Recommendation systems via approximate matrix factorization

Semenova Diana, Luzinsan Anastasia, Soluyanova Zlata

November 2024

Introduction

Recommendation systems play a crucial role in predicting user preferences for products by utilizing collaborative filtering techniques. This approach analyzes collective user behavior to enhance user experience, particularly in e-commerce and streaming platforms like Netflix. The economic implications of effective recommendations are significant, as evidenced by initiatives like the Netflix Prize competition in 2007, which offered $1 million for a 10% improvement in prediction accuracy.

The success of these systems hinges on the quality of their algorithms, with matrix factorization being a prominent method due to its ability to decompose user-item interaction matrices into lower-dimensional representations that capture latent user preferences and item characteristics.

Problem statement

We seek to decompose a matrix X into two low-rank matrices $U \in \mathbb{R}^{m \times r}$ and $V \in \mathbb{R}^{n \times r}$, such that:

$$X \approx UV^T$$

The optimization problem can be written as:

$$\min_{U,V} \|W \odot (X - UV^T)\|_F^2 + \lambda_{\text{reg}} \left(\|U\|_F^2 + \|V\|_F^2\right)$$

where:

- W is a binary mask matrix indicating observed entries ($W_{ij} = 1$ if X_{ij} is observed, 0 otherwise).
- $\odot$ represents element-wise multiplication (Hadamard product).
- λ_{reg} is a regularization parameter to prevent overfitting.
- $\| \cdot \|_F$ denotes the Frobenious norm.

Nonnegative matrix factorization

The **Multiplicative Update (MU)** algorithm iteratively updates W and H by minimizing the reconstruction error: $\min_{W,H} ||X - WH||_F^2$ subject to $W \geq 0$ and $H \geq 0$. Here, $|| \cdot ||_F$ denotes the Frobenius norm.
Update Equations
The MU algorithm applies the following element-wise update rules for W and H:
1. Update for H: $[H \leftarrow H \circ \frac{W^T X}{W^T W H}]$
2. Update for W: $[W \leftarrow W \circ \frac{X H^T}{W H H^T}]$
where $(\circ)$ denotes element-wise multiplication, and division is also element-wise.

Nonnegative matrix factorization result

Parameters:

- Initialization of U and V - random
- Number of iterations - 100
- Tolerance = 10^{-10}

RMSE obtained:

- validation loss - 1.0897
- test loss - 1.1098

Block coordinate descent

The method alternates between solving for U and V while fixing the other. These updates involve solving quadratic subproblems derived from the above objective.
a) fix U and update V: For each column j of V, the update is:

$$V[j,:] = \arg\min_{v_j} \|W[:,j] \odot (X[:,j] - Uv_j^T)\|_2^2 + \lambda_{\text{reg}} \|v_j\|_2^2$$

Extract only the known entries in column j:
Known indices: $\text{known\_idx} = \{i : W[i,j] = 1\}$. $U_{\text{known}} \in \mathbb{R}^{|\text{known\_idx}| \times r}$ and $X_{\text{known}} \in \mathbb{R}^{|\text{known\_idx}|}$.
Solve the normal equation:

$$A = U_{\text{known}}^T U_{\text{known}} + \lambda_{\text{reg}} I_r, \quad b = U_{\text{known}}^T X_{\text{known}}$$

$$V[j,:] = A^{-1} b$$

b) the same, but now fix V, update U

Block coordinate descent result

Parameters:

- Initialization of U and V - SVD
- Number of iterations - 5
- Tolerance = 10^{-4}
- Regularization parameter - 0.99

RMSE obtained:

- validation loss - 0.87613
- test loss - 0.9386

Update: gradient descent

Let the objective function to be minimized be the same. Now:

1. **Update** U: The gradient of the loss function with respect to U is given by:

$$\nabla_U L(U, V) = -2(X - UV^T)V + 2\lambda U$$

The update for U is performed by applying gradient descent:

$$U \leftarrow U - \alpha_U \nabla_U L(U, V)$$

where α_U is the learning rate (step size).

Update V: The same as for U

Update: gradient descent result

Parameters:

- Initialization of U and V - random
- Number of iterations - 100
- Tolerance = 10^{-10}

RMSE obtained:

- validation loss - 0.8761
- test loss - 0.8564

Hyperparameters tuning

Using Optuna to understand hyperparameter importances

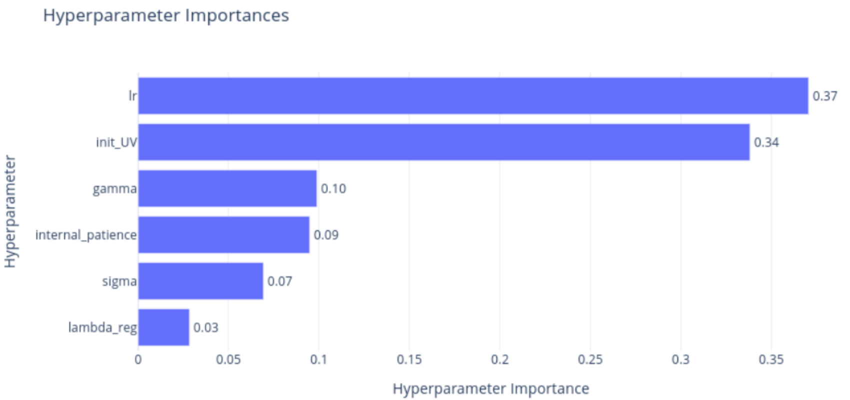

Figure: Hyperparameters importances.

Using Optuna to understand hyperparameter importances

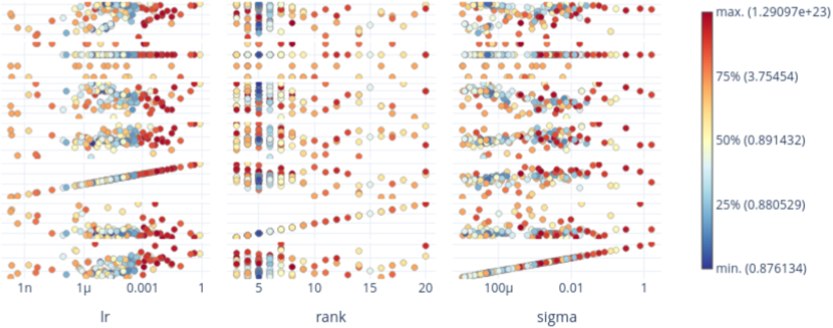

Figure: Hyperparameters importances.

Nonnegative matrix factorization

Hyperparameter	Value
Best Rank (r)	5
Initialization Method	SVD
Regularization Parameter (λ_{reg})	0.99

Table: The final results from NMF

Update: gradient descent

Hyperparameter	Best Value
r	5
`init_UV`	SVD
λ_{reg}	0.0562
c	12
`Internal Patience`	9
γ	0.0240
β	0.0416
`Initial Validation RMSE`	3.7532

Table: The final results from BCGD

Model comparison

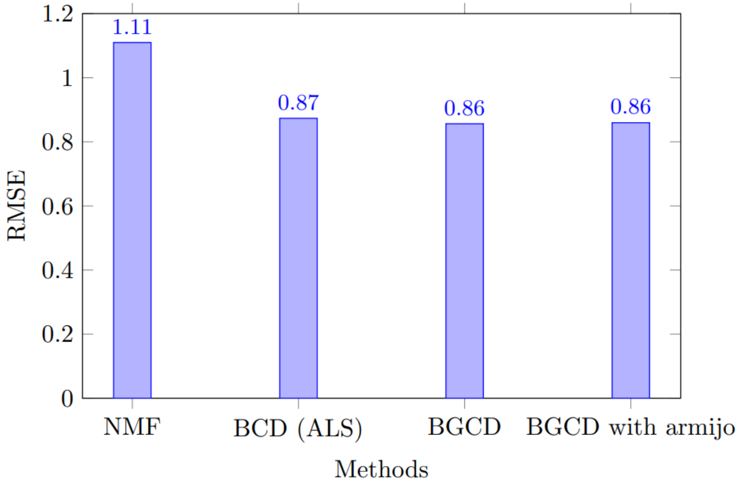

Figure: Error value for different models

Summary

We employed three algorithms for matrix factorization:

- Non-negative Matrix Factorization (NMF)
- Block Coordinate Descent (BCD)
- BCD with Gradient Descent (BGCD)

To enhance performance, a range of parameter settings were investigated:

- Initializations: random initialization, Singular Value Decomposition (SVD), with SVD demonstrating better results.
- Number of Latent Factors
- Step Size Adaptation: two step size strategies were tested: the Armijo rule and adaptive step size calculation

Thank you for your attention!