Université Dauphine - PSL

Master 2 Intelligence Artificielle, Systèmes, Données (IASD)

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 Methods

2.1 Baseline: Matrix Factorization with Alternating Least Squares (ALS)

Given an observed ratings matrix $R \in \mathbb{R}^{m\times n}$, ALS approximates R as:

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

where:

- $U \in \mathbb{R}^{m\times k}$ and $V \in \mathbb{R}^{n\times k}$ are user and item latent factors, respectively,
- k is the latent dimension, and λ is the regularization coefficient.

The objective is to minimize the regularized squared reconstruction error over observed entries Ω:

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

To minimize this objective, the ALS algorithm alternates between updating U and V via regularized least squares for n_{iter} iterations:

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

2.2 Enhancements to Matrix Factorization

2.2.1 Incorporating Bias Terms

To capture global and per-entity rating tendencies, we extend the model to include bias terms:

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

where:

- μ is the global mean,
- b_u and b_i are user and item biases, respectively.

The objective becomes:

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

In this case, ALS Updates become:

$$U_u \leftarrow \arg\min_x \|r_u - Z_u x\|_2^2+\lambda_u\|x\|_2^2,$$
$$b_u[u] \leftarrow \frac{\sum_{i\in\Omega(u)}(R_{ui}-\mu-b_i[i]-U_u^\top V_i)}{|\Omega(u)|+\lambda_{bu}}.$$

2.2.2 Popularity-Based Item Regularization

To address the imbalance between popular and rare items, we introduce item-specific regularization based on popularity (number of ratings c_i):

$$\lambda_{v,i} = \frac{\lambda_v}{\sqrt{c_i + 1}}.$$

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.

2.3 Integration of Movie Features

2.3.1 Genre-Enriched Item Factors

When genre information is available, we enrich the item factors with a genre-specific correction term:

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

where:

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

2.3.2 Integration of Release Year

We further incorporate release year information using two approaches:

- **Continuous standardized feature**:
$$Y_i = \frac{\text{year}_i - \mu_y}{\sigma_y}.$$
- **Binned categorical representation**: Release years are grouped into quantile-based bins, yielding a one-hot matrix $Y \in \mathbb{R}^{n\times p}$.

The model becomes:

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

where $W_y \in \mathbb{R}^{1\times k}$ is the projection of year information into the latent space.

2.4 Graph-Regularized Matrix Factorization

We extend the standard 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].

2.4.1 Construction of the Item Similarity Graph

To leverage genre information, we construct an item similarity graph:

1. **Row-normalize** the genre matrix G to ensure equal contribution from each genre.
2. Compute **cosine similarity** between items.
3. **Zero the diagonal** to exclude self-similarities.
4. **Sparsify** the graph by retaining only the top-k strongest similarities per item.
5. **Symmetrize** the similarity matrix S using $S = \max(S, S^\top)$.

2.4.2 Laplacian Smoothness Regularization

To leverage the similarity structure between items, we introduce Laplacian regularization as a term that we add to the objective:

$$\min_{U,V,b_u,b_i,\mu} \sum_{(u,i)\in\Omega} \Big(R_{ui} - U_u^\top V_i - \mu - b_u[u] - b_i[i]\Big)^2 + \alpha\, \mathrm{Tr}(V^\top L V),$$

where:

- $L = D - S$ is the graph Laplacian,
- D is the diagonal degree matrix with entries $D_{ii} = \sum_{j=1}^{n} S_{ij}$,
- α controls the strength of the smoothness regularization.

The penalty term encourages similar items to have similar latent representations:

$$\mathrm{Tr}(V^\top L V) = \frac{1}{2} \sum_{i,j} S_{ij} \|V_i - V_j\|^2.$$

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 becomes:

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

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.

3 Results and Interpretation

3.1 Experimental setup

We evaluate six matrix–factorization (MF) variants[1] (Table 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.

To assess statistical significance, we perform paired t-tests on per-rating errors for each fold and popularity bin, comparing all models against the MF_baseline. P-values are adjusted using the Benjamini-Hochberg False Discovery Rate (FDR) correction to control for multiple comparisons. The complete set of FDR-corrected p-values for all model and bin comparisons is available in the project GitHub repository.

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).

3.2 Hyperparameter tunning

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)

[1]While a full ablation study could have clarified the individual contributions of each feature (e.g., genres, years, or graph regularization), we opted for a comparative evaluation of some model variants due to time and computational constraints. Preliminary experiments revealed that no single model dominated across all metrics (Table 2), and hyperparameter sensitivity suggested that isolated feature effects might not generalize robustly. Thus, we prioritized comparing some tested model performance while acknowledging this as a limitation for future work.

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).

3.3 Global performance

Figure 1 and Table 2 present the test RMSE results across all models, evaluated via 3-fold cross-validation. The baseline MF model achieves a test RMSE of 0.8666, significantly outperforming naive baselines (random: 1.4607, mean: 1.0347). Some enhanced models incorporating side information (genres and release years) show modest but consistent improvements, with MF_genres_years_cont achieving the lowest test RMSE (0.8654), though this improvement is not statistically significant after BH-FDR correction.

Model	Train RMSE	Test RMSE	Unpopular Test RMSE	Time (s)
random_prediction	1.4654 ± 0.0047	1.4607 ± 0.0050***	1.5097 ± 0.0316**	0.02 ± 0.00
mean_prediction	1.0347 ± 0.0019	1.0347 ± 0.0038***	1.1239 ± 0.0172**	0.01 ± 0.00
MF_baseline	0.7737 ± 0.0018	0.8666 ± 0.0018	0.9631 ± 0.0092	4.44 ± 0.12
MF_popreg	0.7576 ± 0.0017	0.8682 ± 0.0008	0.9596 ± 0.0077	4.52 ± 0.28
MF_laplacian	0.3572 ± 0.0004	0.8730 ± 0.0035	0.9619 ± 0.0121	51.44 ± 2.21
MF_genres	0.7742 ± 0.0024	0.8678 ± 0.0011	0.9657 ± 0.0112	6.71 ± 0.20
MF_genres_popreg	0.7595 ± 0.0052	0.8721 ± 0.0082	0.9647 ± 0.0079	6.74 ± 0.21
MF_genres_years_bins_1	0.7730 ± 0.0014	0.8659 ± 0.0023	0.9615 ± 0.0087	6.61 ± 0.22
MF_genres_years_cont	0.7733 ± 0.0005	0.8654 ± 0.0028	0.9632 ± 0.0081	6.63 ± 0.16

Table 2: Overall performance (mean ± sd over 3 folds).Naive baselines (random, mean) are evaluated on the same folds for context. Superscripts mark a statistically significant difference vs.MF_baseline on test RMSE after BH–FDR at α=0.05. Adjusted p-values for the naive baselines: random 2.5×10^{-4}, mean 9.0×10^{-5}. The "Unpopular Test RMSE" column reports the test RMSE on the least popular items (the ones that have only 1 rating).

MF_laplacian exhibits the lowest train RMSE (0.3572) but the highest test RMSE (0.8730), indicating possible overfitting probably due to excessive smoothness. It is also computationally expensive (51.44s vs. 5–7s for other models), but not as a method itself, just because the latent dimension used for it was way larger than for others. Meanwhile, MF_popreg performs best on unpopular items (RMSE: 0.9596), highlighting the value of popularity-awareness in cold-start scenarios.

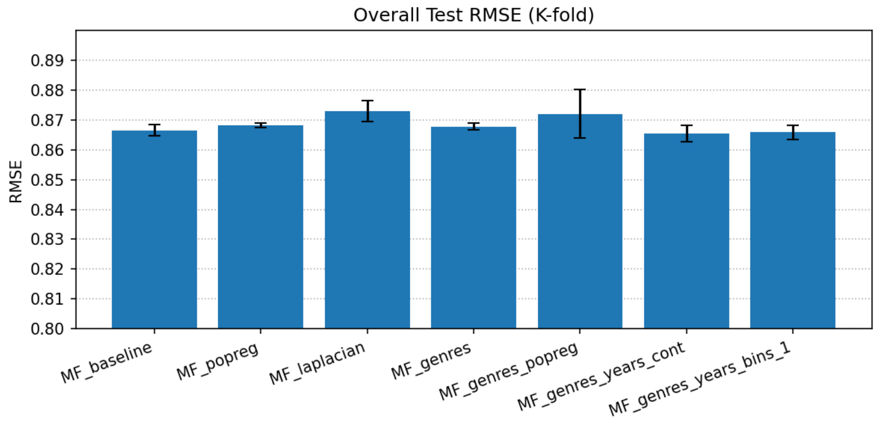

Figure 1: Overall test RMSE across models (3-fold). Error bars represent ±sd across folds.

Figure 2 illustrates test RMSE across item-popularity bins, revealing showing that RMSE is inverse correlated with the popularity of films. For unpopular items (1-2 ratings), MF_popreg outperforms all other models, suggesting that popularity-aware regularization is most effective in extreme cold-start scenarios. As items gain more ratings (3-5 ratings), MF_genres_years_cont and MF_genres_years_bins_1 emerge as the top performers, indicating that genre and year features become increasingly valuable with slightly more collaborative data. Throughout these bins, MF_laplacian clearly performes better on the unpopular items while struggles with the popular ones. These trends, although insignificant most likely due to the low $k = 3$ in k-fold cross-validation, underscores the importance of tailoring model enhancements to item popularity: while popularity-aware regularization excels for the coldest starts, genre and year features prove more beneficial as collaborative signals grow.

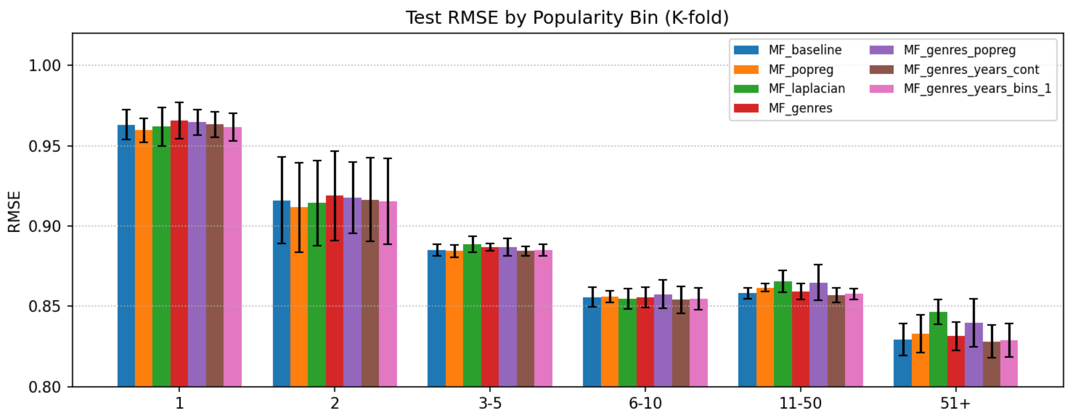

Figure 2: Test RMSE by item-popularity bin (3-fold).

3.4 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 ~ 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.

4 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.