a) My First algorithm approach (calculate_matches)

1) First I defined a function calculate_matches which accepts two parameters user of type Dictionary and all users of type List(Dictionary)

2) In this function user's comparison is done with all others users with the help of for loop on the basis of compatibility score arranged in descending order of score.

3) In this function compatibility score is calculated on the basis of common interests and common hobbies, so in common_hobbies attribute I stored the list of set of hobbies which are common in both users and same I did with the common_interests attribute.

4) Then I appended the empty list match_results with List of MatchResult objects which has (user_id, name, compatibilty_score,common_interests,common_hobbies)

5) I called the function get_compatibility_score inside this function which is returning the score.

---

b) My second algorithm approach (get_compatibility_score)

1) In this algorithm i defined a function named get_compatibility_score which accepts two parameters user1 of type Dictionary and user2 of also type Dictionary, and this

Function is returning compatibility score between 0 and 1 in float data type.

2) First score variable is assigned 0.0 value then its value is updated according to compatibility score of every attribute.

Scoring Methodology used is

Like in age I used the basic concept the smaller the age difference, higher the compatibility score.

I normalizes the score, assuming a maximum difference of 20 in age.

So age_score=max (0, 1-age_diff/20), which means age_score will store the maximum of between 0 and the (1-age_diff/20).

I assigned an equal weightage of 0.2 to every attribute, so five attributes sum up to 1.

Now score will be updated to (score+0.2*age_score.)

For gender_score and location_score value will be either 0 or 1 no between value is possible because location will be matched or not and same with gender interested or not.

Score again updated to (score+0.2*gender_score)

Score again updated to (score+0.2*location_score)

For common hobbies and interests

hobbies_score=Len (common_hobbies) which is the length of Set/max (Len (user1["hobbies"]),Len(user2["hobbies"]) will be calculated if user1["hobbies"] and user2["hobbies"] both the condition is true that means they both have any 1 hobby at least otherwise return 0.0

Score value will again be updated to (score+0.2*hobbies_score)

Same with the interests_score calculation

Score value be again updated to (score+0.2*interests_score)

Finally the function is returning score value.

Then FastAPI was imported and also the json

users.json file was loaded in USER_DATA

Then get("/") request and function read_root() which is printing message “Welcome to the Dating App Matchmaking API".

post("/api/v1/match/{user_id}") request and function generate_matches which accepts one parameter as user_id and returns List of MatchResult.

get("/api/v1/compatibility/{user_id1}/{user_id2}") request and function get_compatibility(user_id1,user_id2) which accepts two parameters and returns a Dictionary.

Trade-off and decision made

1) I made few decisions like in normalizing age I assumed that maximum age difference of 20 is there, that I thought is idle.

And assigned equal weightage to all the five attributes on the basis of which compatibility score is calculated.

2) Important decision made was when I was importing user and MatchResult from .models and calculate_matches and get_compatibility_score from .matching. Algorithm

When I was starting the server using uvicorn main:app --reload where my main.py file is saved then server was not starting due to this error

"Attempted relative import with no known parent package"

So all the four classes I copied in the main.py itself and then I was able to get the proper results.

Setup and running instruction

A) I saved the main.py in my C driver in a folder from there through command prompt I started the server using the command uvicorn main:app --reload

b) Swagger API UI was opened on the port http://127.0.0.1:8000/docs and then first GET request using end point

http://127.0.0.1:8000/

I got the response 200 ok with message

{

"Message": "Welcome to the Dating App Matchmaking API"

}

POST request with endpoint

http://127.0.0.1:8000/api/v1/match/user3

I got the response 200 ok

Then finally GET request with endpoint

http://127.0.0.1:8000/api/v1/compatibility/user1/user2

I got the response 200 OK.
