Optimized Scheme Recommendation System

# 1. Methodology

🔹 Step 1: Product Recommendation (Collaborative Filtering)

- Approach: Partner-product binary matrix created from historical purchase data.  
- Technique: Cosine similarity used to find similar partners.  
- Top 3 products recommended for each partner based on aggregation of similar partner choices.

🔹 Step 2: Scheme Optimization (Custom Linear Programming Approach)

- Goal: Optimize the allocation of up to 3 schemes per partner-product pair to drive engagement.  
- Technique:  
 - Linear optimization model (similar to LP) used to maximize scheme effectiveness.  
 - Considered partner profile (e.g., retailer/distributor), region, growth potential, and compatibility between schemes.

🔹 Step 3: Mapping (Product × Partner → Scheme Set)

- Final mapping performed to assign Scheme\_1, Scheme\_2, and Scheme\_3 to each (Product, Partner) pair.  
- Conflict checks and redundancy filtering were applied (e.g., no duplicate schemes across the same product).

# 2. Key Findings and Business Insights

- Frequent Schemes: Scheme1 and Scheme4 are the most commonly optimized schemes across products.  
- High Coverage: All 15 unique products received optimized scheme combinations.  
- Top Partners (e.g., 001\_3) received diversified scheme portfolios covering growth, loyalty, and volume-based incentives.  
- Business Value: This approach allows dynamic bundling of products with schemes tailored to user profiles and purchase behavior.

# 3. Assumptions

- Binary purchase representation is sufficient for recommendation modeling.  
- Partners are influenced similarly by schemes irrespective of exact product pricing.  
- All schemes have static benefits (e.g., no seasonality or expiry modeling).  
- Constraints and objectives used in LP mimic real-world constraints closely.

# 4. Limitations

- Scheme effectiveness is assumed, not measured via actual ROI.  
- Real-time constraints (like logistics, stock availability) not incorporated.  
- Cold-start partners receive only default recommendations (no learning).  
- Model assumes independence between partner-product pairs for scheme allocation.

# 5. Edge Cases

- Duplicate Scheme Detection: Prevented same scheme being allocated multiple times to the same product.  
- Partners with Many Products: Handled separately to ensure LP doesn’t over-allocate same schemes across different products.

# 6. Reference Links

https://scikitlearn.org/stable/modules/generated/sklearn.metrics.pairwise.cosine\_similarity.html  
https://coin-or.github.io/pulp/  
https://medium.com/s/story/recommender-systems-collaborative-filtering-d9b75a9dcb49  
https://towardsdatascience.com/linear-programming-using-python-practical-guide-to-pulp-449f3c5f6d9f