Introduction

A recommender system is a type of information filtering mechanism that anticipates and proposes items or content to users by analyzing their preferences, behavior, or demographic details.
The information about the dataset used can be found at data/raw/ml-100k/README.

Data Analysis

Some conclusions about data:

1. We have 943 users, and most of them are in the age group 20-30
2. We have 1682 movies, and most of them are Drama
3. Most of the movies are released in 1993-1995
4. Most of the ratings are 4, and also there is a bias toward higher ratings
5. We have 21 occupations, and most of the users are students

Model Implementation

I've chosen the Surprise library. After experimenting with various models (collaborative filtering), the choice is the SVD++ algorithm. SVD++ (Singular Value Decomposition with Implicit Feedback) is a collaborative filtering algorithm that extends the classic SVD by incorporating implicit feedback information. It factors in both explicit user ratings and implicit feedback, making it a powerful choice for recommendation systems.

Model Advantages and Disadvantages

Advantages:

1. Matrix Factorization. It enables the model to represent user-item matrix in a lower dimensional space.
2. Incorporation of Implicit Feedback. SVD++ extends traditional collaborative filtering by incorporating implicit feedback, enhancing the model's ability to handle scenarios where explicit ratings may be sparse.

Disadvantages:

1. Memory intensive
2. Scalability issues
3. Dependency on quality of input data

Training Process

1. Data Preprocessing
   a. Loading Data: Data is loaded using Reader and Dataset classes from the surprise library
   b. Splitting Data: Data is splitted into training and testing sets
2. Model Training
   a. Each model from surprise library was tested using cross_validate function provided in the surprise library
   b. The results of tests showed that the best performing model is SVD++
   c. Then, hyperparameter tuning was done using GridSearchCV.

Evaluation

Initially, the task was treated as a regression problem. The model predicts the rating, so the quality of the model is measured by RMSE.
How to use evaluate.py:

```
py evaluate.py --user_id USER_ID --top_k TOP_K
```

If not stated, user_id is random and top_k is 5. Also, the script will use the whole dataset to test the model on it and will print the RMSE metric.

Results

Root Mean Squared Error (RMSE) : 0.7496

The achieved RMSE indicates quite satisfactory performance for this simple model.

What can be improved: make a more complex model that also takes in consideration another data than user-item matrix, for example, it is possible to find similar users by their personal data(age, gender, location), and then recommend movies to them.