Data Science Lab

Collaborative Filtering for Movie Recommendations

Team: better_than_random

Anthony Boulos
Rebecca El Chidiac
Nadezhda Zhukova

October 21, 2025

1 Problem statement

We are given a partially observed rating matrix $R \in \mathbb{R}^{m \times n}$, where each entry R_{ui} corresponds to the rating of user u for item i. Most entries are unobserved, leading to strong sparsity and the presence of "cold" users or items with few interactions. The objective is to estimate the missing ratings $\hat{R}_{ui}$ for unseen (u, i) pairs, allowing personalized recommendations.

Model performance is measured using the Root Mean Squared Error (RMSE) between predicted and true ratings on a held-out validation set:

$$\text{RMSE} = \sqrt{\frac{1}{|\Omega_{\text{test}}|} \sum_{(u,i) \in \Omega_{\text{test}}} (R_{ui} - \hat{R}_{ui})^2}.$$

2 Baseline : Matrix Factorisation with Alternating Least Squares (ALS)

Let the observed ratings matrix be denoted by $R \in \mathbb{R}^{m \times n}$. The standard ALS factorizes R as

$$R \approx UV^\top,$$

with user latent factors $U \in \mathbb{R}^{m \times k}$ and item latent factors $V \in \mathbb{R}^{n \times k}$.

U and V are set by minimizing the regularized squared reconstruction error (Ω denotes only over observed entries):

$$\min_{U,V} \sum_{(u,i) \in \Omega} (R_{ui} - U_u^\top V_i)^2 + \lambda \left(\|U\|_F^2 + \|V\|_F^2 \right),$$

where λ is the regularization coefficient and k is the latent dimension (number of features).

Optimization. ALS alternates between updating user and item factors via regularized least squares, for a fixed number of iterations n_{iter}:

$$U_u \leftarrow (V_\Omega^\top V_\Omega + \lambda I)^{-1} V_\Omega^\top R_{u,\Omega}, \qquad V_i \leftarrow (U_\Omega^\top U_\Omega + \lambda I)^{-1} U_\Omega^\top R_{\Omega,i}.$$

This alternating procedure ensures that, at each step, one of U or V is optimized while the other is held fixed, yielding a simple yet effective algorithm for large-scale collaborative filtering.

3 Enhancement of the Matrix Factorization

3.1 Addition of bias

Beyond plain factorization, we include standard bias terms to capture global and per-entity rating tendencies. We model ratings as

$$\hat{R}_{ui} = U_u^\top V_i + \mu + b_u[u] + b_i[i],$$

where μ is the global mean, b_u and b_i are user and item biases. Training minimizes ridge-regularized squared error on observed entries:

$$\min_{U,V,b_u,b_i,\mu} \sum_{(u,i) \in \Omega} \left(R_{ui} - U_u^\top V_i - \mu - b_u[u] - b_i[i]\right)^2 + \lambda_u \|U\|_F^2 + \lambda_v \|V\|_F^2 + \lambda_{bu} \|b_u\|_2^2 + \lambda_{bi} \|b_i\|_2^2.$$

The ALS updates become:

$$U_u \leftarrow \arg\min_x \|r_u - Z_u x\|_2^2 + \lambda_u \|x\|_2^2,$$

$$b_u[u] \leftarrow \frac{\sum_{i \in \Omega(u)} (R_{ui} - \mu - b_i[i] - (U_u^\top V_i))}{|\Omega(u)| + \lambda_{bu}},$$

and analogously for items.

3.2 Popularity-based Item Regularization

To prevent over-penalizing rare items and overfitting frequent ones, we replace the uniform item ridge λ_v with an item-specific value depending on popularity (number of ratings) c_i: $\lambda_{v,i} = \frac{\lambda_v}{\sqrt{c_i+1}}$ so the item update becomes

$$V_i \leftarrow \arg\min_x \ \|r_i - U_i x\|_2^2 + \lambda_{v,i}\|x\|_2^2.$$

Intuitively, popular items get smaller effective regularization, letting the model fit their richer signal, while very sparse items are regularized more strongly.

4 Integration of Movie Features to the representation

4.1 Genre-Enriched Item Factors

When genres information is available, a correction term is included so that

$$R \approx U(V + GW_g)^\top \ + \ \mu \ + \ b_u[u] \ + \ b_i[i],$$

where:

- $G \in \mathbb{R}^{n \times d}$ is the one-hot or multi-hot encoding of item genres,
- $W_g \in \mathbb{R}^{d \times k}$ is the projection of genres into the latent space.

4.2 Integration of Years

We also try adding a correction term to integrate the information we have on the release year of each movie.

$$R \approx U(V + GW_g + YW_y)^\top \ + \ \mu \ + \ b_u[u] \ + \ b_i[i],$$

Here, Y encodes year-related information for each item and can take two forms:

- a **continuous standardized feature**, where each year is centered and scaled as $\mathrm{Y}_i = \frac{\mathrm{year}_i - \mu_y}{\sigma_y}$;
- or a **binned categorical representation**, where release years are grouped into quantile-based bins, yielding a one-hot matrix $\mathrm{Y} \in \mathbb{R}^{n \times p}$.

The corresponding projection $W_y \in \mathbb{R}^{1 \times k}$ is also estimated periodically.

5 Graph-Regularized Matrix Factorization

We extend the standard Alternating Least Squares (ALS) algorithm for matrix factorization by incorporating Laplacian regularization. The purpose of this regularization is to encourage items that are similar (as measured by a graph derived from genre information) to have similar latent representations as introduced by Belkin and Niyogi [1].

5.1 Construction of the Item Similarity Graph

To incorporate genre information into the latent representations, we define an item graph for which the weights are represented by a similarity matrix based on the genre encoding. The similarity matrix $S \in \mathbb{R}^{n \times n}$ is constructed as follows:

1. **Row-Normalization:** The genre matrix $G \in \mathbb{R}^{n \times d}$ is normalized by dividing each row by its row sum, ensuring that each item has unit contribution from its genres. This normalization avoids bias towards items with a large number of genre tags.
2. **Cosine Similarity:** After normalization, the cosine similarity is computed between each pair of items. Cosine similarity is chosen because it quantifies the orientation between genre vectors, and it is differentiable.

3. **Diagonal Zeroing:** This step ensures that self-similarities (which are trivially equal to 1) do not influence the regularization.
4. **Sparsification:** To mitigate computational costs and potential noise from weak similarities, only the top-k strongest similarities per item are retained.
5. **Symmetrization:** Finally, to ensure that the resulting similarity matrix is symmetric, the matrix is symmetrized using the element-wise maximum: $S = \max(S, S^\top)$.

5.2 Addition of a Laplacian smoothness term to the MF objective

To leverage the similarity structure between items, we introduce Laplacian regularization. Given an item similarity matrix $S \in \mathbb{R}^{n \times n}$ computed from the genres, we define the graph Laplacian L as:

$$L = D - S,$$

where D is the diagonal degree matrix with entries $D_{ii} = \sum_{j=1}^{n} S_{ij}$.

We add a Laplacian smoothness term to the standard Matrix Factorization objective:

$$\min_{U,V,b_u,b_i,\mu} \sum_{(u,i)\in\Omega} \left(R_{ui} - U_u^\top V_i - \mu - b_u[u] - b_i[i]\right)^2 + ... + \alpha \operatorname{Tr}(V^\top L_v V),$$

where α controls the strength of the smoothness regularization.

The penalty term can be rewritten as: $\operatorname{Tr}(V^\top L_v V) = \frac{1}{2} \sum_{i,j} S_{ij} \|V_i - V_j\|^2$,

So, this regularization promotes smoothness across the item latent factors. That is, if two items have a high similarity (large S_{ij}), their latent representations V_i and V_j are encouraged to be similar, thereby effectively integrating content information into the collaborative filtering model.

The item update in the ALS framework is then modified to include the graph Laplacian terms:

$$V_i \leftarrow \left(U_{\Omega_i}^\top U_{\Omega_i} + (\lambda + \alpha D_{ii}) I\right)^{-1} \left(U_{\Omega_i}^\top R_{\Omega_i} + \alpha \sum_j S_{ij} V_j\right).$$

This formulation, aligned with the framework proposed by Li et al. [2], leads to a more robust factorization by leveraging the underlying item similarity structure derived from the genres.

6 Results and Interpretation

6.1 Experimental setup

We evaluate six matrix–factorization (MF) variants 1 with 3-fold entry-wise cross validation on the merged ratings matrix. All observed entries $\Omega = \{(u, i) \mid R_{ui} \text{ observed}\}$ are randomly shuffled with a fixed seed and partitioned into three disjoint folds. For a given fold f, the train matrix contains all observed entries not in f, the test matrix contains the entries in f, and all other cells remain NaN.

Model	Pop-reg	Laplacian	Genres	Years	Description
`MF_baseline`	No	No	No	No	Baseline matrix factorization.
`MF_popreg`	Yes	No	No	No	Popularity-aware regularization.
`MF_laplacian`	No	Yes	No	No	Graph smoothness regularization.
`MF_genres`	No	No	Yes	No	Includes genre features.
`MF_genres_popreg`	Yes	No	Yes	No	Genre features with popularity-aware reg.
`MF_genres_years_cont`	No	No	Yes	Cont.	Genre and continuous year features.
`MF_genres_years_bins_1`	No	No	Yes	1 bin	Genre and binned year features.

Table 1: All models tested (all include user/item biases and L_2 regularization on U, V).

6.2 Hyperparameters and fitting

We tuned all models in two stages: a coarse Bayesian search over wide, log–scaled ranges followed by a small grid around the best region to stabilize choices. During tuning we used the provided `test_ratings.npy` as a validation signal to select ranges for k, regularizations, and (when applicable) the Laplacian weight α. A consistent pattern emerged: without a graph prior the optimum collapsed to very low rank (k=1), while with a Laplacian prior the best setting shifted to high rank (k=147), where smoothness supports richer item factors. We also observed mild interactions (larger k prefers stronger λ_v/λ_{W_g}; α partly substitutes λ_v).

All results reported in this paper are obtained on an independent entry–wise 3–fold cross-validation built from the merged ratings matrix, with folds fixed before evaluation and no further retuning. This is not fully nested CV, so slight selection bias may remain; we mitigate this by reporting mean±std across folds, running paired significance tests vs. the baseline, and releasing the sweep curves for transparency (available in the project GitHub repository).

6.3 Global performance

Model	Train RMSE	Test RMSE	Time (s)
MF_baseline	0.7737 ± 0.0018	0.8666 ± 0.0018	4.53 ± 0.10
MF_popreg	0.7576 ± 0.0017	0.8682 ± 0.0008	4.43 ± 0.09
MF_laplacian	0.3572 ± 0.0004	0.8730 ± 0.0035	53.64 ± 1.35
MF_genres	0.7742 ± 0.0024	0.8678 ± 0.0011	7.03 ± 0.46
MF_genres_popreg	0.7595 ± 0.0052	0.8721 ± 0.0082	6.70 ± 0.03
MF_genres_years_cont	0.7733 ± 0.0005	0.8654 ± 0.0028	6.72 ± 0.17
MF_genres_years_bins_1	0.7730 ± 0.0014	0.8659 ± 0.0023	6.79 ± 0.08

Table 2: Global performance (mean ± std across folds).

Figure 1 and Table 2 compare mean test RMSE (error bars = ±1 sd across folds). All content–aware variants outperform the plain `MF_baseline`; the best average score is obtained by `MF_genres_years_cont`. Popularity–aware regularization alone (`MF_popreg`) yields a small but consistent gain over the baseline. In contrast, `MF_genres_popreg` shows higher variance and no average improvement, indicating that coupling content with strong head/tail regularization can over-constrain V on this dataset. The Laplacian model (`MF_laplacian`) is slightly worse on average—graph smoothness helps some items but appears to underfit globally at the tuned α.

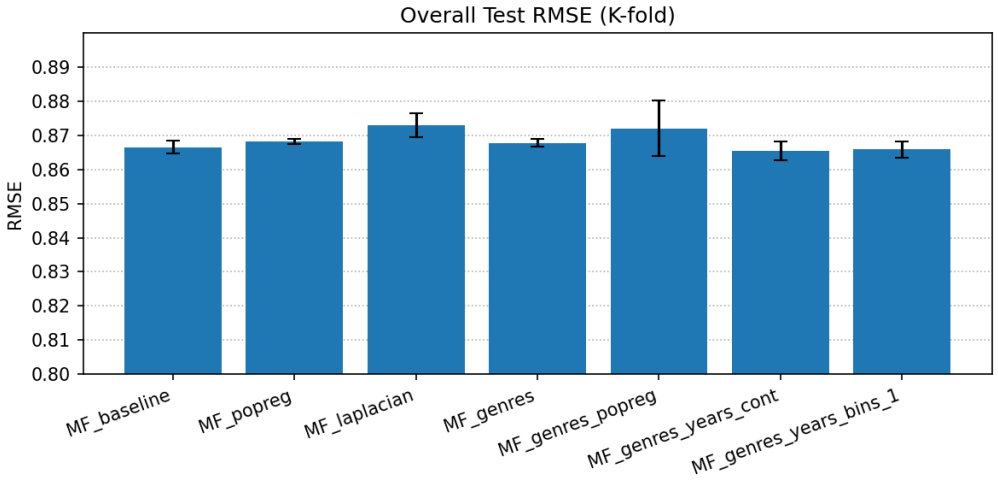

Figure 1: Overall test RMSE across models (K-fold).

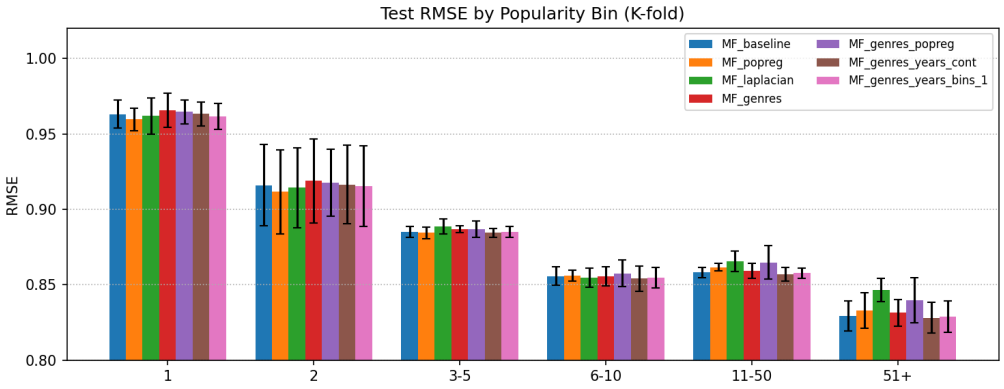

Figure 2: Test RMSE by item-popularity bin (K-fold).

Figure 2 breaks results by item popularity (defined on the train split). Benefits of side information are largest in the cold-item regime (1–2 ratings), where collaborative signals are weakest; as popularity grows, gaps narrow and methods converge, consistent with factors being well constrained by data. Notably, `MF_genres_years_cont` remains competitive across all bins, while `MF_laplacian` only matches others on head items.

6.4 Are improvements significant?

We assess significance with paired t-tests computed on per–rating errors, fold by fold and within each popularity bin, comparing every model to $MF\_baseline$. In the cold–item regime (1–2 train ratings) several variants yield statistically reliable gains, confirming that side information helps where CF is weakest. As item popularity increases, effect sizes shrink and many comparisons are no longer significant—consistent with Figure 2, where models converge on head items. Exact FDR-corrected p-values for all (model, bin) pairs are provided in the project GitHub repository.

6.5 Hyperparameter sensitivity of $MF\_genres\_years\_cont$

A comprehensive analysis of hyperparameter sweeps is available in the results section of our GitHub repository. The rank sweep shows a shallow local minimum at $k = 1$; performance then degrades for very small ranks and steadily improves again, reaching a broad, near-flat optimum for $k \approx 400$. Increasing the number of ALS iterations consistently lowers test RMSE across the explored range. The core factor shrinkage terms exhibit clear U-shapes: λ_u and λ_v work best in the low–mid tens. For biases, lighter is preferable: λ_{bi} is most effective around ~ 3, while λ_{bu} is best near the smallest tested values $\sim$ 1–3 and degrades as it grows. On the side projections, genres benefit from moderate λ_wg (about 50–150), whereas the year projection is largely insensitive — λ_{wy} stays flat over several orders of magnitude with only a slight uptick at 10^4. Finally, updating W_g, W_y less frequently (larger $update\_w\_every$) yields a small but consistent gain, likely by reducing alternating-update noise.

7 Limitations & Future Work

While the proposed extensions to matrix factorization improved accuracy and robustness, some limitations remain. The improvements in RMSE, though consistent, are moderate, suggesting that the side information used captures only part of the variability in user preferences.

Future work could explore richer item representations (e.g., embeddings from movie plots) and more expressive architectures such as graph neural networks that jointly learn the similarity structure. Finally, an end-to-end hybrid model combining user–item graph regularization with deep embeddings could further enhance generalization in sparse or cold-start regimes.

References

[1] M. Belkin and P. Niyogi, "Laplacian eigenmaps and spectral techniques for embedding and clustering," in *Advances in Neural Information Processing Systems (NeurIPS)*, MIT Press, 2001, pp. 585–591.

[2] Y. Li, J. Lee, and Y. Lee, "Matrix factorization with graph regularization for collaborative filtering," *Proceedings of the 24th ACM international conference on information and knowledge management*, pp. 79–88, 2015.