import pandas as pd

from flask import Flask, request, jsonify

from surprise import Dataset, Reader, SVD

from surprise.model\_selection import train\_test\_split

app = Flask(\_\_name\_\_)

# Sample user data

users\_data = {

"user\_id": [1, 2, 3],

"age": [23, 30, 26],

"gender": ["M", "F", "F"],

"location": ["New York", "London", "Tokyo"]

}

users\_df = pd.DataFrame(users\_data)

# Sample interaction data

interactions\_data = {

"user\_id": [1, 1, 2, 2, 3],

"item\_id": [101, 102, 103, 104, 105],

"interaction\_score": [5, 3, 4, 2, 5]

}

interaction\_df = pd.DataFrame(interactions\_data)

# Load data into surprise

reader = Reader(rating\_scale=(1, 5))

data = Dataset.load\_from\_df(interaction\_df[['user\_id', 'item\_id', 'interaction\_score']], reader)

trainset, testset = train\_test\_split(data, test\_size=0.2)

# Train recommendation model

model = SVD()

model.fit(trainset)

# Recommendation function

def get\_recommendations(user\_id, all\_items, model):

item\_ids = list(set(interaction\_df['item\_id'].unique()).union(set(range(101, 106))))

predictions = [model.predict(user\_id, iid) for iid in item\_ids if iid not in interaction\_df[interaction\_df['user\_id'] == user\_id]['item\_id'].values]

predictions.sort(key=lambda x: x.est, reverse=True)

top\_5 = [{"item\_id": int(pred.iid), "predicted\_score": round(pred.est, 2)} for pred in predictions[:5]]

return top\_5

# API Endpoint

@app.route("/recommend", methods=["GET"])

def recommend():

try:

user\_id = int(request.args.get("user\_id"))

if user\_id not in users\_df["user\_id"].values:

return jsonify({"error": "User not found"}), 404

all\_items = interaction\_df['item\_id'].unique()

recommendations = get\_recommendations(user\_id, all\_items, model)

return jsonify({

"user\_id": user\_id,

"recommendations": recommendations

})

except Exception as e:

return jsonify({"error": str(e)}), 500

# Run app

if \_\_name\_\_ == "\_\_main\_\_":

app.run(debug=True)