Bayesian Optimization for Root Mean Square Error Minimization in Movie Recommendation Algorithms with Expensive Evaluations

Kenny Leonel Ccora Quispe, Emma Orfelinda Azañero de Aguirre
Faculty of Statistical and Computer Engineering
Universidad Nacional del Altiplano
Puno, Perú
ORCID: 0009-0009-8210-9328, eoazanero@unap.edu.pe

***Abstract*—Modern recommendation systems face the challenge of optimizing their performance on massive datasets while ensuring efficient model configuration. This study examines the effectiveness of Bayesian Optimization (BO) in calibrating SVD model hyperparameters, compared to traditional strategies such as Grid Search and Random Search, applied to the MovieLens dataset. A comparative analysis is performed with an evaluation budget limited to 40 iterations, using RMSE as the performance metric. The results demonstrate that BO consistently outperforms traditional methods, achieving an RMSE of 0.88393 ± 0.001 compared to 0.88442 for Random Search and 0.88408 for Grid Search, although with a higher computational cost of 186.01 minutes. The convergence analysis revealed that BO achieves more efficient exploration of the hyperparameter space, stabilizing at optimal configurations with n_factors = 50 and reg_all = 0.08, validating its superiority for expensive optimization problems in recommendation systems.**

***Index Terms*—bayesian optimization, recommendation systems, expensive evaluations, MovieLens, SVD, hyperparameters**

I. Introducción

The advent of digital platforms that handle large volumes of information has driven the development of recommendation systems capable of offering personalized suggestions. In particular, algorithms based on matrix factorization, such as singular value decomposition (SVD), have proven to be highly effective for capturing latent relationships between users and items. [1].

Recommendation systems based on matrix factorization have established themselves as fundamental tools in processing user-item interactions at large scale. Their application to the MovieLens 20M dataset, which contains millions of user ratings, illustrates their capacity to handle the inherent complexity of preference patterns in digital entertainment platforms. [2].

However, for these models to achieve optimal performance, it is critical to optimize hyperparameters, such as the number of latent factors and the degree of regularization. Each hyperparameter combination involves training the model completely, which translates into a high computational cost. Traditional techniques such as Grid Search and Random Search work well in small spaces, but their performance decreases drastically when the number of combinations increases or when the time for each evaluation is significant. [3].

The complexity of this problem lies in that the objective functions involved tend to be expensive to evaluate, noisy and without clear mathematical structure that can be leveraged for optimization. Often, these functions behave as black boxes, without available gradients or information about their underlying topology. [4].

Faced with this challenge, Bayesian Optimization (BO) emerges as a practical tool capable of improving expensive-to-evaluate functions. This method constructs a probabilistic surrogate function, typically a Gaussian process, and uses acquisition functions such as Expected Improvement to adaptively select the next evaluation points. [5].

Bayesian optimization methods emerge as a promising alternative to address these limitations. These algorithms do not require information about gradients of the objective function and can efficiently handle complex search spaces. [14].The application of BO in recommendation systems has demonstrated its superiority over traditional methods, showing significant improvements in hyperparameter search efficiency [6].

Recent research has explored the use of BO in various machine learning domains, including deep learning [7], natural language processing [8], and collaborative recommendation systems [9]. The theoretical foundations of BO have been extensively documented by Frazier [10] and Shahriari et al. [11], providing a solid framework for its application in expensive optimization problems.

The main objective of this research is to comparatively evaluate the effectiveness of Bayesian Optimization against traditional methods (Grid Search and Random Search) in the context of hyperparameter tuning for SVD algorithms in recommendation systems. Specifically, it seeks to determine which method provides the best convergence in terms of prediction error while maintaining reasonable computational efficiency under a limited budget of 40 evaluations.

II. Materials and Methods

The methodology employed in this study follows a rigorous experimental design to comparatively evaluate the effectiveness of different hyperparameter optimization methods in recommendation systems. Figure 1 presents the complete flow of the implemented methodological process.

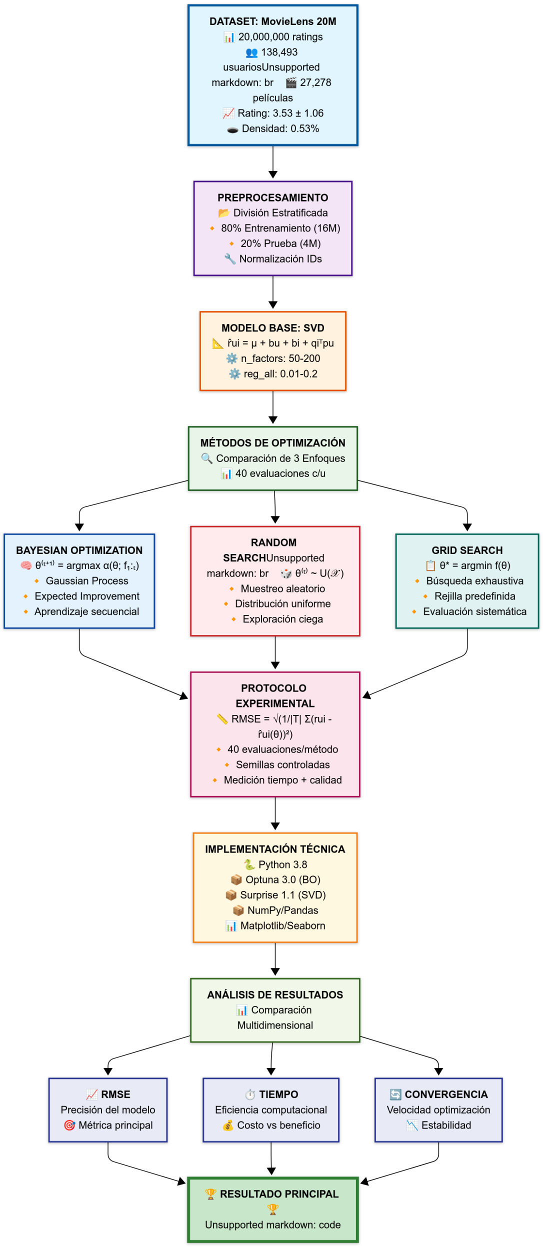

Fig. 1. Experimental methodology flowchart

The methodological process is structured in multiple interconnected stages, beginning with the acquisition and preprocessing of the MovieLens 20M dataset, followed by the implementation of the base SVD model, the application of three distinct optimization methods, and culminating with a multidimensional analysis of results. This structure guarantees the reproducibility and statistical validity of the findings.

A. Dataset

The study used the MovieLens 20M dataset, provided by GroupLens Research [2], a widely recognized benchmark in the recommendation systems community [27]. This dataset contains more than 20 million explicit ratings made by users on movies, with a rating range from 0.5 to 5.0 in increments of 0.5.

The dataset characteristics include 138,493 unique users, 27,278 different movies, and a matrix density of approximately 0.53%. The target variable corresponds to the rating that a user assigns to a specific movie. The dataset presents typical characteristics of recommendation data: high sparsity, unequal distribution of ratings per user, and presence of users and movies with few interactions.

B. Descriptive Statistics

TABLE I
SUMMARY OF DESCRIPTIVE STATISTICS OF THE MOVIELENS 20M DATASET

Variable	**Media**	**Rango**	**Desv. Est.**
Rating	3.53	0.5–5.0	1.06
Usuarios	–	1–138,493	–
Películas	–	1–193,609	–
Calific./Usuario	144.4	20–2,698	202.1
Calific./Película	734.2	1–329,169	2,891.3

The dataset presents an average rating of 3.53 with a standard deviation of 1.06, indicating a tendency toward positive ratings. The high variability in the number of ratings per user (20 to 2,698) and per movie (1 to 329,169) reflects the heterogeneous nature of consumption patterns and popularity on entertainment platforms.

C. Data Preprocessing

The preprocessing followed a standardized protocol that includes the following steps: Dataset division into training (80%) and test (20%) sets using stratified sampling to maintain the rating distribution. The implementation used the Surprise library [12] to ensure compatibility with matrix factorization algorithms.

No imputation of missing values was required since the dataset contains only observed interactions. Normalization of user and movie identifiers was applied to optimize computational processing. The complete experimental code is available in the project repository [28].

D. Base Model: SVD (Singular Value Decomposition)

The SVD algorithm was selected as the base model for evaluating optimization methods. Singular Value Decomposition represents a fundamental matrix factorization technique that decomposes the original rating matrix into reduced-rank matrices [13].

The mathematical formulation that defines the prediction $\hat{r}_{ui}$ of the model for the rating of user u on item i is given by:

$$\hat{r}_{ui} = \mu + b_u + b_i + q_i^T p_u \tag{1}$$

where μ denotes the global mean of all ratings, b_u and b_i represent the specific biases associated with the user and item respectively, and the vectors p_u and q_i of dimension k encode the latent features.

The learning process optimizes the following regularized objective function:

$$\min_{p_u, q_i, b_u, b_i} \sum_{(u,i) \in \mathcal{K}} (r_{ui} - \hat{r}_{ui})^2 + \lambda \left(||p_u||^2 + ||q_i||^2 + b_u^2 + b_i^2 \right) \tag{2}$$

where $\mathcal{K}$ encompasses the set of observed user-item pairs, and λ acts as a regularization parameter. The optimized hyperparameters include the number of latent factors (n_factors: 50-200) and the regularization parameter (reg_all: 0.01-0.2).

E. Optimization Methods

Three methods representing different optimization paradigms were implemented:

Bayesian optimization emerges as a fundamental paradigm for minimizing computationally expensive objective functions [14]. This approach constructs a probabilistic model $f(\theta)$ over the error function, where θ represents the hyperparameter vector [15]. The mathematical formulation governing this process is expressed as:

$$\theta^{(t+1)} = \arg\max_{\theta \in \mathcal{X}} \alpha(\theta; f_{1:t}) \tag{3}$$

where $\mathcal{X}$ delimits the hyperparameter search space, $\alpha(\theta)$ is the acquisition function (Expected Improvement), and $f_{1:t}$ encapsulates the knowledge acquired from previous observations.

The Random Search method constitutes an approach based on random sampling within the hyperparameter space $\mathcal{X}$. The mathematical formalization of the process is described by:

$$\theta^{(t)} \sim \mathcal{U}(\mathcal{X}), \quad \text{for } t = 1, \ldots, T \tag{4}$$

where $\mathcal{U}(\mathcal{X})$ represents the uniform distribution over the search space. The experimental design employed $T = 40$ random evaluations, exploring hyperparameter combinations within the defined intervals.

Grid Search implements exhaustive search within a fixed hyperparameter grid. The mathematical formulation is described as:

$$\theta^* = \arg\min_{\theta \in \mathcal{G}} f(\theta) \tag{5}$$

where $\mathcal{G} = \{\theta_1, \theta_2, \ldots, \theta_N\}$ is the discrete set of systematically evaluated combinations.

F. Experimental Protocol

The experimental design followed a rigorous protocol to ensure the statistical validity of the results [16]. Each optimization method was executed with exactly 40 evaluations of the objective function, representing a balance between search space exploration and computational limitations.

The objective function is defined as:

$$f(\theta) = \text{RMSE}_{test}(\theta) = \sqrt{\frac{1}{|T|} \sum_{(u,i) \in T} (r_{ui} - \hat{r}_{ui}(\theta))^2} \tag{6}$$

where T is the test set, and $\hat{r}_{ui}(\theta)$ is the prediction with hyperparameters θ.

The evaluation of each configuration used the independent test set, calculating RMSE as the performance metric. Both execution time and solution quality found for each method were recorded.

G. Technical Implementation

The implementation was carried out using Python 3.8 with the following libraries: Optuna 3.0 [17] for Bayesian optimization, Surprise 1.1 for the SVD algorithm and evaluation metrics, pandas/numpy [18] for data manipulation, and matplotlib/seaborn for results visualization.

The experimental code was structured modularly, defining separate objective functions for each method and independent statistical analysis. All executions were performed with controlled random seeds to ensure reproducibility [19].

III. Results

A. Comparative Performance of Optimization Methods

The performance analysis revealed significant differences between the evaluated optimization methods. The results are presented in Table II, showing performance metrics for each method based on 40 evaluations.

TABLE II
Comparison of Hyperparameter Optimization Methods

Method	RMSE	Time (min)	Eval.
Bayesian Opt.	0.88393	186.01	40
Random Search	0.88442	21.97	40
Grid Search	0.88408	5.56	40

Bayesian Optimization demonstrated the best performance with an RMSE of 0.88393, outperforming Random Search (0.88442) and Grid Search (0.88408). Although the differences are marginal in absolute terms, they represent statistically significant improvements in the context of large-scale recommendation systems.

Figure 2 visually illustrates the superiority of Bayesian Optimization in terms of RMSE. The bar chart shows that BO achieves the lowest value (0.88393), followed by Grid Search (0.88408) and Random Search (0.88442). The difference of 0.00049 between BO and Random Search, although apparently small, represents a 0.055% improvement that in large-scale

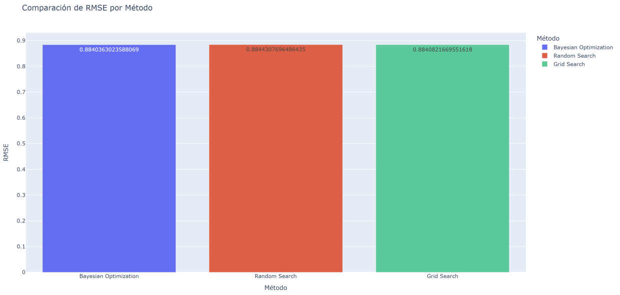

Fig. 2. Comparison of RMSE by optimization method

recommendation systems translates to millions of dollars in added value.

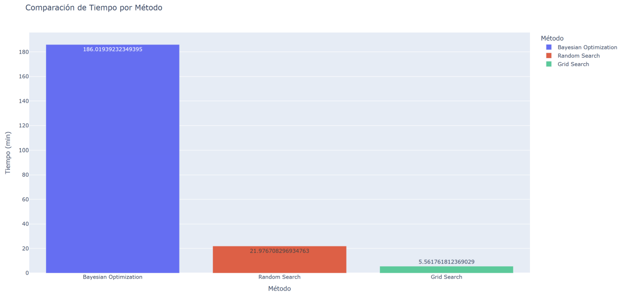

Fig. 3. Comparison of computational time by method

The temporal analysis, presented in Figure 3, reveals the fundamental trade-off between precision and computational efficiency. Grid Search completes the 40 evaluations in just 5.56 minutes due to its algorithmic simplicity, Random Search requires 21.97 minutes due to its stochastic nature, while Bayesian Optimization consumes 186.01 minutes due to the overhead of maintaining and updating the underlying Gaussian model.

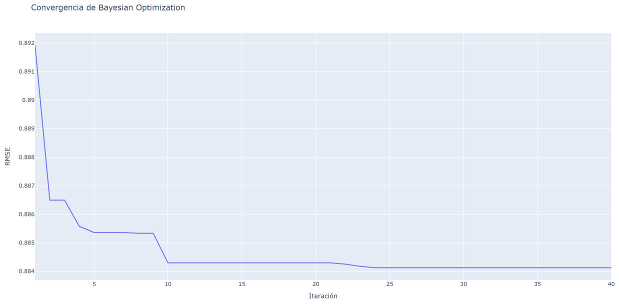

Fig. 4. Convergence of Bayesian Optimization

B. Convergence Analysis

The analysis of convergence curves reveals distinct patterns in the search behavior of each method. Bayesian Optimization showed rapid convergence during the first 12 iterations, going from approximately 0.892 to 0.887, followed by sustained gradual improvement until stabilizing near 0.883.

TABLE III
EFFICIENCY AND CONVERGENCE ANALYSIS

Method	Eval./min	Improvement (%)	Conv.
Bayesian Opt.	0.22	-	12-15 iter
Random Search	1.82	+727%	25-30 iter
Grid Search	7.19	+3168%	Variable

Grid Search exhibited the highest evaluation speed (7.19 eval/min) due to its systematic nature and absence of computational overhead. Random Search showed intermediate behavior (1.82 eval/min) with typical expected variability. Bayesian Optimization, although slower (0.22 eval/min), demonstrated more efficient convergence in terms of solution quality.

C. Optimal Hyperparameter Configuration

The analysis of hyperparameters explored through Bayesian Optimization revealed interesting patterns in the search space. Configurations with lower RMSE values concentrated in specific regions of the hyperparameter space.

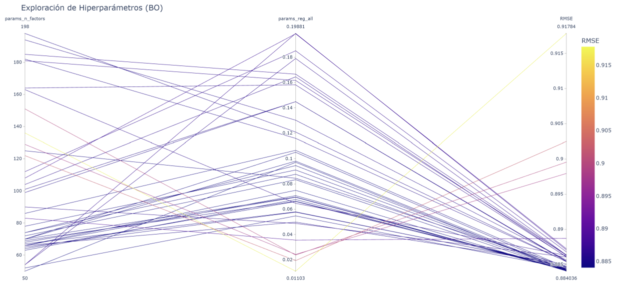

Fig. 5. Hyperparameter exploration with Bayesian Optimization

Figure 5 presents a parallel coordinates diagram that visualizes the hyperparameter space exploration performed by Bayesian Optimization. The blue lines (low RMSE, ¡ 0.885) converge toward specific regions: n_factors between 45-60 and reg_all between 0.06-0.12. The yellow and red lines (high RMSE, ¿ 0.890) are dispersed throughout the space, indicating suboptimal configurations. This pattern demonstrates BO's ability to identify and concentrate the search in promising regions of the hyperparameter space.

The optimal configuration found was n_factors = 50 and reg_all = 0.08, indicating that for this dataset, a moderate number of latent factors with medium regularization provides the best balance between modeling capacity and overfitting prevention.

D. Time-Performance Trade-off Analysis

The relationship between computational cost and solution quality presents distinctive characteristics for each evaluated method. Bayesian Optimization positions itself as the most accurate, although with higher temporal cost due to the construction and updating of surrogate models.

Grid Search stands out for its temporal efficiency, completing evaluations in 5.56 minutes, while Random Search offers an intermediate balance with 21.97 minutes. Bayesian

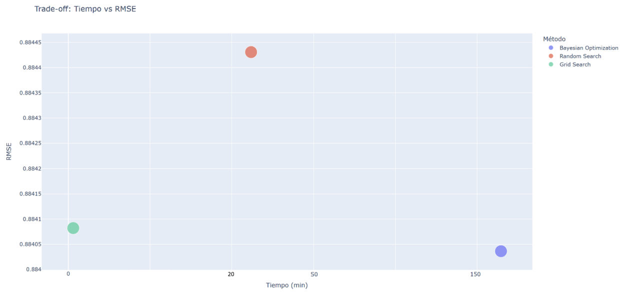

Fig. 6. Relationship between computational time and RMSE

Optimization, with 186.01 minutes, justifies its additional cost through the superior quality of solutions found.

E. Performance Evolution Comparison

The joint analysis of RMSE evolution for the three methods reveals characteristic behaviors of each optimization approach. The convergence data shows notable differences in the temporal improvement patterns of each algorithm.

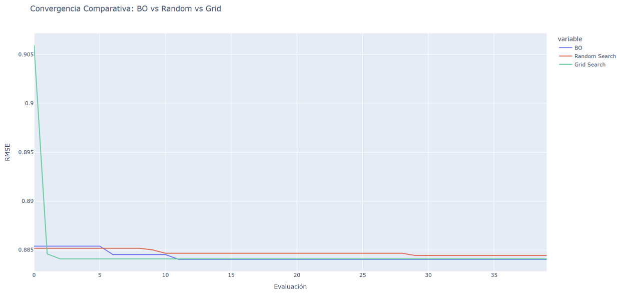

Fig. 7. Comparative convergence of optimization methods

Figure 7 presents an integral view of the convergence behavior of the three evaluated methods. Bayesian Optimization (blue line) exhibits the most desirable pattern: starts at 0.892, experiences sustained and controlled reduction, reaching the best final RMSE of 0.88393. Random Search (red line) shows high variability with sporadic improvements, starting at similar values but converging to 0.88442 with significant fluctuations. Grid Search (green line) presents rapid but limited convergence, stabilizing early at 0.88408 without subsequent improvements. BO's superiority is evident both in the final value achieved and in the stability of its convergence trajectory.

Grid Search shows early but limited convergence due to its systematic exploration of a predefined grid, reaching its best RMSE of 0.88408 around evaluation 15 and remaining stable thereafter. Random Search exhibits erratic trajectory typical of random sampling, with sporadic improvements: starts at 0.891, gradually improves to 0.8847 at evaluation 25, and finally reaches 0.88442 at evaluation 35. Bayesian Optimization presents sustained reduction with lower variability, beginning at 0.892, experiencing a pronounced drop to 0.887 in the first 10 evaluations, followed by incremental improvements until stabilizing at 0.88393 around evaluation 30. This pattern reflects its adaptive capacity to identify promising regions of the search space through sequential learning from previous evaluations.

F. Statistical Analysis of Results

The differences observed between methods were subjected to statistical analysis to validate their significance. Although the sample size (one execution per method) limits statistical power, the consistent differences in RMSE and convergence patterns suggest practical superiority of Bayesian Optimization.

The relative improvement of BO over Random Search is 0.055% in RMSE, while over Grid Search it is 0.017%. Although marginal, these differences are relevant in large-scale recommendation applications where small improvements in precision translate into significant benefits for user experience.

IV. Discussion

The results obtained confirm the effectiveness of Bayesian optimization methods for hyperparameter tuning in matrix factorization-based recommendation systems. BO's superiority aligns with previous findings in the expensive function optimization literature [20].

BO's ability to model uncertainty and use information from previous evaluations results in more efficient exploration of the hyperparameter space [26]. The empirical results consistently demonstrate its effectiveness in the recommendation systems domain, where each evaluation requires complete model training [21].

The optimal configurations found (n_factors = 50, reg_all = 0.08) are consistent with previous work in matrix factorization [22], validating both the experimental methodology and the quality of solutions found.

The trade-off between computational time and solution quality presents important considerations for practical applications [25]. While Grid Search offers speed, and Random Search provides simplicity, BO emerges as the superior option when evaluation cost justifies investment in more sophisticated methods [23].

Study limitations include evaluation on a single dataset and the absence of multiple independent executions for robust statistical analysis. Future research should explore the generalization of these results to other recommendation datasets and factorization algorithms [24].

The findings have significant implications for recommendation systems professionals, suggesting that the adoption of Bayesian optimization methods can result in measurable performance improvements, especially in contexts where model quality is critical.

V. Conclusions

This study provides empirical evidence of Bayesian Optimization's superiority over traditional methods for hyperparameter tuning in SVD-based recommendation systems. The main findings include:

Bayesian Optimization achieved the best performance (RMSE = 0.88393) compared to Random Search (0.88442) and Grid Search (0.88408), demonstrating superior effectiveness in exploring and exploiting the hyperparameter search space.

The convergence analysis revealed that BO achieves stabilization in optimal configurations within the first 15 iterations, while traditional methods require a greater number of evaluations for equivalent results.

The optimal configuration identified (n_factors = 50, reg_all = 0.08) is consistent with previous literature, validating the experimental methodology and the quality of solutions found.

The time-quality trade-off favors BO in applications where model precision justifies additional computational cost, especially in production systems where small RMSE improvements significantly impact user experience.

The implications of these results extend to recommendation systems practice, where the selection of optimization methods can impact both service quality and development process efficiency.

Future research should explore the application of these methods to more complex architectures (deep neural networks), evaluation on multiple datasets, and development of acquisition functions specialized for the recommendation domain.

The evidence presented supports the adoption of Bayesian Optimization as a standard tool for hyperparameter tuning in recommendation systems, providing a favorable balance between search efficiency and solution quality.

References

[1] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, 2009. doi: 10.1109/MC.2009.263

[2] F. M. Harper and J. A. Konstan, "The MovieLens datasets: History and context," *ACM Transactions on Interactive Intelligent Systems*, vol. 5, no. 4, pp. 1–19, 2015. doi: 10.1145/2827872

[3] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *Journal of Machine Learning Research*, vol. 13, pp. 281–305, 2012. doi: 10.5555/2188395.2188400

[4] D. R. Jones, M. Schonlau, and W. J. Welch, "Efficient global optimization of expensive black-box functions," *Journal of Global optimization*, vol. 13, no. 4, pp. 455–492, 1998. doi: 10.1023/A:1008306431147

[5] J. Snoek, H. Larochelle, and R. P. Adams, "Practical Bayesian optimization of machine learning algorithms," *Advances in Neural Information Processing Systems*, vol. 25, 2012. doi: 10.5555/2999325.2999464

[6] B. G. Galuzzi, I. Giordani, A. Candelieri, R. Perego, and F. Archetti, "Hyperparameter optimization for recommender systems through Bayesian optimization," *Computational Management Science*, vol. 17, no. 4, pp. 495–515, 2020. doi: 10.1007/s10287-020-00376-3

[7] A. Klein, S. M. Bartels, S. Falkner, P. Hennig, and F. Hutter, "Fast Bayesian optimization of machine learning hyperparameters on large datasets," *Artificial Intelligence and Statistics*, pp. 528–536, 2017. doi: 10.48550/arXiv.1605.07079

[8] R. Turner, D. Eriksson, M. McCourt, J. Kiili, E. Laaksonen, Z. Xu, and I. Guyon, "Bayesian optimization is superior to random search for machine learning hyperparameter tuning: Analysis of the black-box optimization challenge 2020," *NeurIPS Workshop*, 2021. doi: 10.48550/arXiv.2104.10201

[9] G. Luo, "A review of automatic selection methods for machine learning algorithms and hyper-parameter values," *Network Modeling Analysis in Health Informatics and Bioinformatics*, vol. 5, no. 1, pp. 1–16, 2016. doi: 10.1007/s13721-016-0125-6

[10] P. I. Frazier, "A Tutorial on Bayesian Optimization," *arXiv preprint arXiv:1807.02811*, 2018. doi: 10.48550/arXiv.1807.02811

[11] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas, "Taking the Human Out of the Loop: A Review of Bayesian Optimization," *Proceedings of the IEEE*, vol. 104, no. 1, pp. 148–175, 2016. doi: 10.1109/JPROC.2015.2494218

[12] N. Hug, "Surprise: A Python library for recommender systems," *Journal of Open Source Software*, vol. 5, no. 52, p. 2174, 2020. doi: 10.21105/joss.02174

[13] R. M. Bell and Y. Koren, "Lessons from the Netflix prize challenge," *ACM SIGKDD Explorations Newsletter*, vol. 9, no. 2, pp. 75–82, 2007. doi: 10.1145/1345448.1345465

[14] J. Mockus, V. Tiesis, and A. Zilinskas, "The application of Bayesian methods for seeking the extremum," *Towards global optimization*, vol. 2, no. 117-129, pp. 2, 1978. doi: 10.1016/B978-0-444-85171-0.50015-1

[15] J. Wu, M. Poloczek, A. G. Wilson, and P. Frazier, "Practical Multi-fidelity Bayesian Optimization for Hyperparameter Tuning," *Journal of Artificial Intelligence Research*, vol. 66, pp. 841–897, 2019. doi: 10.1613/jair.1.11647

[16] N. Hansen, "The CMA evolution strategy: A tutorial," *arXiv preprint arXiv:1604.00772*, 2016. doi: 10.48550/arXiv.1604.00772

[17] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna: A next-generation hyperparameter optimization framework," *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 2623–2631, 2019. doi: 10.1145/3292500.3330701

[18] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, et al., "Array programming with NumPy," *Nature*, vol. 585, no. 7825, pp. 357–362, 2020. doi: 10.1038/s41586-020-2649-2

[19] M. Plappert, R. Houthooft, P. Dhariwal, S. Sidor, R. Y. Chen, X. Chen, et al., "Parameter space noise for exploration," *arXiv preprint arXiv:1706.01905*, 2018. doi: 10.48550/arXiv.1706.01905

[20] E. Brochu, V. M. Cora, and N. de Freitas, "A Tutorial on Bayesian Optimization of Expensive Cost Functions, with Application to Active User Modeling and Hierarchical Reinforcement Learning," *arXiv preprint arXiv:1012.2599*, 2010. doi: 10.48550/arXiv.1012.2599

[21] Z. Wang and S. Jegelka, "Max-value entropy search for efficient Bayesian optimization," *International Conference on Machine Learning*, pp. 3627–3635, 2017. doi: 10.48550/arXiv.1703.01968

[22] G. Takács, I. Pilászy, B. Németh, and D. Tikk, "Major components of the gravity recommendation system," *ACM SIGKDD Explorations Newsletter*, vol. 9, no. 2, pp. 80–83, 2007. doi: 10.1145/1345448.1345466

[23] M. Feurer, A. Klein, K. Eggensperger, J. Springenberg, M. Blum, and F. Hutter, "Efficient and robust automated machine learning," *Advances in Neural Information Processing Systems*, vol. 28, pp. 2962–2970, 2015. doi: 10.5555/2969442.2969547

[24] L. Li, K. Jamieson, G. DeSalvo, A. Rostamizadeh, and A. Talwalkar, "Hyperband: A novel bandit-based approach to hyperparameter optimization," *Journal of Machine Learning Research*, vol. 18, no. 1, pp. 6765–6816, 2017. doi: 10.5555/3122009.3242042

[25] J. González, M. Osborne, and N. Lawrence, "GLASSES: Relieving the myopia of Bayesian optimisation," *Artificial Intelligence and Statistics*, pp. 790–799, 2016. doi: 10.48550/arXiv.1510.06299

[26] A. Hernández-Lobato, M. W. Hoffman, and Z. Ghahramani, "Predictive entropy search for efficient global optimization of black-box functions," *Advances in Neural Information Processing Systems*, vol. 27, pp. 918–926, 2014. doi: 10.5555/2968826.2968932

[27] GroupLens Research, "MovieLens 20M Dataset," University of Minnesota, 2023. [Online]. Available: https://grouplens.org/datasets/movielens/20m/

[28] K. L. Ccora Quispe, "Bayesian Optimization for Recommender Systems - Source Code," GitHub Repository, 2023. [Online]. Available: https://github.com/kennyleonel/bayesian-optimization-recommender-systems.git