**1. Câu hỏi về thuật toán**

* **Cosine similarity / Pearson correlation**: cách tính độ tương đồng giữa người dùng hoặc sản phẩm.
* **Item-based vs User-based CF**: ưu nhược điểm, khi nào dùng loại nào.
* **Cold-start problem**: làm sao gợi ý khi user mới hoặc item mới.
* **Overfitting / underfitting trong latent factors**: chọn k thế nào hợp lý.

**2. Câu hỏi về đánh giá**

* **RMSE / MAE**: cách tính độ chính xác dự đoán.
* **Precision, Recall, F1**: khi gợi ý top-N sản phẩm.

**3. Câu hỏi nhanh lý thuyết**

* Nếu tăng/giảm số latent features k thì dự đoán thay đổi thế nào.
* Nếu ma trận quá thưa, SVD/NMF có vấn đề gì.
* Tại sao chuẩn hóa user mean giúp cải thiện dự đoán.

**4. Câu hỏi nâng cao / tình huống**

* “User mới chỉ đánh giá 1 item, làm sao gợi ý được?” → cold-start, dùng content-based hoặc hybrid.
* “Có nên dùng tất cả dữ liệu để training không?” → nên tách train/test, tránh overfitting.

💡 **Mẹo**: Nếu nắm chắc **code SVD/NMF**, **cách dự đoán giá trị thiếu**, **top-N recommendation**, **cosine similarity**, **cold-start** và **effect của k**, thì 80–90% các câu hỏi sẽ giải quyết được.

Nếu muốn, mình có thể làm **1 bảng “câu hỏi dự đoán – câu trả lời siêu ngắn gọn”**, chỉ 1 trang thôi, bạn cầm vô kiểm tra đọc là xong.  
Bạn có muốn mình làm luôn không?

Ok, kiểm tra về **Phân tích ma trận (Matrix Factorization - MF)** mà cho code sẵn rồi hỏi "what if..."1. Dựa trên nội dung ôn tập và các slide, thầy có thể sẽ tập trung vào các tham số cốt lõi của MF, cách nó hoạt động, và lý do tại sao lại chọn các kỹ thuật đó.

Dưới đây là 10 câu hỏi lý thuyết mà thầy có thể hỏi, xoay quanh một đoạn code về MF:

**🎯 Câu hỏi về Tham số mô hình (K và $\lambda$)**

1. **Tham số K (số lượng latent factors) trong code có ý nghĩa gì?** 222
   * **Gợi ý trả lời:** K là số chiều của không gian ẩn (latent space)3. Nó đại diện cho số lượng "đặc trưng ẩn" 4 mà mô hình tự học để mô tả mỗi user và mỗi item. Ví dụ, K có thể học ra các chiều như "phim nghiêm túc vs. phim hài", "dành cho trẻ em vs. người lớn", v.v., nhưng bản thân các chiều này không được gán nhãn rõ ràng5555.
2. **Điều gì sẽ xảy ra nếu bạn thay đổi giá trị K (ví dụ: tăng lên rất lớn hoặc giảm xuống rất nhỏ)?**
   * **Gợi ý trả lời:**
     + **K quá nhỏ:** Mô hình có thể bị "underfitting", không nắm bắt đủ các đặc điểm phức tạp của sở thích user, dẫn đến dự đoán kém chính xác.
     + **K quá lớn:** Mô hình có thể bị "overfitting"6. Nó học thuộc lòng dữ liệu training nhưng dự đoán kém trên dữ liệu test. Ngoài ra, việc training sẽ chậm hơn và tốn nhiều bộ nhớ hơn777.
3. **Mục đích của tham số regularization ($\lambda$, lambda) trong hàm mất mát là gì?** 88
   * **Gợi ý trả lời:** Mục đích chính là để **chống overfitting** (khớp quá mức)9. Nó thêm một "phần phạt" vào hàm loss dựa trên độ lớn của các vector đặc trưng (user-factors $X$ và item-factors $W$)10. Điều này ngăn cản các trọng số trở nên quá lớn, giúp mô hình tổng quát hóa tốt hơn.
4. **Nếu ta cài đặt $\lambda = 0$ (loại bỏ regularization) thì mô hình sẽ bị ảnh hưởng thế nào?**
   * **Gợi ý trả lời:** Nếu $\lambda = 0$, mô hình sẽ không có cơ chế chống overfitting11. Nó sẽ cố gắng giảm sai số (RMSE) trên tập train xuống mức thấp nhất có thể, dẫn đến việc các giá trị trong vector $X$ và $W$ trở nên rất lớn và mô hình bị overfitting nặng, hoạt động kém hiệu quả trên dữ liệu mới.

**🔧 Câu hỏi về Kỹ thuật và Đánh giá**

1. **Tại sao code lại dùng RMSE (Root Mean Squared Error) làm độ đo lỗi thay vì MAE (Mean Absolute Error)?** 12
   * **Gợi ý trả lời:** RMSE nhạy cảm hơn với các lỗi lớn (dự đoán sai nhiều)13. Vì RMSE bình phương sai số, một dự đoán sai 4 điểm (ví dụ: dự đoán 1 sao trong khi thực tế là 5 sao) sẽ bị phạt nặng hơn nhiều so với MAE. Việc này hữu ích nếu hệ thống muốn tránh các đề xuất *rất tồi tệ*.
2. **Phương pháp Matrix Factorization (model-based) này khác gì so với phương pháp lân cận (k-NN, memory-based) đã học?** 141414
   * **Gợi ý trả lời:**
     + **k-NN (Memory-based):** Không thực sự "học". Nó lưu trữ toàn bộ dữ liệu 15và tính toán độ tương đồng (ví dụ: Pearson) giữa các user (user-based) 16161616hoặc item (item-based) 17 khi cần dự đoán. Tốn bộ nhớ 18, dự đoán chậm19.
     + **MF (Model-based):** "Học" một mô hình nhỏ gọn (các ma trận $X$ và $W$) từ dữ liệu20. Quá trình học (training) chậm 21, nhưng khi dự đoán thì rất nhanh (chỉ là phép nhân vector)22. Thường xử lý ma trận thưa tốt hơn k-NN23.
3. Nếu code dùng thư viện TruncatedSVD24, nó khác gì so với việc tối ưu hàm loss bằng Gradient Descent (hoặc ALS)?
   * **Gợi ý trả lời:** TruncatedSVD là một kỹ thuật phân rã ma trận. Để dùng nó, code thường phải điền các giá trị bị thiếu (NaN) bằng một số nào đó (ví dụ: 0 hoặc trung bình)25. Ngược lại, các phương pháp tối ưu hàm loss (như trong slide 4) 26được thiết kế để **chỉ hoạt động trên các rating đã biết** ($r\_{mn}=1$) 27 và bỏ qua các ô trống, cách này thường chính xác hơn cho bài toán RS.

**💡 Câu hỏi về Hạn chế và Giải thích**

1. **Đoạn code này sẽ xử lý vấn đề "cold start" (user mới tinh, chưa rate item nào) như thế nào?** 28
   * **Gợi ý trả lời:** Code MF này sẽ **không xử lý được**. Mô hình MF cần dữ liệu rating trong quá khứ của một user để "học" ra vector đặc trưng ($w\_n$) cho user đó29292929. Nếu không có rating, không thể dự đoán. Hệ thống sẽ phải dùng một giải pháp dự phòng, ví dụ như đề xuất các item phổ biến nhất (non-personalized)30303030.
2. **Làm thế nào bạn có thể giải thích một đề xuất (recommendation) từ mô hình MF này cho người dùng?**
   * **Gợi ý trả lời:** Đây là điểm yếu lớn của MF. Rất khó giải thích vì mô hình là một dạng "black box"31313131. Ta không thể nói "Bạn được đề xuất phim này vì nó giống phim X" (đó là item-based CF 32) hay "vì nó cùng thể loại" (đó là content-based 33). Lời giải thích duy nhất là "vì các đặc trưng toán học ẩn của phim này khớp với sở thích ẩn của bạn" 34, nhưng điều này không thỏa mãn người dùng35.
3. **(Câu hỏi tổng hợp) Tại sao phương pháp này lại được xếp vào Lọc cộng tác (Collaborative Filtering)?** 36
   * **Gợi ý trả lời:** Mặc dù nó học "model", nó vẫn là CF vì các đặc trưng ẩn ($X$ và $W$) được học **đồng thời** từ rating của **tất cả** user373737. Vector của user A ảnh hưởng đến việc học vector của item B, và vector của item B lại ảnh hưởng đến việc học vector của user C38. Tức là, mô hình học sở thích dựa trên sự "cộng tác" của toàn bộ cộng đồng user, chứ không chỉ dựa trên nội dung của item (content-based)39.

Ok, mình tổng hợp thêm **10 câu hỏi nữa** mà thầy rất có khả năng hỏi, kèm trả lời cực ngắn gọn, tiện ôn chiều nay:

**10 câu hỏi bổ sung – Hệ Khuyến Nghị**

1. **User-based CF vs Item-based CF khác nhau thế nào?**

User-based: dự đoán dựa trên user giống nhau; Item-based: dựa trên item giống nhau.

1. **Cosine similarity tính ra sao?**

Cos = (A·B) / (||A|| \* ||B||), đo độ tương đồng hướng vector.

1. **Pearson correlation dùng để làm gì?**

Đo độ tương quan tuyến tính giữa user hoặc item, loại bỏ bias.

1. **Cold-start problem là gì?**

Khi user mới hoặc item mới, chưa có dữ liệu → khó dự đoán.

1. **Làm sao giải quyết cold-start?**

Dùng content-based, hybrid, hoặc yêu cầu rating ban đầu.

1. **Matrix factorization khác memory-based CF thế nào?**

Memory-based: dựa trên similarity trực tiếp;  
MF: học latent features ẩn, dự đoán giá trị chưa biết.

1. **Rank-k approximation là gì?**

Chỉ giữ k latent features lớn nhất trong SVD → giảm chiều, gần đúng R.

1. **RMSE dùng để làm gì?**

Đo sai số trung bình dự đoán với đánh giá thực tế.

1. **Precision@N và Recall@N khác gì?**

Precision@N: % gợi ý đúng trên tổng gợi ý N;  
Recall@N: % gợi ý đúng trên tổng item thực sự thích.

1. **Nếu ma trận thưa quá, dự đoán có vấn đề gì?**

Dễ overfitting, dự đoán kém chính xác → cần regularization hoặc imputation.