from surprise import SVD, Dataset, Reader

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

from surprise import accuracy

# Load MovieLens dataset (small version, u.data is typically used)

data\_path = 'ml-100k/u.data'  # Adjust the path if necessary

reader = Reader(line\_format='user item rating timestamp', sep='\t', rating\_scale=(1, 5))

# Load data into Surprise format

data = Dataset.load\_from\_file(data\_path, reader)

# Split data into training and testing sets

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

# Use SVD (Singular Value Decomposition) for collaborative filtering

model = SVD()

# Train the model

model.fit(trainset)

# Predict ratings for the test set

predictions = model.test(testset)

# Calculate RMSE (Root Mean Square Error) to evaluate the model

rmse = accuracy.rmse(predictions)

print(f"RMSE: {rmse}")

# Function to get top N movie recommendations for a user

def get\_top\_n(predictions, n=5):

    top\_n = {}

    # Organize predictions by user

    for uid, iid, true\_r, est, \_ in predictions:

        if uid not in top\_n:

            top\_n[uid] = []

        top\_n[uid].append((iid, est))

    # Sort the predictions for each user and get the top N

    for uid, user\_ratings in top\_n.items():

        user\_ratings.sort(key=lambda x: x[1], reverse=True)

        top\_n[uid] = user\_ratings[:n]

    return top\_n

# Get the top 10 movie recommendations for each user

top\_n = get\_top\_n(predictions)

# Print top 5 movie recommendations

user\_id = '2'

print(f"Top 5 recommendations for User {user\_id}:")

for movie\_id, rating in top\_n[user\_id]:

    print(f"Movie ID: {movie\_id}, Predicted Rating: {rating:.2f}")