matrices to minimize the reconstruction error. Training continues for a predetermined number of iterations, ensuring convergence or until a convergence criterion is met.

**Function to Generate Top Predicted Movies for a User:**

We provide a utility function that facilitates the generation of top-N movie recommendations for a given user. Leveraging the learned user and item factor matrices, this function computes the predicted ratings for the user and retrieves the top-N movies with the highest predicted ratings.

Demonstrative usage of the provided functions involves generating personalized recommendations for specific users and evaluating the model's performance through RMSE calculations across all users and movies. These analyses provide insights into the model's efficacy in predicting user preferences and offer a means to assess the effectiveness of the collaborative filtering approach.

## Experimental Results:

### 1. Content Based filtering methods:

| | |
|---|---|
| Training set MSE | **0.0721** |

| | |
|---|---|
| Test Set MSE-Loss | **0.08160261809825897** |

## 2. Matrix Factorization Approach:

| | |
|---|---|
| RMSE loss | **0.57486** |

In implementing and evaluating three different recommendation techniques, our findings indicate that the content-based model outperforms the collaborative filtering models in terms of predictive accuracy and performance. Specifically, we observed a significant improvement in Mean Squared Error (MSE) loss when compared to the other models.

The superiority of the content-based approach can be attributed to its utilization of additional attributes beyond just user ratings. By incorporating features such as movie genres and other movie attributes, the content-based model is able to capture more nuanced patterns and preferences. This enriched input enables the model to generate more accurate and personalized recommendations for users.

Furthermore, the content-based model's ability to leverage diverse attributes of both users and movies contributes to the learning of more robust embeddings. These embeddings capture the underlying relationships between users, movies, and their features, resulting in more effective recommendation predictions.

## Conclusion:

Overall, our findings underscore the effectiveness of content-based recommendation techniques, particularly in scenarios where rich feature information is available. By harnessing the power of diverse attributes and genres, content-based models offer a promising approach to enhancing recommendation systems and delivering tailored content recommendations to users.

## Source code:

You can access all the code files here

## References:

Roy, D. and Dutta, M. (2022) ‘A systematic review and research perspective on Recommender Systems’, *Journal of Big Data*, 9(1). doi:10.1186/s40537-022-00592-5.