A Formal Optimisation Model for Weighting Diverse Implicit Signals in Product Recommender Systems

Joydeep Banik Roy
PGD AI
Indian Institute of Technology Jodhpur
Jodhpur, India
Email address: g25ait1076@iit.ac.in

Nishant Kumar
Department of Electrical Engineering
Indian Institute of Technology Jodhpur
Jodhpur, India
Email address: nishantkumar@iitj.ac.in

***Abstract*—Product Recommender Systems are at the heart of many online e-commerce and retail websites. Popular techniques like Content-based and Collaborative Filtering are used to do such product recommendations at scale. All of these methods rely on a User-Item (U-I) Interaction Matrix for learning latent vector spaces for different users and the products they bought. While explicit feedback (e.g., 5-star ratings) is ideal, it is generally not available, so most systems resort to using implicit signals derived from user behavior events. However, the key challenge is that these signals are not universal; different websites track diverse event types, such as standard product-views, purchase and add-to-carts, or custom business-specific events like download_pdf or click_phone_number. This diversity makes traditional heuristic weights (e.g., view=1, purchase=10) unreliable and non-portable because determining the relative importance of these signals becomes difficult.**

This paper addresses two fundamental questions: *how to determine which of these diverse events are meaningful signals versus noise*, and *how to mathematically derive the optimal weights which are most relevant for deriving the user interaction score analogous to user ratings*. We propose a formal methodology by framing the weight-selection process as a classical constrained optimization problem that finds an optimal vector of weights W combined with implicit feedback data to arrive at a reliable user interaction score.

Keywords—recommender systems, collaborative filtering, user item interaction matrix, latent vector spaces, multi objective recommender systems, product recommender systems, implicit feedback, constrained optimization, e-commerce, retail, content based filtering

I. Introduction

Recommender systems are omnipresent on every website carefully nudging users in the right direction. From Amazon shopping to Youtube videos, they drive revenue for most websites especially in the e-commerce, media and retail space.

Collaborative Filtering uses matrix factorization techniques along with the Alternative Least Squares algorithm to build recommendation systems by decomposing the user-item rating (or interaction) matrix into smaller matrices that represent latent user and item factors. Content based filtering uses the same User-Item (U-I) Interaction matrix to recommend products to users based on their past preferences.

The most important point to note is that the User-Item Interaction Matrix is the key input to all these recommendation engines and the quality of this matrix governs the quality of the recommendations. At its core, the U-I Matrix represents the relationship between users and items. They may be provided explicitly by the user in the form of user ratings. Table A shows the explicit feedback provided by the user where each cell contains a direct rating or review.

Table A: User-Item Matrix with Explicit Feedback

Product	Tshirt	Jeans	Shoes	Jacket	Hat
User 1	5	4	-	1	-
User 2	4	-	3	-	-
User 3	1	-	-	-	3
User 4	-	-	2	-	-

Not every user has purchased every product, so when there are millions of users and thousands of products in a catalog, these matrices become extremely sparse. However, this is not the only problem for these e-commerce websites. The vast majority of customers never leave a review so explicit user ratings are a rarity. Instead we have to rely on what is called implicit feedback. The implicit feedback are actually implicit signals collected based on user events performed on websites like add-to-wishlist, add-to-cart, view and purchases. In most cases we have to calculate a 'good-enough' notion of review score derived from these implicit signals. Unfortunately, although these signals are abundant, they can be extremely noisy mimicking the noisy trail of user behavior. Table B shows the different event types and their frequency for each user.

Table B: User-Item Data with Implicit Feedback

User	Product	Views	Add-to-cart	Purchases
User 1	Tshirt	100	2	1
User 1	Jeans	5	0	0
User 2	Tshirt	7	3	1
User 2	Jeans	1	1	0
User 3	Jeans	30	1	1

As you can see above the core challenge is how to interpret this ambiguous data. Does page view really mean strong interest or an accidental click? How does its value compare to a purchase? This ambiguity is a central problem in building a useful and relevant U-I matrix.

The Flaw in the Heuristic Solution

To solve this ambiguity, many systems adopt a simple heuristic approach. Engineers manually assign static weights based on intuition, such as view=1, add-to-cart=5, and purchase=10. This common practice has two critical, systemic flaws.

First, the weights are arbitrary. There is no mathematical evidence that a purchase is worth exactly 10 views. This guess is almost certainly suboptimal. Second, this method is prone to noise. A malicious bot generating 1000 page view events for an item creates a spam score of 1000, which falsely appears 100 times more important to the model than a real user's single purchase (core 10). The simple heuristic model has no mechanism to distinguish between noise and a true signal.

The CDP : Diverse Schema Complication

CDP stands for Customer Data Platform. As a CDP, our organisation hosts data for clients across dozens of different industries, like retail, football clubs, food delivery apps, and luxury car manufacturers, to name a few. The heuristic model fails magnificently for us. This is because each client has a unique business model and critically, a unique different schema. Therefore, while it is easier to interpret familiar standard events like purchase and add-to-cart and assign weights to them, we don't exactly know what weights to assign when the clients come with their custom event types.

Table C: Sample of Client Event Type Diversity and User Counts from an Actual CDP client dataset

event_value	**unique_users**
page_view	1,219,380
first_visit	956,756
user_needs_configurator_add_tag_lvl1	42,009
download_pdf	18,119
click_phone_number	16,785
purchase	15,225
begin_form_lead	13,407
add_shipping_info	12,438
add_payment_info	12,389
complete_health_questions	4,321
begin_checkout	3,997
video_complete	1,217
add_to_cart	1

This real world data makes the failure of the heuristic approach clear. A purchase (15,225 users) is less frequent than a click_phone_number (16,785 users) and a download_pdf (18,119 users). Furthermore, the dataset contains highly specific business-process events like begin_health_questions (4,535 users) and complete_health_questions (4,321 users).

This data diversity raises the core questions about custom event types:

- How much is complete_health_questions worth? Is it a stronger signal of intent than a purchase?
- How should click_phone_number be weighted against download_pdf?
- How do we treat the page_view event (1.2M users), which looks like 99% noise, versus the add_to_cart event (1 user), which is a statistical anomaly?

It is impossible to eyeball these weights. This proves that a portable, **one-size-fits-all model** for implicit feedback data does not exist, especially when the event types are non-standardized for each client but clearly has deep meaning corresponding to their business. **The optimal weights are a function of the dataset itself.** Although we aim to generate a user review (or rating), what we end up generating can be viewed more like an engagement score.

Temporal Focus

Finally, the importance of event type itself is dynamic in nature. Over a period of time, the relative importance of certain event types may increase or decrease based on how they were collected by the website. For instance, a website creates some default event type whenever the homepage is opened. As a result, this pipeline needs to be run at some agreed upon frequency to mirror this change showcasing the temporal nature of such event types and their overall relevance in the generation of the U-I matrix.

II. Problem Formulation

The system will ingest a set of raw user-item interactions and, as its output, produce an optimal vector of weights.This weight vector, when applied to the interaction data, generates a reliable, implicit User-Item Interaction (U-I) Matrix favorable for use in downstream recommendation models.

Input Data:

The raw interaction data for a given client will be a set of event counts E_{ij}^{k} representing the total number of times a user i performed an event type k (e.g., page_view, download_pdf) on an item (or product) j.

Feature Transformation (Outlier Handling)

As established in the introduction, raw event counts E_{ij}^{k} are highly susceptible to noise and outliers (e.g., bot activity inflating page_view counts into the millions). A model that uses these raw counts will be unstable and produce biased results.

The log-transform perfectly captures the business logic of user intent.The difference between 0 views and 1 view is massive.The difference between 1 view and 10 views is significant.The difference between 100 views and 1,000 views is almost meaningless.The log function $y = log(x)$ has this exact "diminishing returns" shape.

Therefore, we first apply a non-linear log transformation to all raw counts to create a robust feature e_{ij}^{k}:

$$e_{ij}^{k} = log(1 + E_{ij}^{k}) \qquad (1)$$

where:

E_{ij}^{k} is the raw count of k^{th} event type for user i on item j,
e_{ij}^{k} is the transformed, robust feature.

Decision Variables

The decision variables for this optimization problem are the set of K unknown weights, one for each of the K unique event types in a client's dataset. This is represented by the weight vector:

$w = [w_1, w_2, \dots, w_k]$

Engagement Score Formulation

The final, single engagement score S_{ij} for a user i and item j is the weighted linear combination of the transformed event features.

$$S_{ij} = \sum_{k=1}^{n} w_k . e_{ij}^{k} \qquad (2)$$

This S_{ij} score is the calculated value that will be used to populate the final U-I matrix with user engagement score. The core problem is to find the optimal w_k that makes all S_{ij} scores as meaningful as possible.

A. *Objective Function Formulation*

The objective function is a continuous and differentiable function representing the relationship of all K event types with their respective weights.The goal of the optimization is to find the weight vector w that maximizes this function.

$$\max_{w} f(w) = f_{engagement}(w) - \lambda_1 . f_{fair}(w) - \lambda_2 . f_{reg} \qquad (3)$$

Here the objective function encodes the trade-off between three competing goals: maximizing overall engagement, ensuring fairness, and maintaining model stability. The function $f(w)$ is a composite of these three parts. Having said this, it would be appropriate to point out here that the objective function is quadratic in nature and has a concave property which should lead us to one single, global optimum.

Component 1: Total Engagement $f_{engagement}(w)$

$$f_{engagement}(w) = \sum_{i=1}^{N_u} \sum_{j=1}^{N_i} e_{ij}^{k}(w_k) \qquad (4)$$

where, e_{ij}^{k} = Transformed feature k type of implicit feedback, w_k=

The primary goal is to maximize the total engagement captured by the system. We define this as the sum of all individual engagement scores in the U-I matrix.This term rewards weights that assign high value to interactions that are frequent and widespread across the user base.

Component 2: Fairness Penalty $f_{fair}(w)$

A system that is highly skewed by some bot activity or only rewards a few super-active "power users" is biased and undesirable. To enforce fairness and diversity, we introduce a penalty term based on the variance of the total aggregated score per user.

First, let $E_{ij}(w)$ be the total aggregated score for a single user i across all items:

$$E_{ij}(w) = \sum_{j,k=1}^{K} w_k . e_{ij}^{k} \qquad (5)$$

The fairness penalty is the variance of this score distribution across all users:

$$f_{fair}(w) = Var_{user}(E_{ij}(w)) \qquad (6)$$

By subtracting this variance term, the optimizer is penalized for solutions that create highly unequal terms. This term forces the model to find weights that are relevant to a broad spectrum of users, not just the noisy outliers.

Component 3: Complexity Penalty $f_{reg}(w)$

To prevent the optimizer from finding unstable, over-fitted solutions with arbitrarily large weights, we introduce a standard L2 Regularization penalty. This term penalizes the sum of the squares of the weights.

$$f_{reg}(w) = \sum_{k=1}^{K} (w_k^2) = ||w||_2^2 \qquad (7)$$

This ensures a stable solution where no single event has a weight that can grow disproportionately large.

$$\max_{w} f(w) = \left(\sum_{i=1}^{N_u} \sum_{j=1}^{N_i} e_{ij}^{k}(w_k) \right) - \lambda_1 . Var(E_{ij}(w)) - \lambda_2 . \sum_{k=1}^{K} (w_k^2) \qquad (8)$$

where,
$w = [w_1, w_2, \dots, w_k]$, Optimization Variable: Vector of relative weights for implicit feedback types.
K = total number of distinct implicit feedback types

$e_{ij}^{k} = log(1 + E_{ij}^{k})$(Transformed feature k^{th} type of implicit feedback)

E_{ij}^{k} =Raw input data: Magnitude of interaction type k between user i and item j

N_u=Total number of users.

N_i= Total number of items.

$E_{ij}(w) = \sum_{k=1}^{K} w_k \cdot e_{ij}^{k}$ (Encouragement Score)

λ_1,λ_2= Hyperparameters: Tuning parameters to control the influence of the variance penalty and the weight complexity penalty, respectively.

2.2.1. Hyperparameters

The final piece of the objective function $f(w)$ are two key hyperparameters that must be set prior to optimization and reflect the business priorities for a given (problem,solution) pair.

The Fairness Coefficient λ_1 : This parameter controls the trade-off between maximizing raw engagement and enforcing fairness.

A low λ_1 prioritizes maximizing total engagement for a website, even if it comes from noisy signals.

A high λ_1 forces the model to find a fairer solution, heavily penalizing weights that create high user-score variance (i.e., it will learn to ignore active bots on websites).

The Regularization Coefficient λ_2**:** This parameter controls the magnitude of the weights to prevent overfitting. A small, non-zero value (e.g., 0.1) is standard.

B. Constraints

To ensure the resulting weights $w*$ are feasible, interpretable, and aligned with business logic, the optimizer must operate within the following set of linear constraints.

1. **Normalization (Equality Constraint):** To ensure the weights represent a relative "importance" distribution and to create a bounded feasible region, we constrain the sum of all weights to 1.

$$\sum_{i=1}^{n} w_j = 1$$

2. **Non-Negativity (Inequality Constraint):** We assume that all tracked user interactions are, at worst, neutral signals (worth 0) and never detrimental to a preference score. This enhances interpretability.

$$w_j \geq 0 \, for \, all \, j \in [1, n]$$

3. **Logical Hierarchy (Inequality Constraints):** This is a critical set of constraints that injects human business logic into the model. For any dataset, we can define a known hierarchy of intent. For example:

   For a standard e-commerce dataset (Case 1), we can enforce that a purchase is more valuable than an add-to-cart, which is more valuable than a view:

$$w_{purchase} \geq w_{cart}$$

$$w_{cart} \geq w_{view}$$

For the insurance dataset in Table C, we can enforce that *completing* a form is more valuable than *beginning* it:

$$w_{complete-health-questions} \geq w_{begin-health-questions}$$

These constraints dramatically reduce the search space and guarantee that the final weights are sensible and explainable.

III. Test Conditions

To ensure that the framework is adaptable and robust, the model will be run on three distinct test cases. Each case simulates a common, real-world client scenario. The resulting optimal weights $w*$ will be analyzed to validate the model's effectiveness.

A. Case-1: The Balanced E-Commerce (Baseline)

Description: A standard, realistic retail dataset with a logical progression of events (e.g., many views, moderate add-to-carts, few purchases). All signals are expected to be meaningful.

Actual Data: `Views`(10M), `Add-to-Carts` (500k), `Purchases` (100k).

Expected Outcome: The optimizer will find a balanced set of weights that respects the hierarchy constraints, such as $w_{purchase} > w_{cart} > w_{view} > 0$

B. Case 2: The Bot-Infested / Noisy (Outlier Test)

Description: A dataset (similar to Table 1) where one low-intent event (`page_view`) is massively inflated by bot traffic, making it a "noise" signal.

Hypothetical Data: `Page-Views` (500M), `Add-to-Carts` (200k), `Purchases` (50k).

Expected Outcome: When the fairness coefficient α is set sufficiently high, the optimizer will be heavily penalized by the high variance. It will be forced to learn that the `page_view` signal is noise and will produce an optimal weight $w_{view} \approx 0$. This test verifies the model's automatic noise-detection capability.

C. Case 3: The Niche-Intent B2B (Adaptability Test)

Description: A non-standardised client (e.g., finance, insurance) where "purchase event" is rare. The true, high-intent signals are custom business or domain specific event types, like those found in Table C.

Data with Custom Event Types: `Page Views` (5M), `Download_PDF` (50k), `Begin_Form_Lead` (20k), `Complete_Form_Lead` (10k).

Expected Outcome: The optimizer will assign minimal weight to w_{view} and should correctly identify the high-intent custom events as the most valuable. The expected result is a hierarchy like $w_{complete-form} > w_{begin-form} > w_{download-pdf} > 0$. This test will be the most interesting due to its generalization capability. In many ways, this is the main objective of this paper. To understand and interpret the relationship between different custom events and how they influence customer behavior and preferences.

Additionally, we will run many different variations of this dataset based on data availability and CDP permissions. Both the use cases of variable data schema and temporal focus will be covered here as well.

References

[1] Yong Zhenga, David (Xuejun) Wang, "A Survey of Recommender Systems with Multi-Objective Optimization" Neurocomputing 474(3), December 2021.

[2] J. Clerk Maxwell, Improving top-n recommendation techniques using rating variance, Proceedings of the 2008 ACM Conference on Recommender Systems: October 2008.

[3] I. S. Jacobs and C. P. Bean, "Fine particles, thin films and exchange anisotropy," in Magnetism, vol. III, G. T. Rado and H. Suhl, Eds. New York: Academic, 1963, pp. 271–350.

[4] K. Elissa, "Title of paper if known," unpublished.

[5] R. Nicole, "Title of paper with only first word capitalized," J. Name Stand. Abbrev., in press.

[6] Y. Yorozu, M. Hirano, K. Oka, and Y. Tagawa, "Electron spectroscopy studies on magneto-optical media and plastic substrate interface," IEEE Transl. J. Magn. Japan, vol. 2, pp. 740–741, August 1987 [Digests 9th Annual Conf. Magnetics Japan, p. 301, 1982].

[7] M. Young, The Technical Writer's Handbook. Mill Valley, CA: University Science, 1989.