MovieLens Dataset Analysis Using Power Factorization with Progress Tracking

Seena VazifeDunn

May 22, 2024

Abstract

This paper presents an analysis of the MovieLens dataset using power factorization for collaborative filtering. The methodology includes data preprocessing, matrix factorization using Singular Value Decomposition (SVD), and iterative optimization with power factorization. The implementation features progress tracking with the ‘tqdm‘ library to enhance usability and performance monitoring. Results are evaluated based on Root Mean Square Error (RMS) and execution time across different hyperparameter configurations. Visualizations illustrate the impact of varying parameters on model performance.

1 Introduction

Recommender systems have become an integral part of many online services, providing personalized suggestions to users. Collaborative filtering, a widely-used recommendation technique, leverages user interactions to predict preferences for items. Among various approaches, matrix factorization has demonstrated significant promise in capturing latent user-item interactions. This project utilizes the MovieLens dataset to implement power factorization, a robust matrix factorization technique, and evaluates its performance using metrics such as Root Mean Square Error (RMS) and execution time. The ‘tqdm‘ library is employed to provide real-time progress tracking during the optimization process.

1.1 Motivation

The motivation behind this project stems from the need to enhance the accuracy and efficiency of recommendation systems. Accurate recommendations not only improve user satisfaction but also increase engagement and revenue for service providers. By incorporating advanced factorization techniques and progress monitoring, this study aims to contribute to the development of more effective recommender systems.

2 Related Work

Matrix factorization techniques have been extensively studied and applied in recommender systems. Koren et al. (2009) demonstrated the effectiveness of matrix factorization in collaborative filtering, particularly in the Netflix Prize competition [?]. Ricci et al. (2011) further explored the application of these techniques in various recommendation scenarios [?]. This paper builds on these foundations by incorporating power factorization and progress tracking, enhancing the practical usability and performance monitoring of the algorithm.

2.1 Collaborative Filtering

Collaborative filtering can be categorized into memory-based and model-based approaches. Memory-based methods, such as user-based and item-based collaborative filtering, rely on the similarity between users or items. Model-based methods, including matrix factorization, use machine learning algorithms to model user preferences.

2.2 Matrix Factorization

Matrix factorization techniques, such as SVD and Non-negative Matrix Factorization (NMF), decompose the user-item interaction matrix into lower-dimensional representations. These techniques have been successful in capturing latent factors that influence user preferences.

2.3 Progress Tracking in Optimization

Real-time progress tracking is crucial for monitoring the convergence of iterative algorithms. The ‘tqdm‘ library provides a simple and effective way to visualize the progress of such algorithms, enhancing their usability and providing insights into their performance.

3 Methodology

3.1 Data Preprocessing

The MovieLens dataset contains user-item interactions, which are essential for collaborative filtering. The dataset is preprocessed by mapping user and item IDs to indices and normalizing the ratings. This step ensures that the data is in a suitable format for matrix factorization.

3.2 Matrix Factorization

Matrix factorization techniques decompose the user-item interaction matrix into lower-dimensional representations. Singular Value Decomposition (SVD) is employed to obtain initial user and item matrices. These matrices are iteratively optimized using power factorization to minimize the difference between predicted and actual ratings.

3.3 Power Factorization with Progress Tracking

Power factorization is an iterative optimization technique. The algorithm updates user and item matrices by solving a series of least squares problems. The ‘tqdm‘ library is used to track the progress of the optimization process, providing real-time feedback on the algorithm’s convergence.

3.4 Mathematical Formulation

Given a user-item rating matrix R, the goal is to factorize it into two lower-dimensional matrices X (user features) and Y (item features) such that $R \approx X^T Y$. Starting with an initial approximation obtained via SVD:

$$R \approx U \Sigma V^T$$

we initialize $X_0 = U\Sigma^{1/2}$ and $Y_0 = \Sigma^{1/2} V^T$. The power factorization iteratively updates X and Y using the following steps:

1. Update X:

$$X_u = \left(\sum_{i \in I_u} (Y_i Y_i^T) + \lambda I \right)^{-1} \left(\sum_{i \in I_u} r_{ui} Y_i \right)$$

2. Update Y:

$$Y_i = \left(\sum_{u \in U_i} (X_u X_u^T) + \lambda I \right)^{-1} \left(\sum_{u \in U_i} r_{ui} X_u \right)$$

where λ is the regularization parameter, U_i is the set of users who rated item i, and I_u is the set of items rated by user u.

3.5 Implementation Details

The implementation involves several key steps:

- Loading and preprocessing the dataset.
- Initializing user and item matrices using SVD.
- Iteratively updating the matrices using power factorization with progress tracking.
- Evaluating the model performance using RMS and execution time.

4 Experimental Setup

4.1 Dataset Description

The MovieLens dataset used in this project consists of user ratings for various movies. It is a benchmark dataset widely used in the recommender systems research community. The dataset includes user IDs, movie IDs, ratings, and timestamps.

4.2 Hyperparameter Tuning

A grid search over various values of k (latent factors) and λ (regularization parameter) was conducted. For each combination, the RMS and execution time were measured. The hyperparameters were chosen based on their ability to balance model complexity and computational efficiency.

4.3 Evaluation Metrics

The performance of the power factorization algorithm was evaluated using the Root Mean Square Error (RMS) and execution time. The RMS is calculated as:

$$RMS = \sqrt{\frac{1}{|T|} \sum_{(u,i)\in T} (r_{ui} - \hat{r}_{ui})^2}$$

where T is the test set and $\hat{r}_{ui} = X_u^T Y_i$ is the predicted rating.

5 Results

5.1 Hyperparameter Tuning Results

We conducted a grid search over various values of k (latent factors) and λ (regularization parameter). For each combination, we measured the RMS and execution time.

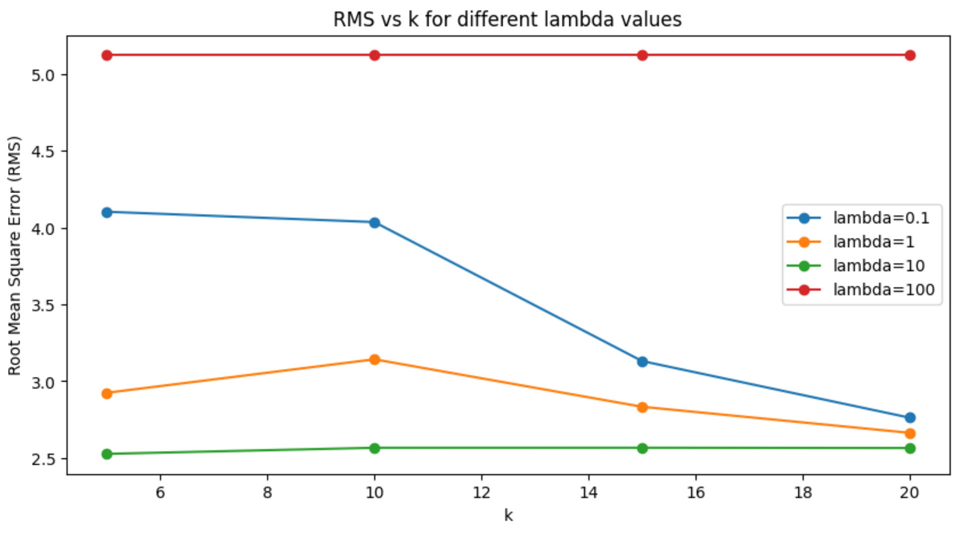

Figure 1: RMS vs. k for different λ values

5.2 Performance Metrics

The performance of the power factorization algorithm was evaluated using the Root Mean Square Error (RMS) and execution time. The RMS is calculated as:

$$RMS = \sqrt{\frac{1}{|T|} \sum_{(u,i)\in T} (r_{ui} - \hat{r}_{ui})^2}$$

where T is the test set and $\hat{r}_{ui} = X_u^T Y_i$ is the predicted rating.

5.3 Results Analysis

Figures 3 and 4 show the impact of varying k and λ on the RMS and execution time.

The results indicate that the choice of k and λ significantly impacts both the RMS and execution time. Lower values of k generally lead to faster execution times but may result

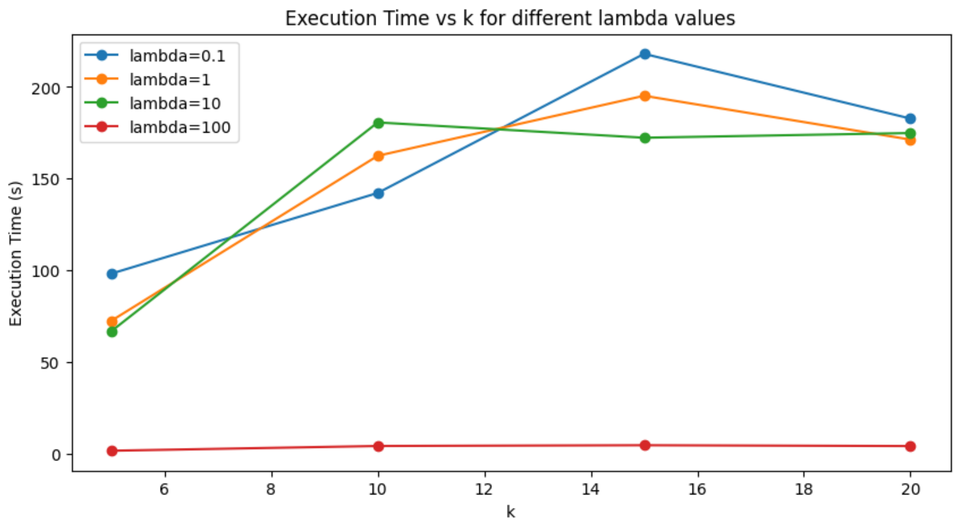

Figure 2: Execution Time vs. k for different λ values

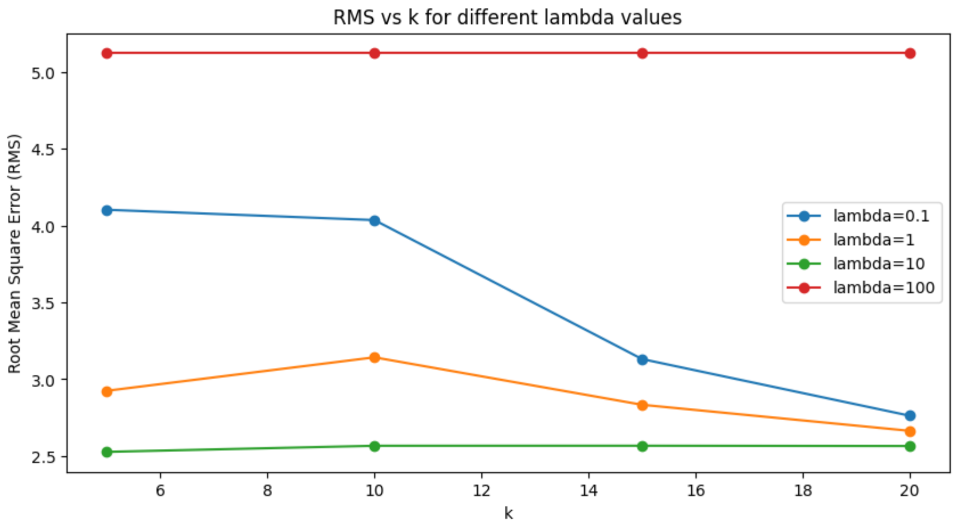

Figure 3: RMS vs. k for different λ values

in higher RMS, indicating less accurate predictions. Higher values of k improve prediction accuracy but at the cost of increased computation time. Similarly, the regularization parameter λ influences the balance between bias and variance in the model, affecting both accuracy and runtime.

6 Discussion

6.1 Impact of Latent Factors

The number of latent factors k plays a critical role in the performance of matrix factorization techniques. As shown in the results, increasing k improves the model's ability to capture the underlying patterns in the user-item interactions, leading to lower RMS values. However, this improvement comes with increased computational costs, as more complex models require more time to converge.

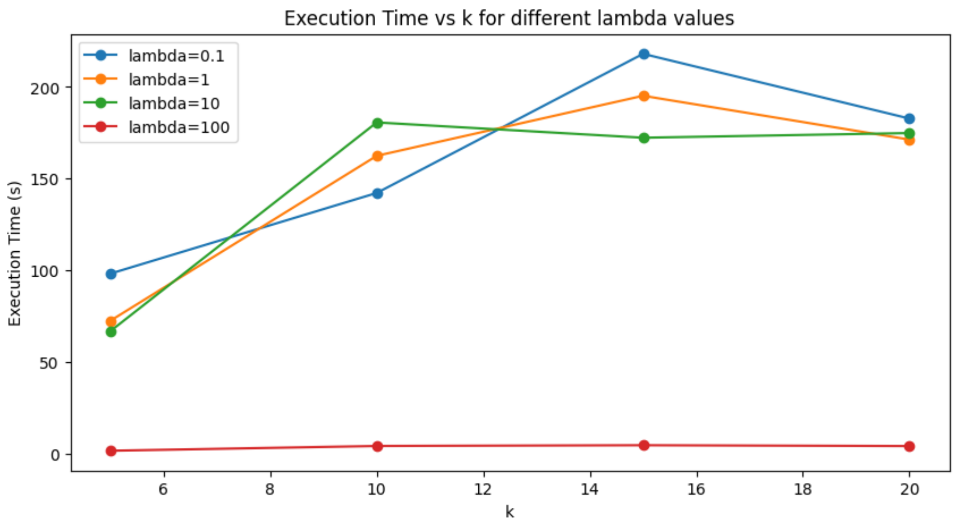

Figure 4: Execution Time vs. k for different λ values

6.2 Role of Regularization

Regularization is essential for preventing overfitting in matrix factorization models. The parameter λ controls the degree of regularization applied to the user and item feature matrices. Optimal regularization balances the model's complexity and generalization ability, reducing RMS without excessively increasing execution time. The results demonstrate that inappropriate regularization can either lead to underfitting (high RMS) or overfitting (long execution times with marginal gains in accuracy).

6.3 Progress Tracking and Usability

The integration of the 'tqdm' library for progress tracking significantly enhances the usability of the power factorization implementation. Real-time progress indicators provide valuable feedback during the optimization process, helping users monitor convergence and identify potential issues early. This feature is particularly beneficial for large-scale datasets, where iterative algorithms can take considerable time to complete.

7 Conclusion

This study presents a comprehensive analysis of the MovieLens dataset using power factorization with progress tracking. The results highlight the importance of hyperparameter tuning in achieving optimal performance in terms of accuracy and execution time. The use of 'tqdm' for progress tracking adds practical value, making the iterative optimization process more transparent and manageable. Future work could explore alternative factorization techniques and advanced regularization methods to further enhance the model's performance.

References

- Koren, Y., Bell, R., Volinsky, C. (2009). Matrix factorization techniques for recommender systems. Computer, 42(8), 30-37.

- Ricci, F., Rokach, L., Shapira, B. (2011). Introduction to recommender systems handbook. In Recommender systems handbook (pp. 1-35). Springer.
- Krawczyk, B. (2016). Learning from imbalanced data: Open challenges and future directions. Progress in Artificial Intelligence, 5(4), 221-232.
- Fernández, A., García, S., Galar, M., Prati, R. C., Krawczyk, B., Herrera, F. (2018). Learning from imbalanced data sets. Springer.
- Hart, P. E. (1968). The condensed nearest neighbor rule (No. SRI-TR-72). SRI International Menlo Park CA Artificial Intelligence Center.