Module 17: Week 19: Required Assignment – AI Driven features

### **1. Collaborative Filtering: User-Based vs. Item-Based (4 Marks)**

a) **Compare and contrast** user-based collaborative filtering with item-based collaborative filtering.

b) Discuss which approaches are **more efficient** in a large-scale system like Netflix.

## 1. Collaborative Filtering: User-Based vs. Item-Based (4 Marks)

### a) Comparison and Contrast

* **User-Based Collaborative Filtering**
  + Finds **similar users** based on their rating patterns.
  + Recommends items that similar users liked but the active user hasn’t seen.
  + Example: *If User A and User B rate movies similarly, then User A may like other movies User B has rated highly.*
  + → similarity between users.
  + needs to recompute similarities when new users join; IBCF reuses item-item similarities (more stable).
  + becomes slow with millions of users
* **Item-Based Collaborative Filtering** 
  + Finds **similar items** based on users’ ratings.
  + Recommends items similar to those the user already liked.
  + Example: *If a user liked Movie X, and Movie Y was rated highly by users who also liked X, recommend Y.*
  + → similarity between items.
  + reuses item-item similarities (more stable).
  + more scalable because the number of items is usually smaller and more static than the number of users.

### b) Efficiency in Large-Scale Systems (e.g., Netflix)

* **Item-Based Collaborative Filtering is generally more efficient** for large-scale systems:
  + The **user base grows continuously** (millions of new users), making UBCF computationally expensive.
  + The **item catalog (movies/TV shows) is smaller and changes less frequently**.
  + Precomputing item-item similarities is feasible and reduces online computation.
* Hence, Netflix and similar platforms rely more on **item-based CF, matrix factorisation, and hybrid models**.

### **2. Sparsity and Matrix Factorisation (2 Marks)**

a) **Discuss** matrix sparsity in collaborative filtering.

b) **Explain** how matrix factorisation helps address the sparsity issue.

## 2a. Sparsity and Matrix Factorisation

### Matrix Sparsity in Collaborative Filtering

* The **user–item matrix** is extremely **sparse**:
  + Users typically rate/watch only a small fraction of all available items.
  + Example: On Netflix, a user may rate 50 movies out of thousands.
* This leads to difficulties:
  + Hard to compute similarities reliably.
  + Many missing entries (unknown ratings) → makes predictions challenging.

### b) How Matrix Factorisation Helps

* **Matrix factorisation (e.g., SVD, ALS)** decomposes the sparse user–item matrix into two smaller dense matrices:
  + **User latent factors** (captures preferences, e.g., love for sci-fi or romance).
  + **Item latent factors** (captures characteristics, e.g., movie belongs to sci-fi genre).
* By multiplying these factors, we can **estimate the missing ratings** in the sparse matrix.
* This addresses sparsity by uncovering **hidden patterns** and enabling accurate predictions even when direct data is missing.

**Summary**

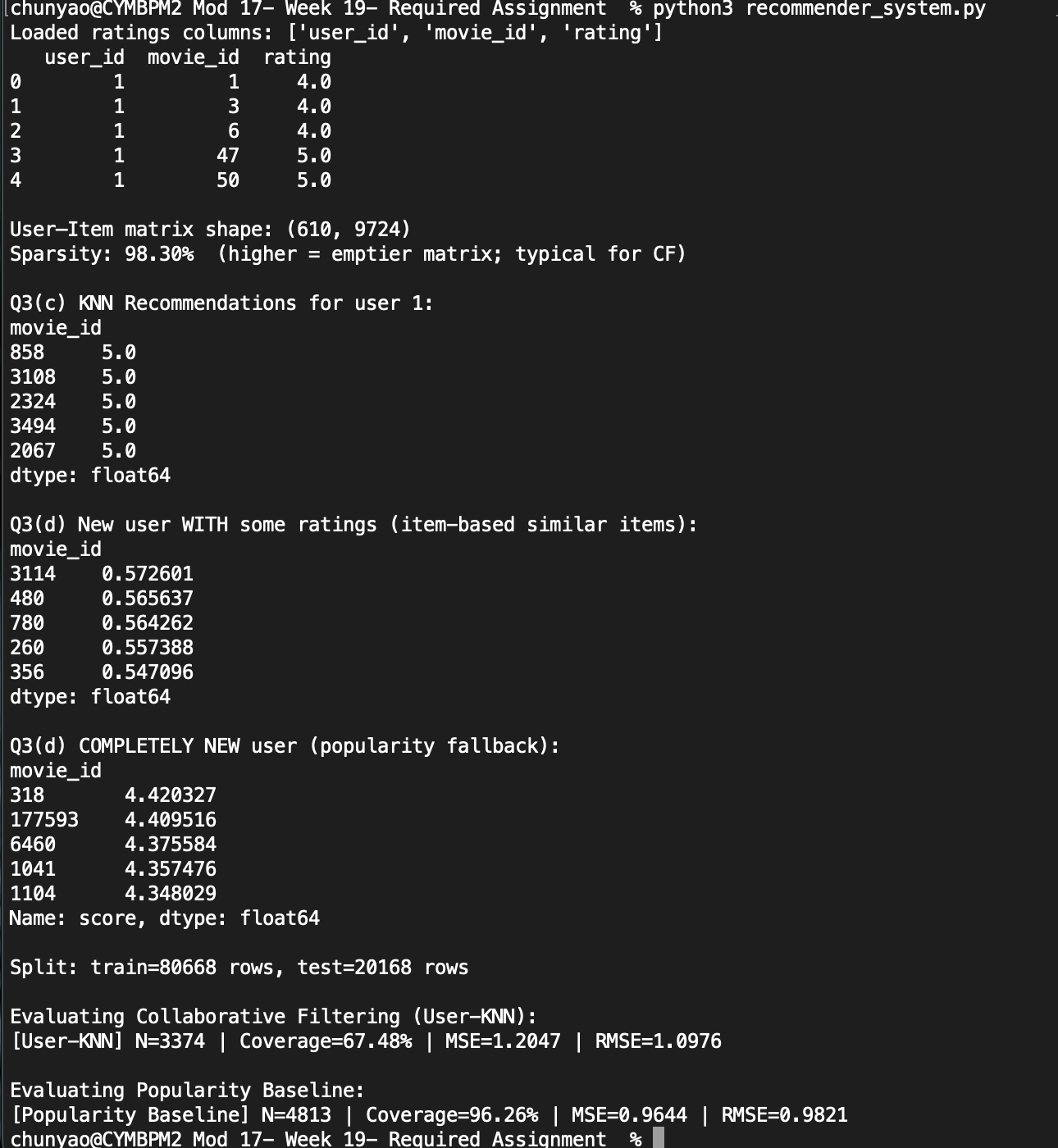
1a) UBCF vs IBCF → difference in similarity focus; UBCF less scalable.

1b) Netflix favors IBCF (and advanced models) for efficiency.

2a) Sparsity = too many missing ratings.

2b) Matrix factorisation fills gaps by learning latent factors.

Q3 and Q4 Screen shot:



Q4 - The dataset provided does not include item metadata (like genres), so a true content-based system could not be evaluated. Instead, we compared collaborative filtering against a popularity baseline.”