Satisfaction-Weighted SDAA

Concept

- Build on the SDAA principle (dynamic α reflecting disagreement)
- α controls the balance between choosing items that are good on average and items that nobody dislikes.
- Add user satisfaction weights that evolve across rounds
- Incorporate a disagreement threshold τ for stopping

What's new?

- Not just averaging or least-misery
- Users influence the group differently over time
- Under-served users gradually gain more weight
- Method stops once the overall satisfaction of the group is high enough (α < τ) or all iterations have been used.

Satisfaction Weights

Why weights?

- In repeated decisions, some users may consistently “lose”
- Satisfaction weights ensure:
  - Under-satisfied users gradually gain influence
  - Over-satisfied users do not dominate every round
  - Group fairness improves over time

Weight update rule

$$w_u^{(j+1)} = w_u^{(j)} \left(1 + \gamma \left(\bar{s}^{(j)} - s_u^{(j)}\right)\right)$$

Where:

- s(u) = user u’s satisfaction this round
- $\bar{s}$ = mean group satisfaction
- γ = strength of compensation

Interpretation

- If user u is below average → weight goes up
- If user u is above average → weight goes down
- Next rounds shift toward previously ignored taste

Implementation

For each iteration j:

1) **Individual top-k lists:** Build A(u, j) for each group member
2) **Candidate pool:** $G_j = \bigcup_{u \in G} A(u, j)$
3) **Prediction matrix M:** Predict scores for all users × items in Gj
4) **Satisfaction-Weighted SDAA scoring:** $\text{score}_j(i) = (1 - \alpha_j)\,\text{WeightedAvg}_j(i) + \alpha_j\,\text{Least}_j(i)$
5) **Group selection:** Choose top-k items for iteration j
6) **User satisfaction update:** Compute sat(u) relative to chosen items
7) **Update α and user weights:** Under-satisfied users get weight boosts $\alpha_{j+1} = \max_{u \in G} \text{sat}_j(u) - \min_{u \in G} \text{sat}_j(u)$
   - High α → group is divided → choose safer items (least misery).
   - Low α → group is aligned → choose higher-utility items (average).
8) **Stopping rule:** Stop if $\alpha_j < \tau$