Collaborative Filtering with Matrix Factorization

Book Recommendation System

Hardcover Book Friend Finder

January 2, 2026

Outline

Overview

Goal

Build a personalized book recommendation system using collaborative filtering

Dataset:

- 1,000 users collected
- 45,203 books
- 95.75% sparse

Results:

- 8.83% precision@10
- 308% better than pure collaborative filtering
- 5 second training time

Dataset Statistics

Raw Dataset:

- 1,000 users collected
- 45,203 books
- Avg: 84.4 books/user
- Median: 1.0 books/user

After Filtering:

- 246 users (24.6%)
- 2,547 books (5.6%)
- Avg: 113.8 books/user
- 26,598 interactions

Filtering Criteria

- **Users**: ≥ 20 book interactions
- **Books**: ≥ 5 users
- **Why?** Better signal-to-noise ratio

Matrix Sparsity: 95.75% (626,562 possible, 26,598 actual)

Feedback Distribution

26,598 total interactions broken down by type:

Type	Count	Percentage
Read	16,340	61.4%
Want to Read	12,102	45.5%
Currently Reading	289	1.1%
DNF	-	-

Implicit Feedback Impact

Adding "want to read" + "currently reading" increased training data by 75%!

The Core Idea

Matrix Decomposition

Factor the sparse user-book matrix Y into two smaller matrices:

$$Y_{m \times n} \approx \sigma(X_{m \times k} \cdot W_{n \times k}^{T} + b)$$

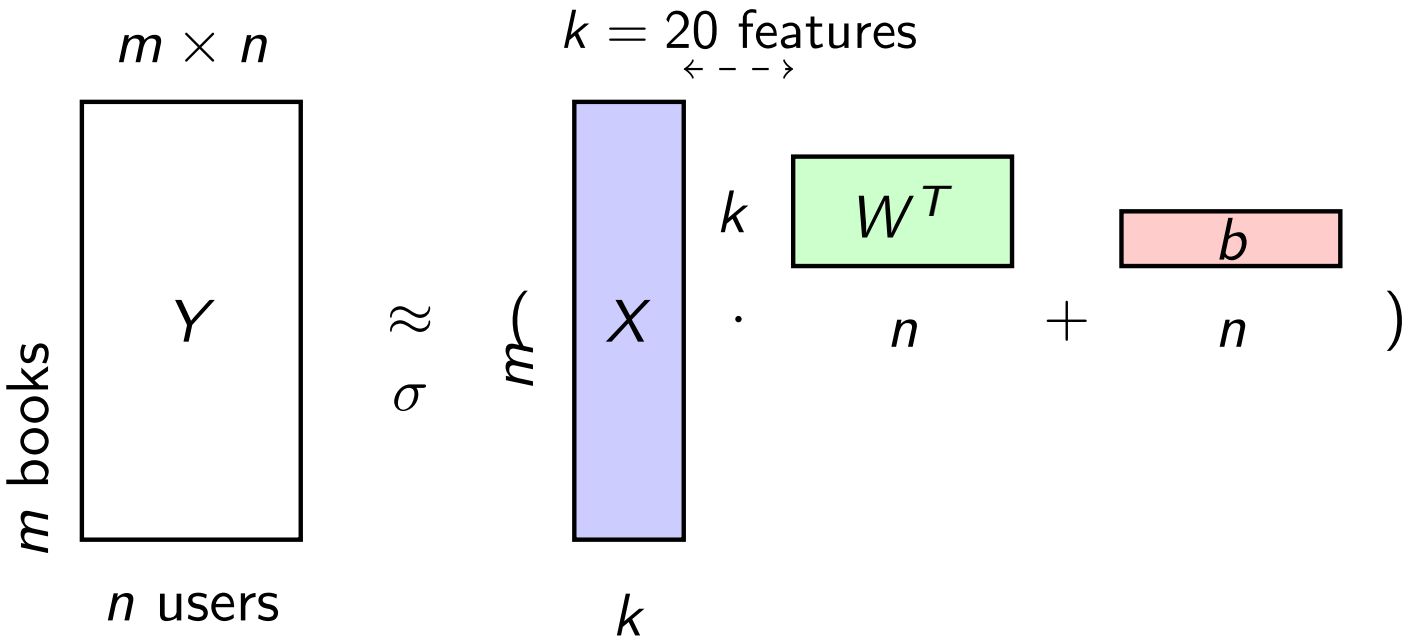
Dimensions:

- $m = 2,547$ books
- $n = 246$ users
- $k = 20$ features

Matrices:

- X: Book features
- W: User features
- b: User bias
- σ: Sigmoid activation

Matrix Multiplication Diagram



Latent Features

What are the $k = 20$ latent features?

Book Features (X)

Each book gets a 20-dimensional vector representing:

- Genre (fantasy, sci-fi, literary fiction)
- Reading level
- Themes and topics
- Popularity

User Features (W)

Each user gets a 20-dimensional vector representing:

- Affinity for each latent dimension
- Reading preferences
- Taste profile

The Masking Approach

Problem: 95.75% of the matrix is unread books!

Solution: Mask Matrix R

$$R_{ij} = \begin{cases} 1 & \text{if user } j \text{ interacted with book } i \\ 0 & \text{otherwise} \end{cases}$$

Rating Scale with Implicit Feedback

$$Y_{ij} = \begin{cases} 1.0 & \text{read with rating } \geq 3 \\ 0.7 & \text{currently reading} \\ 0.3 & \text{want to read} \\ 0.0 & \text{disliked or DNF} \\ 0.5 & \text{unread (ignored!)} \end{cases}$$

Masked Loss Function

Key Innovation: Only train on known interactions!

Cost Function

$$J = \sum_{i,j} R_{ij} \left(\sigma(x_i^T w_j + b_j) - Y_{ij} \right)^2 + \frac{\lambda}{2} \left(\|X\|_F^2 + \|W\|_F^2 \right)$$

- R_{ij} masks out unread books (where $Y_{ij} = 0.5$)
- Without masking, 95.75% unread would dominate the loss
- Model would just learn the global average

Result: Model learns from actual preferences, not missing data!

Optimization

Algorithm: Gradient descent with Adam optimizer

Configuration

- Learning rate: $\alpha = 0.1$
- Regularization: $\lambda = 1.0$
- Iterations: 300
- Features: $k = 20$

Training Time (on 246 users)

- Data loading: 1.40s
- Matrix building: 0.02s
- Model training: 3.26s
- User clustering: 0.04s
- **Total: 4.72s**

Scalability

How does it scale?

Users	Active	Interactions	Time
1,000	246	26,598	4.4s
5,000	1,230	132,990	17.0s
10,000	2,460	265,980	32.7s
100,000	24,600	2,659,800	5.2 min

Complexity: $O(\text{iterations} \times k \times m \times n)$
Rate: 12.6ms per user (linear scaling)

Hybrid Model

Pure collaborative filtering only achieved 2.45%!

Solution: Hybrid Approach

$$\text{score}_{ij} = 0.5 \cdot \text{popularity}_i + 0.5 \cdot \sigma(x_i^T w_j + b_j)$$

Why this works:

- **Popularity (50%)**: Ensures quality recommendations
- **Collaborative (50%)**: Adds personalization
- **Implicit feedback**: +75% more training data

Performance Comparison

Method	Precision@10
Pure collaborative filtering	2.16%
Improved collaborative (filtered)	2.45%
Popularity baseline	7.56%
Hybrid (50/50)	**8.83%**

Improvement

- +308% vs pure collaborative filtering
- +17% vs popularity baseline

Netflix vs. Hardcover

Aspect	Netflix	Hardcover
Rating scale	$\{-1, 1, 0\}$	$\{0, 0.3, 0.7, 1, 0.5\}$
Unranked	0 = unranked, predicted	0.5 = masked out
Activation	Linear ($\in \{-1, 1\}$)	Sigmoid ($\in [0, 1]$)
Loss	All entries	Only known (R=1)
Sparsity	95-98%	95.75%

Key difference: We mask unread books, Netflix predicts them
Why? Our data is too sparse to predict missing values reliably

User Clustering

Goal: Find users with similar reading preferences

Method: K-means Clustering

- Cluster on L2-normalized user embeddings W
- Tested $K \in [3, 15]$ using silhouette score
- Optimal: $K = 13$ clusters (score: 0.084)

Note:

- $k = 20$ latent **features**
- $K = 13$ **clusters**

Examples:

- Classic YA & Fantasy
- Sci-Fi Enthusiasts
- Literary Fiction

Friend matching: Cosine similarity within clusters

$$\text{sim}(u_i, u_j) = \frac{w_i \cdot w_j}{\|w_i\| \|w_j\|}$$

Cluster Size Distribution

13 clusters with varying sizes:

ID	Size	%
7	52 users	21.1%
2	35 users	14.2%
10	21 users	8.5%
5	20 users	8.1%
1,3	19 users each	7.7%
13	17 users	6.9%
6	15 users	6.1%
4,8	12 users each	4.9%
11	11 users	4.5%
9	9 users	3.7%
12	4 users	1.6%

Largest cluster: 21.1% of users — **Smallest:** 1.6%

Most Popular Books

Top 10 books in the dataset:

#	**Title**	**Users**
1	1984	106 (43.1%)
2	Harry Potter and the Sorcerer's Stone	88 (35.8%)
3	Project Hail Mary	80 (32.5%)
4	The Hobbit	80 (32.5%)
5	Animal Farm	76 (30.9%)
6	The Hunger Games	76 (30.9%)
7	To Kill A Mockingbird	74 (30.1%)
8	Dune	67 (27.2%)
9	The Hitchhiker's Guide to the Galaxy	64 (26.0%)
10	Mistborn: The Final Empire	63 (25.6%)

Key Innovations

1. **Masking** unread books to avoid learning from noise
2. **Implicit feedback** to increase training signals by 75%
3. **Hybrid ensemble** balancing popularity and personalization
4. **Low-dimensional embeddings** ($k = 20$) capturing preferences
5. **Clustering** for friend matching (13 reading groups)

Summary

Achievement

8.83% precision@10 for 246 active readers

Pros:

- Fast training (5s)
- Scales linearly
- Personalized
- Friend matching

Future Work:

- Content features
- More implicit feedback
- Neural networks
- More users

Questions?