Title: Movie Recommendation Systems
Team members: Sunayana Gupta, Jayashree Natarajan, Dharmit Prajapati, Aakash Mathai, Abhinav Venepally, Vishnuraj Kunnathveetil
Method attributes and presentation: The goal of this project is to create a movie recommendation system based on the 'MovieLens' dataset. We want to combine the user's personalisation with the movie's general qualities, such as genre, popularity, and so on. We used a popularity model, a content-based model, a collaborative filtering model, models based on latent factors and an auto encoder model. We divided the dataset into test and train categories based on UserID, such that 80 percent of each user's reviews are utilized to train the model and the remaining 20 % are used to evaluate the model's accuracy. The train test split function in the scikit-learn package is used to do the aforementioned. The function's stratify property defines the characteristic that divides the dataset into testing and training.

Auto Encoders based model: A 2D matrix is generated for Auto Encoder training. The first n columns indicate to the number of movies that the user has rated. If the user hasn't given the film a rating, it will have a value of 0. The number of genres will be shown by the next n columns. Here, we're constructing a list of movies that the user is likely to view. As a result, the column for user 1 and Genre 1 reflects the number of movies that user 1 has rated 3 or higher in the Genre 1 category. This is used to assess a user's preferences for a specific genre. The user characteristics, such as gender, age, and the number of months after registration, are listed in the last few rows. We picked three layers since the number of layers (and hence neurons) rises exponentially with the number of levels. In our scenario, the results of 5 and 7 layered neural networks are just marginally better than the results of 3 layered neural networks. Below are the results we obtained by varying the number of layers:

Auto Encoder results					
Tuples for training	Hidden layers	Epochs	Training loss	Test RMSE	Test MAE
Movie Rating only	3	200	0.9158	0.952	0.913
Movie Rating only	3	1000	0.7523	1.134	1.235
Movie Rating, Genre	3	200	0.883	0.952	0.839
Movie Rating, Genre	5	200	0.887	0.952	0.843
Movie Rating, Genre, User attributes	3	200	0.862	0.952	0.39
Movie Rating, Genre, User attributes	3	1000	0.853	0.952	0.38
Movie Rating, Genre, User attributes	5	200	0.862	0.952	0.39

Content Based Recommendation: We attempted to incorporate the most important aspects of the film, such as genres, release year, summary, and tagline. The genre that a film belongs to is a good way to categorize it. This attribute is used to prepare a user vector once again. We retrieved movie summary and taglines information by integrating MovieLens data with TMDB data. After that, the TF-IDF vectorizer was used to build a TF-IDF vector for each movie. Cosine similarity was determined for each movie-movie combination once the TF-IDF vector was generated. We are trying to improve the model by using other features like genre also.

Collaborative Filtering: This technique differs from content-based methods in that the neighborhood or similarity is determined only by rating behavior. The performance of the KNN (K-nearest neighbor) algorithm for collaborative filtering was evaluated using the Surprise library prediction algorithms. We analysed the performance of KNN based algorithms for item-item and user-user similarity.

Matrix Factorization The latent factor approach, SVD, is used for forecasting the unknown ratings. The SVD algorithm uses a lower-dimensional representation of movies to map people who like similar films together. It finds hidden latent characteristics that allow movies to be mapped into the same space as the user. We have used the Surprise library to analyse the matrix factorization models.

Future planned work We plan to experiment more with the hyper-parameters of the models and the attributes used for training and get optimal recommendations. We also plan to create a hybrid model using the models which have lower RMSE to get recommendations which are the best considering the genre and user similarity with other user. We are yet to formulate the specifics of the hybrid model.