Project Report: Movie Recommendation System Task

1. The Task I Was Assigned

I was given the task of building a movie recommendation system using the MovieLens 100K dataset. The core requirements included building a system based on user similarity, generating recommendations for a given user, and evaluating its performance using precision at K. Bonus objectives included implementing item-based collaborative filtering and matrix factorization (SVD).

2. The Problem and Its Solution

The Problem: In today's digital landscape, users are often overwhelmed by choice. Recommender systems solve this problem by filtering through vast amounts of information to suggest items (in this case, movies) that are most likely to be of interest to a specific user. The challenge lies in creating a system that can accurately predict user preferences from sparse data (where most users have only rated a small fraction of the available movies) and do so efficiently. Building a robust system requires careful consideration of different methodologies and their suitability for the dataset.

The Solution: This project addresses the problem of information overload by developing and evaluating three different recommendation systems:

- **User-Based Collaborative Filtering:** This traditional approach finds users with similar tastes and recommends movies that those users liked.
- **Item-Based Collaborative Filtering:** This method finds movies that are similar to the ones a user has already rated highly and recommends them.
- **Matrix Factorization (SVD):** This advanced technique uncovers latent features that explain the ratings, providing a more robust and scalable way to make predictions.

By comparing these three methods, the project identifies the most effective approach for this dataset, providing a powerful tool for personalized movie suggestions.

3. How I Approached the Task

To complete the assignment, I followed a structured, multi-stage process, focusing on a logical progression of data science steps.

Data Acquisition, Merging, and Exploration

- **My Goal:** To load the dataset files (u.data, u.item) and merge them into a single, comprehensive DataFrame for analysis.

- **What I Did:** I loaded the ratings and movie information files, defining column headers manually. I then merged the two DataFrames on item_id to create a master DataFrame. I performed initial data exploration to check for missing values and understand the data's structure.
- **The Result:** I successfully created a master DataFrame of 100,000 ratings and confirmed that the video_release_date column was empty and could be dropped. I also noted the presence of some missing values in release_date and IMDb_URL, which were not critical for this project.
- **My Conclusion:** The data was successfully prepared and ready for matrix creation.

Creating the User-Item Matrix

- **My Goal:** To transform the long-format data into a wide-format user-item interaction matrix, the foundation for collaborative filtering.
- **What I Did:** I used the pivot_table function to create a matrix with user_id as rows and movie_title as columns, with the values being the rating.
- **The Result:** I successfully created a sparse matrix with dimensions (943 users, 1664 movies) that contained a large number of NaN values, as expected.
- **My Conclusion:** This matrix was the correct format for proceeding with collaborative filtering.

Handling Sparse Data and Similarity Calculation Prep

- **My Goal:** To prepare the user-item matrix for similarity calculations by handling sparse data and removing user rating bias.
- **What I Did:** I filled all NaN values in the matrix with 0, effectively treating unrated movies as a neutral score for similarity calculation purposes. I then normalized the data by subtracting each user's mean rating, a crucial step for ensuring that similarity is based on rating patterns rather than a user's overall rating scale.
- **The Result:** The matrix was transformed into a dense, normalized format ready for similarity calculations.
- **My Conclusion:** This preprocessing step was vital for ensuring the accuracy and effectiveness of the similarity-based models.

User-Based Collaborative Filtering

- **My Goal:** To build a user-based recommendation system that predicts ratings and generates recommendations.
- **What I Did:** I calculated the user-to-user similarity matrix using cosine_similarity on the normalized user-item matrix. I then developed a function (predict_rating) that uses this matrix to predict a rating for an unseen movie based on the ratings of all other users. Finally, I created a function (get_recommendations) to generate a list of the top recommended movies for a specific user.
- **The Result:** The system successfully generated a list of movie recommendations for a test user. The recommendations were logical, with predicted ratings on the expected 1-5 scale after fixing a sparsity issue by considering all users for predictions.
- **My Conclusion:** The user-based recommender was fully functional and ready for evaluation.

Item-Based Collaborative Filtering (Bonus)

- **My Goal:** To implement an item-based recommendation system and compare it to the user-based approach.
- **What I Did:** I implemented an item-based system by transposing the user-item matrix and calculating item-to-item similarity. I then developed a prediction function that calculates a user's likely rating for a movie based on their own ratings for similar movies.
- **The Result:** The item-based system successfully generated a list of recommendations. Initial issues with inflated predicted ratings were fixed by using a corrected normalization and prediction formula.
- **My Conclusion:** The item-based recommender was functional and provided a second model for comparison.

Matrix Factorization (SVD) (Bonus)

- **My Goal:** To implement a more advanced recommender system using Matrix Factorization and compare it to the collaborative filtering methods.
- **What I Did:** I used TruncatedSVD to decompose the filled user-item matrix into latent factors. I then reconstructed the matrix to get predicted ratings for all users and all movies. I created a function to generate recommendations based on these predicted ratings, and I clipped the ratings to the 1-5 scale to ensure interpretability.
- **The Result:** The SVD-based system successfully generated a list of recommendations with interpretable ratings on the correct 1-5 scale.

- **My Conclusion:** The SVD model provided a third, more advanced recommendation system for the final evaluation.

Final Model Evaluation and Comparison

- **My Goal:** To conduct a final, comprehensive evaluation of all three models to select the best performer.
- **What I Did:** I used the Precision at K metric (k=10) to evaluate the performance of the user-based, item-based, and SVD-based models on a random sample of 50 users from the test set. I compiled the average precision scores for all three models.
- **The Result:** The SVD-based model was selected as the best performer, with the item-based model coming in second.
- **My Conclusion:** The final evaluation showed that the SVD-based model's ability to uncover latent factors proved to be the most effective strategy for this dataset, surpassing the traditional collaborative filtering methods.

4. Final Result of My Work

After completing all assigned tasks and bonus objectives, my final conclusion is that the **SVD-Based Matrix Factorization model** is the most effective solution for building a movie recommendation system with the MovieLens 100K dataset. Its ability to discover latent features that govern user-item interactions led to the highest average Precision at K score in my final evaluation.

This project successfully demonstrated the implementation and comparison of three key recommendation methodologies. The insights gained highlight that a model's effectiveness is highly dependent on the dataset's characteristics, and for this dataset, an advanced technique like SVD proved to be a superior approach. The system is now capable of generating accurate, personalized, and relevant movie recommendations, providing a powerful solution to the problem of information overload.