SPAD: Subprofile Aware Diversification

- Goal: Achieve recommendation diversity while considering user interest

Approach:

- Identify subprofile within a user's profile (liked items only)
  : Subprofile represent user's distinct interests.
- use subprofiles instead of predefined aspects for diversification.

- process:

1. Subprofile Extraction:
   - Recommend top-n items (items similar to liked items) using IBR Recommender: with there explanation (score)
   - Explanation becomes candidate subprofiles
   - Refine - Subprofile: Remove subprofiles completely contained within others (to avoid redundancy).

2. Recommendation Re-Ranking
   - using MF-ALS Recommender (Alternation least Squares Method) to generate initial recommendation.
   - Re-Rank Recommendation using objective function that Consider.
     - Relevance of items to the user (S(u,i))
     - Diversity based on Subprofile coverage

Key difference between Quad & Spad.

- Existing methods use a global set of aspects for all users.
- SPAD uses user specific subprofile as aspects, leading to personalized diversification.

$$SPAD = \sum_{S \in \mathcal{S}} \left[P(S|u) \cdot P(i|u,S) \prod_{j \in R} (1 - P(j|u,S)) \right]$$

$P(S|u)$: Probability of Subprofile S given user u.
This represents how likely Subprofile S is for user u.
It is calculated based on the Subprofile Size relative to all Subprofiles of user u.

ex: $P(S|u) = |S| / \sum |S'|$ (for all S' in S_u)

$P(i|u,S)$: probability of choosing item i given user u and subprofile S.

Equation: $$P(i|u,S) = \frac{ind(i,S) \times S(u,i)}{\sum (ind(j,S) \times S(u,j))} \quad (\text{for all } j \text{ in } R_S)$$

$ind(i,S)$: It's an indicator function (1 if item is "related" to subprofiles, 0 otherwise).

$S(u,i)$: Relevance score of item i for user u (obtained from the chosen recommendation System)

here,
- Indicator function ensures the probability is only non-zero for items relevant to the subprofile.
- The relevance score ($S(u,i)$) indicates how much the user might like the item.
- denominator normalizes the probability by considering all candidate items in recommendation set. (relative to subprofile S).

How The Sub profiles are generated ?

Step 1 generate Candidate Subprofile :

- The System uses an item - based nearest neighbour Recommender (IBT) designed for implict feedback (positive - only ratings).
- for each item (i) already liked by the user ($i \in I_u$), IBT identifies its k-nearest neighbours within user's profile ($S^*i = \{j \in I_u \mid i \in k\text{NN}\}$)
- These k - nearest neighbours essentially represent items similar to the liked item (i)

Step 2. Refining Candidate Subprofile :

- It first Sort Candidate Subprofile (S^*i) in descending order of Size (number of items)
- It iterates through the sorted list, removing any subprofile entirely contained within another Subprofile (already chosen).
  This avoids redundancy and ensures each Subprofile capture distinct user interests.

Subprofile Extraction

Let I be the set of all items.

Subprofile detection works on positively-rated items in the user's profile.
In the case of positive-only feedback, user u's profile, $Iu \subseteq I$,
is the set of items she has interacted with (liked, clicked on, purchased, etc.)

A user's subprofiles are subsets of Iu.

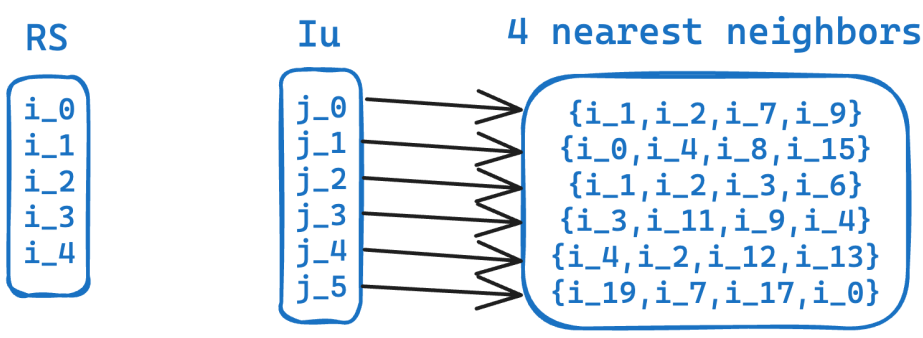
Firstly, we produce a set of recommendations RS using some recommender.

To help build subprofiles, we use an item-based nearest-neighbours
recommender. We will call it IB+

Now, For each candidate item $i \in RS$ and $i \notin Iu$,
IB+ finds items in the user's profile that have the candidate as one
of their k-nearest-neighbours:

$Si* = \{j \in Iu | i \in KNN(j)\}$

Let us take an example to understand this better.



So, S0* = set of items in Iu that have item i_0 as one of their 4 nearest neighbors.
∴ S0* = {j_1,j_5}
Similarly,
S1* = {j_0,j_2}
S2* = {j_0,j_2,j_4}
S3* = {j_2,j_3}
S4* = {j_1,j_3,j_4}

The set Si* is the explanation for why i should be recommended.

Now, IB+ scores each candidate by taking the sum of the similarities(cosine similarity)
of the candidate to the items in Si*:

$$s(u,i) = \sum_{j \in Si*} sim(i,j)$$

Let us assume that the scores for each candidate in RS are:

s(u,0) = 1.5
s(u,1) = 1.8
s(u,2) = 2.4
s(u,3) = 1.6
s(u,4) = 2.3

Now, Let Eu be the explanations for the n candidates whose scores, s(u,i), are highest.

Here, for the sake of example, let us take n = 3

∴ Eu = {S2*, S4*, S1*} (since they have the top-3 scores)
∴ Eu = {{j_0,j_2,j_4}, {j_1,j_3,j_4}, {j_0,j_2}}

We define the set of subprofiles for user u, Su ,
to be those members of Eu that do not contain any other members of Eu :

$$Su = \{Si_* \in Eu | \neg \exists S \in Eu, Si_* \subset S\}$$

what that means is, we only include those member sets in Eu,
that doesn't completely overlap with any other member in Eu.

i.e. it should not be a subset of any other member.

So, in our example :
Eu = {{j_0,j_2,j_4}, {j_1,j_3,j_4}, {j_0,j_2}}

we could observe that third member set is a subset of first member set.
Hence, we would remove third set from Eu to get Su.

∴ Su = {{j_0,j_2,j_4}, {j_1,j_3,j_4}}

Finally, we get Su by sorting the explanations in descending order of size.

In our example, Su is already sorted.

∴ Su = {{j_0,j_2,j_4}, {j_1,j_3,j_4}}

Here, each member of Su is a subprofile.