Personalized Recommendations in E-commerce: A Case Study on Sports and Outdoor Activities

Patricia Da-Concepcion-Sarrate, Carlos Fernandez-Lozano, Adrian Carballal, and Francisco Cedron

ISLA: Laboratorio de Ingeniería del Software, Faculty of Computer Science, Universidade da Coruña, 15071 A Coruña, Spain
MALL: Machine Learning in Life Sciences, Faculty of Computer Science, Universidade da Coruña, 15071 A Coruña, Spain
RNASA-IMEDIR: Redes de Neuronas Artificiales y Sistemas Adaptativos - Imagen Médica y Diagnóstico Radiológico, Faculty of Computer Science, Universidade da Coruña, 15071 A Coruña, Spain
Centro de Investigación CITIC, Universidade da Coruña, 15071 A Coruña, Spain
Correspondence: francisco.cedron@udc.es

DOI: *https://doi.org/10.17979/spudc.9788497498913.8*

Abstract: Nowadays, recommendation systems have become essential tools in e-commerce and social networks, offering personalized content, product, and service suggestions based on user behaviour and preferences. This document focuses on collaborative filtering, which makes recommendations using users' past product ratings. Several collaborative filtering algorithms will be compared, including Alternating Least Squares (ALS), Non-negative Matrix Factorization (NMF), Singular Value Decomposition (SVD), Bayesian Personalized Ranking (BPR), and Neural Collaborative Filtering (NCF). These methods will be tested on a dataset of sports and outdoor products from Amazon. The performance will be evaluated with two types of metrics: for rating predictions, RMSE, MAE, and R^2; and for ranking, Precision, Recall, and nDCG.

1 Introduction

There is currently a huge amount of content in the digital world, and as time goes by, this content grows. This is where the need for personalization arises, helping the user by filtering the information available, reducing the number of options (Reutskaja et al., 2018). This is what recommendation systems do, which, in addition to showing users relevant elements that they already know, offer them other products that are also of interest to them but that they did not know existed (Nabavi and Nihtianov, 2012). These systems allow the user to be shown useful information without the user having to ask for it.

These recommendation systems are applied to various fields such as the field of music. Companies such as Spotify use them to show elements that may be relevant to their users, such as discovering new songs or artists that they were previously unaware of (Allawadi and Vij, 2023). Netflix also makes use of these recommendation systems in the field of movies and series (Gupta and Dave, 2020). This is reflected in the top 10 weeks they carry out with the most popular elements among their users.

The use of recommendation systems can be extended to other fields, in this case to sports and outdoor activities. A study of different recommendation system algorithms will be carried out to determine which of them make the best recommendations for each user.

2 Materials and methods

2.1 Datasets

The dataset used in this work comes from reviews of products related to sports and outdoor activities obtained from the Amazon store (Jianmo Ni, 2018; Ni et al., 2019). The dataset contains almost 13 million reviews, of which only the user ID, the review and the product ID will be used.

To avoid problems such as cold-start, a data preprocessing was performed that consisted of using only users who have 10 or more product reviews. This processing reduced the number of reviews to 1.5 million reviews.

2.2 Methodology

To evaluate the different recommendation system algorithms selected for this work, a machine learning technique known as k-fold cross-validation was implemented (Ilias and andEl Mhamdi Jamal, 2021). Specifically, a k value of 10 was used, meaning that the data was split into 10 partitions.

Additionally, for all the algorithms employed in this study, a hyperparameter optimization process will be carried out using grid search (Belete and Huchaiah, 2022). This approach consists of an exhaustive search over a predefined set of parameters, evaluating all possible combinations to identify the values that provide the best model performance.

2.3 ALS: Alternative Least Square

It is a matrix factorization algorithm whose main objective is to decompose a matrix of interactions between users and items into two lower-dimensional matrices, one for users and one for items (Takács and Tikk, 2012). It uses the least squares technique to adjust the values of these matrices, minimizing the error in reconstructing the original matrix. In each iteration, one of the lower-dimensional matrices is optimized while the other is fixed, alternating between them. This algorithm is very effective for sparse data.

The hyperparameters used for searching the best possible combination are the following:

- **rank:** Represents the number of latent factors or features considered when decomposing the original matrix. The tested values for this parameter are 15, 30, 50, and 65.
- **regparam:** This is the regularization parameter corresponding to λ, which controls the level of penalty applied to large values of the latent factors. The tested values for this parameter are 0.01, 0.1 and 0.3.
- **maxiter:** This refers to the maximum number of iterations. The tested values for this parameter are 25, 50 and 75.

2.4 NMF: Non Negative matix Factorization

It is a matrix factorization algorithm whose main objective is to decompose a matrix of interactions between users and items into two lower-dimensional matrices, one for users and one for items (Xin Luo and Zhu, 2014). In this case, there is a restriction that there cannot be negative values in the decomposition of the original matrix. This is adjusted according to the characteristics of each problem, which in this case involves information about product ratings that are within a range of 1 to 5, where negative ratings are not considered.

The hyperparameters used for searching the best possible combination are the following:

- **n_factors:** Represents the number of latent factors or features considered when decomposing the original matrix. The tested values for this parameter are 15, 30 and 50.
- **reg_pi:** This is the regularization parameter for the items matrix. The tested values for this parameter are 0.001, 0.02 and 0.1.
- **reg_qu:** This is the regularization parameter for the users matrix. The tested values for this parameter are 0.001, 0.02 and 0.1.

- **n_epochs:** This is the maximum number of iterations. The tested values for this parameter are 50, 100 and 150.

2.5 SVD: Singular Value Decomposition

It is a matrix factorization algorithm whose main objective is to decompose a matrix of interactions between users and items into three lower-dimensional matrices: one for users, one for items, and a diagonal matrix of singular values (Chang et al., 2024). This dimensionality reduction allows the identification of the main features of the dataset while reducing noise. The singular values indicate how important each feature is.

The hyperparameters used for searching the best possible combination are the following:

- **n_factors:** Represents the number of latent factors or features considered when decomposing the original matrix. The tested values for this parameter are 5, 10 and 20.
- **reg_all:** This is the regularization parameter corresponding to λ, which controls the level of penalty applied to large values of the latent factors. The tested values for this parameter are 0.01, 0.1 and 0.2.
- **lr_all:** This is the global learning rate of the model used in optimization. The tested values for this parameter are 0.001, 0.01 and 0.05.
- **n_epochs:** This is the maximum number of iterations. The tested values for this parameter are 50, 100 and 150.

2.6 BPR: Bayesian Personalized Ranking

It is an algorithm whose objective is to create a personalized ranking of recommendations for each user (Wang and Han, 2020). Pointwise learning is used through a regression method that minimizes the squared loss between the actual and predicted values. The main goal of this algorithm is to maximize the distance between items preferred by the user and those not preferred. This allows the items to be correctly ranked according to user preference.

The hyperparameters used for searching the best possible combination are the following:

- **k:** Represents the number of latent factors or features considered when decomposing the original matrix. The tested values for this parameter are 5, 10 and 20.
- **lambda_reg:** This is the regularization parameter corresponding to λ, which controls the level of penalty applied to large values of the latent factors. The tested values for this parameter are 0.001 and 0.01.
- **learning_rate:** This is the global learning rate of the model used in optimization. The tested values for this parameter are 0.01, 0.1, 0.4 and 0.6.
- **max_iter:** This is the maximum number of iterations. The tested values for this parameter are 100 and 150.

2.7 NCF: Neural Collaborative Filtering

It is a hybrid algorithm that combines two algorithms to model interactions between users and items (He et al., 2017). One of the algorithms used is GMF, which is a generalized matrix factorization algorithm. It introduces a predictable activation function that allows for modeling linear relationships between users and items.

The other algorithm used is MLP, which is a multilayer perceptron algorithm. It uses a deep neural network that captures nonlinear and complex relationships between users and items.

The hyperparameters used for searching the best possible combination are the following:

- **embed_size:** This is the size of the latent representations between users and items. The tested values for this parameter are 8, 16 and 32.
- **batch_size:** This is the size of the batches in which the data will be divided during training. The tested values for this parameter are 1024 and 2048.
- **learning_rate:** This is the global learning rate of the model used in optimization. The tested values for this parameter are 0.001, 0.01 and 0.1.
- **n_epochs:** This is the maximum number of iterations. The tested values for this parameter are 10, 15 and 20.

3 Results

When evaluating the different algorithms, two types of metrics will be used: rating and ranking (Karatzoglou et al., 2013). Rating metrics measure the difference between the ratings predicted by the algorithm and the original ones. Ranking metrics evaluate the ability of the systems to correctly order the elements based on their relevance to users.

The rating metrics used are RMSE, MAE and R^2 while the ranking metrics correspond to the Precision@k, Recall@k and NDGC@k metrics. For the three ranking metrics, 10 has been used as the value of the parameter k.

The different algorithms have been trained and evaluated by testing various combinations of hyperparameters on these metrics. The best combination for each algorithm will be specified below according to the hyperparameters tested:

- For the ALS algorithm the best combination of hyperparameters is 65 of rank, 0.3 of regparam and 50 of maxiter.
- For the NMF algorithm the best combination of hyperparameters is 50 of n_factors, 0.1 of reg_pi, 0.02 of reg_qu and 100 of n_epochs.
- For the SVD algorithm the best combination of hyperparameters is 20 of n_factors, 0.2 of reg_all, 0.01 of lr_all and 50 of n_epochs.
- For the BPR algorithm the best combination of hyperparameters is 5 of k, 0.001 of lamda_-reg, 0.4 of learning_rate and 150 of max_iter.
- For the NCF algorithm the best combination of hyperparameters is 32 of embed_size, 2048 of batch_size, 0.01 of learning_rate and n_epochs of 20.

Table 1: Rating metrics of the models using the best combination of parameters for each algorithm used.

Rating metric	**ALS**	**NMF**	**SVD**	**BPR**	**NCF**
RMSE	1.1009	0.9418	0.8748	1.6306	0.9985
MAE	0.8822	0.6022	0.6105	0.9547	0.6746
R2	-0.207	-0.0844	0.0724	-1.6901	-0.0156

Table 2: Evaluated ranking metrics of the models using the best combination of parameters for each algorithm used.

Ranking metric	**ALS**	**NMF**	**SVD**	**BPR**	**NCF**
NDCG@10	0.9320	0.8769	0.8764	0.9133	0.9123
Precision@10	0.3068	0.7551	0.8778	0.8394	0.7872
Recall@10	0.9008	0.7591	0.8055	0.8765	0.7642

In table 1 we can see the rating metrics obtained with the best combination of hyperparameters. It can be seen that in the case of rating metrics, the algorithms that provide good results are SVD and NMF. These provide low MAE and RMSE metrics, below 1, which indicates a good approximation of the rating predicted by the model and the original. Other algorithms such as ALS or NCF do not provide such good results but are quite acceptable. They present an RMSE around 1 in both cases and an MAE of 0.88 in the case of ALS and 0.67 in the case of NCF. In the case of the BPR algorithm, they do not provide good results in these metrics since the main objective is to perform a ranking and not to predict the ratings. This is reflected in the RMSE metric, which is 1.6, which is too high, and the MAE metric, which has a value very close to 1.

Table 2 shows the ranking metrics obtained with the best combination of hyperparameters. In this case, the algorithms that give us good results are the SVD and BPR algorithms. In the case of SVD, it gives us a precision metric of 0.87 and a recall of 0.80. In the case of BPR, the precision is 0.83 and the recall is 0.87. In both algorithms these results are very good. Other algorithms that give acceptable results but not the best are the NCF and NMF algorithms. In this case, NMF in both precision and recall do not give values of 0.75, while NCF presents 0.78 in precision and 0.76 in recall. Finally, the ALs algorithm gives us poor results in this metric, since its precision is very low, at 0.3.

To find out which is the best recommendation system algorithm for the data set available, we must not only take into account the result of the metrics but also the execution time of the algorithm. This is because an algorithm can give us very good results, but if it takes a long time to execute, it is not profitable when it comes to implementation. This information is collected in table 3.

Table 3: Resources used by each algorithm. The second column, Static Training Time, indicates the time taken to train the model using a fixed set of hyperparameters. The third column, Total Training Time, represents the overall training time, which includes the time required to evaluate all tested hyperparameter combinations through grid search along with the 10-fold cross-validation process. The last column indicates the amount of memory used by the algorithm during the training step.

Algorithm	**Static Training time**	**Total Training Time** (Grid Search + CV)	**Memory**
ALS	15 min	94 h	40 GiB
NMF	1 h 45 min	1240 h	50 GiB
SVD	1 h 15 min	990 h	30 GiB
BPR	5 min	50 h	20 GiB
NCF	45 min	445 h	20 GiB

4 Conclusion

In this work, a study has been carried out on 5 algorithms for recommendation systems, based on a database of outdoor activities and sports. The main objective is to determine which of them is the most suitable for making relevant recommendations for each user. To do so, both the rating and ranking metrics and the execution time of the algorithms will be taken into account.

The ALS algorithm provides quite acceptable rating results, while the ranking results are not suitable because the precision metric is very low. Although the total training time is low compared to other algorithms, due to the poor ranking metric, this algorithm is not considered suitable for the data set.

The NMF algorithm provides acceptable results for both the rating and ranking metrics. But its total training time is too high, and therefore it is not suitable for the selected data.

The SVD algorithm provides very good results for both the rating and ranking metrics, being the best algorithm if only these metrics are taken into account. But if you take into account the execution time, the algorithm is quite expensive, which can be a problem.

The BPR algorithm provides very bad results in the case of rating metrics, but very good results in the ranking metrics. Also if you take into account the time cost, it is a very fast algorithm. If only the ranking were taken into account, this would be the ideal algorithm for our data.

The NCF algorithm provides good results for both the rating and ranking metrics, but they are not the best. Its execution time is acceptable compared to the other algorithms, since, for example, it takes half the time of the SVD algorithm.

In conclusion, if only the ranking is taken into account, the best algorithm is the BPR since it is fast and provides very good results in this metric. If, on the other hand, both metrics are taken into account, the best algorithm is the SVD algorithm. This presents a problem due to the time cost, and only if there is enough time will it be suitable. If aligning with consumption efficiency and sustainable AI practices is necessary, the NCF algorithm delivers satisfactory results in both metrics (Bolón-Canedo et al., 2024).

Acknowledgements

We are particularly grateful to CESGA (Galician Supercomputing Centre) for providing access to their infrastructure.

Bibliography

K. Allawadi and C. Vij. A smart spotify assistance and recommendation system. In *2023 International Conference on Advancement in Computation and Computer Technologies (InCACCT)*, pages 286–291, 2023.

D. M. Belete and M. D. Huchaiah. Grid search in hyperparameter optimization of machine learning models for prediction of hiv/aids test results. *International Journal of Computers and Applications*, 44(9):875–886, 2022.

V. Bolón-Canedo, L. Morán-Fernández, B. Cancela, and A. Alonso-Betanzos. A review of green artificial intelligence: Towards a more sustainable future. *Neurocomputing*, 599:128096, 2024.

S. Y. Chang, H.-C. Wu, K. Yan, X. Chen, S. C.-H. Huang, and Y. Wu. Personalized multimedia recommendation systems using higher-order tensor singular-value-decomposition. *IEEE Transactions on Broadcasting*, 70(1):148–160, 2024.

S. Gupta and M. Dave. *An Overview of Recommendation System: Methods and Techniques*, page 231–237. Springer Singapore, 2020. ISBN 9789811502224.

X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua. Neural collaborative filtering, 2017.

T. Ilias and J. A. andEl Mhamdi Jamal. Impact of the choice of cross-validation techniques on the results of machine learning-based diagnostic applications. *hir*, 27(3):189–199, 2021.

Jianmo Ni. Amazon review data (2018). *https://nijianmo.github.io/amazon/index.html*, 2018. [Online; accessed 13-September-2024].

A. Karatzoglou, L. Baltrunas, and Y. Shi. Learning to rank for recommender systems. In *Proceedings of the 7th ACM Conference on Recommender Systems*, pages 493–494, 2013.

M. R. Nabavi and S. N. Nihtianov. Design strategies for eddy-current displacement sensor systems: Review and recommendations. *IEEE Sensors Journal*, 12(12):3346–3355, 2012.

J. Ni, J. Li, and J. McAuley. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pages 188–197, 2019.

E. Reutskaja, A. Lindner, R. Nagel, R. A. Andersen, and C. F. Camerer. Choice overload reduces neural signatures of choice set value in dorsal striatum and anterior cingulate cortex. *Nature Human Behaviour*, 2(12):925–935, Oct. 2018.

G. Takács and D. Tikk. Alternating least squares for personalized ranking. In *Proceedings of the Sixth ACM Conference on Recommender Systems*, RecSys '12, page 83–90, New York, NY, USA, 2012. Association for Computing Machinery. ISBN 9781450312707.

J. Wang and P. Han. Adversarial training-based mean bayesian personalized ranking for recommender system. *IEEE Access*, 8:7958–7968, 2020.

Y. X. Xin Luo, Mengchu Zhou and Q. Zhu. An efficient non-negative matrix-factorization-based approach to collaborative filtering for recommender systems. *IEEE Transactions on Industrial Informatics*, 10(2):1273–1284, 2014.