Portfolio Project

Ryan Thompson

Colorado State University - Global

CSC 510

Professor Banerjee

11 February 2025

AI Movie Recommender

The movie recommender system described in this paper is a practical example of applying artificial intelligence (AI) to personalized content delivery. The objective of this AI program is to recommend movies to users based on their ratings, using collaborative filtering techniques to suggest movies that align with the user’s preferences. By leveraging data from the MovieLens dataset, the program aims to provide relevant movie recommendations based on the similarity between user preferences and item characteristics. This essay explores the tools, methods, and approaches employed in the final version of the movie recommender system, highlighting the use of machine learning, expert system principles, and symbolic planning.

The program is built using Python, a widely used programming language for data science and AI applications. Several libraries were utilized to implement the recommender system effectively. The first library is pandas, which is essential for data manipulation, such as loading datasets, organizing data into DataFrames, and performing operations like pivoting and sorting. NumPy is used to handle numerical computations, while scikit-learn's cosine\_similarity function is used to measure the similarity between movies based on the user ratings.

The data used for training the recommendation model comes from the MovieLens 100K dataset (Harper & Konstan, 2015). This dataset consists of ratings data from 100,000 movie ratings provided by 943 users for 1682 movies, making it an ideal dataset for testing collaborative filtering algorithms. The file paths to the dataset are specified in the code and are used to read the necessary movie and ratings data. Through these tools, the program can perform calculations to determine which movies are most similar to each other based on user preferences.

In the movie recommender system, the primary search method used is cosine similarity, which computes the similarity between movies based on user ratings. Cosine similarity is a popular method in collaborative filtering as it measures the cosine of the angle between two vectors, allowing the system to determine how similar two movies are, based on the users who have rated them.

The process of generating recommendations begins with constructing a user item matrix from the MovieLens ratings data. This matrix has users as rows and movies as columns, with ratings as values. Cosine similarity is then computed on this matrix, transposed so that the similarity is measured between movies (instead of users). This results in a similarity matrix that indicates how closely related each movie is to every other movie. When a user selects a movie, the program retrieves the most similar movies based on the pre-calculated similarity scores, offering personalized suggestions for the user.

The movie recommender system presented in this code does not use deep learning models, but rather relies on traditional machine learning techniques, specifically collaborative filtering using cosine similarity. While deep learning models like neural collaborative filtering (NCF) and recurrent neural networks (RNNs) can offer more sophisticated methods for recommendation tasks (Huang et al., 2017), the current implementation focuses on a simpler and computationally more efficient approach. Deep learning models could be explored in future versions of the program for improved performance, especially in dealing with larger datasets or complex user behaviors.

Although deep learning models are not used in this program, it does contain elements of expert systems. Expert systems in AI are designed to replicate human expert decision making by applying knowledge and inference rules to solve problems (Jackson, 1999). In this case, the program utilizes the knowledge embedded in the MovieLens ratings dataset to make recommendations. The similarity matrix is a form of knowledge representation that encodes the relationships between movies, and the program uses this knowledge to provide movie recommendations.

The core of the expert system aspect is the recommendation algorithm, which works similarly to how an expert would suggest movies based on patterns they observe in the data. The program’s recommendation logic mimics an expert’s decision process by considering the ratings data and identifying movies that share similarities with those that a user has previously rated highly. In this sense, the recommender system acts as a rule based expert system, guiding the user to movies they are likely to enjoy based on historical preferences.

The knowledge in the movie recommender system is represented primarily through the user-item rating matrix and the movie similarity matrix. The user item matrix represents explicit user preferences, with each entry indicating how a particular user has rated a particular movie. This matrix is filled with zeros for unrated movies, representing unknown or uncollected user preferences. Once the movie similarity matrix is computed, it becomes the key source of knowledge that drives the recommendation process. This matrix encodes the relationships between movies, reflecting how similar movies are based on user ratings.

Furthermore, the program includes a dictionary that maps movie IDs to their corresponding titles, allowing for easy translation between numerical IDs and human readable movie names. This form of knowledge representation makes it easier for the program to interact with users, as they can select and receive recommendations in terms of actual movie titles rather than numerical IDs.

Symbolic planning refers to the use of symbolic representations to plan and make decisions based on available knowledge. In the context of this movie recommender system, symbolic planning is utilized in how the program generates and manages the sequence of movie recommendations. While the system is not concerned with robot navigation or physical space, the planning involves symbolic reasoning about the relationships between movies and users’ preferences. When a user selects a movie, the program plans the next steps by analyzing the similarity matrix, using the symbolic representation of movies and their relationships to suggest the top five most similar movies.

Moreover, the program’s interaction flow represents a type of symbolic decision making. It guides the user through a decision process selecting a movie, receiving recommendations, and possibly making further selections. The system's decision making process is symbolic in that it uses abstract representations (movie titles, ratings, and similarity scores) to generate meaningful outputs for the user.

In conclusion, the movie recommender system described in this paper effectively applies machine learning techniques, expert system principles, and symbolic planning to deliver personalized movie recommendations. By leveraging libraries like pandas, NumPy, and scikit-learn, the program computes similarity scores between movies and recommends relevant titles to users based on their previous ratings. While deep learning models were not used, the system provides a functional and efficient solution to the problem of movie recommendation. Future versions of the program could benefit from incorporating more advanced techniques, such as deep learning, to improve recommendation accuracy.

### 

### 

### 

### 

### 

References

Harper, F. M., & Konstan, J. A. (2015). The MovieLens Datasets: History and Context. *ACM Transactions on Interactive Intelligent Systems (TiiS), 5*(4), 1-19.<https://doi.org/10.1145/2827872>

Jackson, P. (1999). *Introduction to Expert Systems* (3rd ed.). Addison-Wesley.

Huang, Z., Yao, L., & Lee, D. (2017). Deep Learning for Recommender Systems. *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, 1973-1982.<https://doi.org/10.1145/3132847.3132906>