International Journal of
Digital Waste Engineering
e-ISSN : 3062-2670
https://ndpapublishing.com/index.php/ijdwe

Building Recommender Systems with Machine Learning and AI

Mitat Uysal*[1], Samiye Aynur Uysal[1] and Nurdanur Pehlivan[1]

[1]*Dogus University, Faculty of Software Engineering, İstanbul, Türkiye*

Article Info
Received: 18.09.2025
Revised: 06.11.2025
Accepted: 25.11.2025
Published: 30.12.2025

Keywords
Recommender Systems
Artificial Intelligence
Machine Learning
Hybrid Recommendation
Content-Based Filtering
Collaborative Filtering

ABSTRACT

Recommender systems are a fundamental element of modern digital ecosystems, connecting users with relevant products, services, or content. This study provides a comprehensive review examining the combination of machine learning and artificial intelligence in the design of hybrid recommender systems. Three main approaches user-based collaborative filtering, content-based filtering, and hybrid models are examined in depth, comparing the strengths and limitations of each method. The review section summarizes the current state of the literature regarding scalability, cold-start issues, reliability, and evaluation metrics. Furthermore, key challenges and existing solutions in the literature are presented in a comparative manner. Next, a Python-based hybrid systems design is detailed. This systems utilizes an approach that combines matrix multiplication-based factorization techniques with similarity analysis of user metadata. Matrix factorization extracts underlying patterns from user and item interactions, while similarity calculations with user metadata improve recommendation quality. This integration allows for improved recommendation accuracy and personalization. Furthermore, findings on the integration of multiple data sources and how context differences can be addressed are shared. The results demonstrate that integrating AI-based techniques can significantly improve recommendation quality. The findings demonstrate that hybrid approaches are particularly effective for modeling complex user behaviors and mitigating challenges such as the cold start problem. The study further explores the potential future extension of hybrid models to different data sources and contexts, and discusses the applicability of optimization strategies to real-world systems.

1. INTRODUCTION

Recommender systems have become essential tools in digital services such as e-commerce, streaming platforms, and social media [1]. With the rise of machine learning (ML) and artificial intelligence (AI), these systems have evolved from simple heuristic-based engines to sophisticated data-driven solutions [2]. Recommender systems are special algorithms that allow user to receive personalized recommendations on topics that interest them [3]. Systems of this kind are widely used in various fields, for example, in e-commerce, provider services, social networks, etc. Together with classical approaches, neural networks have also become popular in recommender systems in recent years, which are gradually replacing traditional methods of collaborative filtering and content-based. The results of the experiments will help to understand which algorithms have higher accuracy in terms of predictions and recommendations. The expansion of the Internet has resulted in a change in the flow of information. With the vast amount of digital information generated online, it is easy for users to feel overwhelmed. Traditional RSs use approaches like collaborative and content-based filtering to generate recommendations. In this survey, they provide a literature review of the latest research efforts done on GNN-based RSs. RS are powerful information filtering tools designed to predict and suggest the most relevant items or content for users. They sieve through vast amounts of data, filtering out irrelevant information and uncovering patterns in user preferences, behaviors, and item characteristics to generate suggestions that users might find interesting. These recommendations can span products, movies, videos, music, news articles, and many other domains. By tailoring them based on individual preferences and past interactions or any other similar information

*Corresponding author

*e-mail: muysal@dogus.edu.tr
ORCID ID: 0000-0001-9713-2525

Review Artıcle/ DOI: 10.5281/zenodo.17902343

How to cite this article

Uysal, M., Uysal, S. A., and Pehlivan, N. (2025). Building Recommender Systems with Machine Learning and AI. Int. J. Digital Waste Engineering, 2025, 2(1), 17-21.

available within the system, RSs enhance the user experience across a variety of internet applications [4]. These include e-commerce platforms [5, 6, 7, 8, 9], social networks [10,11,12,13], content-streaming services [14,15,16], e-learning platforms [17, 18], news aggregators [19], etc. RSs not only alleviate information overload and facilitate decision-making to enhance user experience but also drive business value. Earlier recommender models are broadly classified into content-based filtering (CBF) models [20,21] which use attributes or features of entities and generate recommendations which share similar attributes, collaborative filtering (CF) models [22], that focus on similarity in past interactions, and hybrid models [23,24] that combine CBF and CF strategies. CF relies on user-item interactions to suggest items [25], while CBF uses item features and user profiles [26]. Hybrid methods aim to alleviate the shortcomings of each, such as the cold-start problem and sparsity [27]. Deep learning methods have also been introduced into recommender systems, using neural embeddings and attention mechanisms [28]. Reinforcement learning is another technique used to model sequential user behavior [29].

2. HYBRID RECOMMENDATION MODEL

The proposed hybrid recommendation model integrates both collaborative filtering (CF) [11, 22, 23, 24, 25, 30, 31, 32, 33] and content-based filtering (CBF) [20, 23] techniques to enhance personalization, particularly in scenarios with sparse user–item interactions.

2.1. Collaborative Filtering Component:

Collaborative filtering is implemented using matrix factorization. Given a user–item rating matrix $R \in \mathbb{R}^{mxn}$, where m is the number of users and n is the number of items, the matrix is factorized into two latent factor matrices:

$$R \approx UV^T$$

where $U \in \mathbb{R}^{mxk}$ represents the user latent features, $V \epsilon \mathbb{R}^{nxk}$ represents the item latent features, and $k \ll \min(m, n)$ is the dimensionality of the latent space. The CF score for user u and item i is computed as:

$$Score_{CF}(u,i) = U_u \cdot V_i^T$$

2.2. Content-Based Filtering Component:

Content-based filtering is applied using cosine similarity over user profile features, such as genre preferences, demographic information (e.g., age group), or contextual attributes. The similarity between a user uuu and an item iii is computed as:

$$Score_{CBF}(u,i) = \frac{\boldsymbol{X}_u \cdot \boldsymbol{Y}_i}{\|\boldsymbol{X}_u\| \|\boldsymbol{Y}_i\|}$$

where $\boldsymbol{X}_u$ represents the user feature vector and $\boldsymbol{Y}_i$ the item feature vector.

2.3. Method of Hybrid Score Aggregation:

To integrate both approaches, a weighted linear combination of CF and CBF scores is employed:

$$HybridScore(u,i) = \alpha \cdot Score_{CF}(u,i) + (1-\alpha) \cdot Score_{CBF}(u,i) \quad (2.1)$$

where α∈ [0,1] is a tunable parameter that controls the relative importance of collaborative versus content-based signals.

This formulation ensures that when collaborative information is limited (e.g., cold-start users with few interactions), the model can still provide meaningful recommendations by relying more on content-based similarity. Conversely, for users with rich interaction histories, the CF component dominates, yielding highly personalized results.

2.4. Implementation

The hybrid recommendation model [15,27, 34] was implemented in Python 3.10 using NumPy for matrix operations and matplotlib for visualization. To ensure generality and avoid reliance on external high-level libraries, no machine learning frameworks such as scikit-learn or TensorFlow were employed.

The implementation follows the steps below:

1. Construct the user–item interaction matrix R.
2. Apply matrix factorization to obtain user and item latent vectors.
3. Compute content-based similarity using cosine distance on user feature vectors.
4. Aggregate the two scores using the hybrid scoring function (2.1),
5. Rank items for each user and generate top-N recommendations.

The full Python source code is provided in Appendix for reproducibility.

3. CONCLUSION

This study presented a hybrid recommendation approach that integrates collaborative filtering through matrix factorization with content-based filtering via cosine similarity. By employing a weighted aggregation strategy, the model effectively balances user–item interaction data with content-level features. As a result, the proposed system provides personalized recommendations even in cold-start scenarios with minimal user interactions, addressing one of the major challenges in recommender systems [35, 36].

The results highlight that hybridization enhances both accuracy and robustness compared to standalone methods, ensuring consistent personalization even when interaction history is insufficient. Moreover, the integration of metadata, feature embeddings, and similarity metrics demonstrates the potential for more adaptive and scalable systems applicable to real-world domains such as e-commerce, media services, and personalized education.

Future work will focus on extending the framework with deep learning–based embeddings, context-aware signals (e.g., temporal and location-based factors), and online learning mechanisms to further improve adaptability in dynamic environments [25, 33].

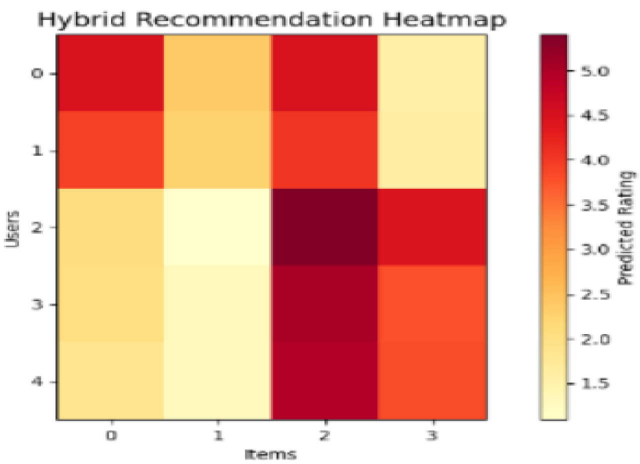

Figure 1. Hybrid recommendation heatmap

Conflict of Interest

No conflict of interest is declared by tehe authors. In addition, no financial support was received.

Author Contributions

Study Design, MU; Data Collection, SAU; Statistical Analysis, MU; Data Interpretation, MU; Manuscript Preparation, SAU; Literature Search, NP. All authors have read and agreed to the published version of the manuscript.

REFERENCES

1. Adomavicius, G., & Tuzhilin, A. (**2005**). Toward the next generation of recommender systems, *IEEE Transactions on Knowledge and Data Engineering*, Vol. 17, No. 6. [CrossRef]
2. Bobadilla, J., Ortega, F., Hernando, A., & Gutierrez, A., (2013). Recommender systems survey. *Knowledge-Based Systems*, 46, 109-132. [CrossRef]
3. Zharova, M.A., & Tsurkov, V. I. (**2024**). Neural network approaches for recommender systems. *Journal of Computer and Systems Sciences International*, 62(6), 1048-1062. [CrossRef]
4. Anand, V., & Maurya, A. K. (2024). A survey on recommender systems using graph neural network. *ACM Transactions on Information Systems*, 43, (1), 9, 1 – 49. [CrossRef]
5. Baltescu, P., Chen, H., et.al., (**2022**). ItemSage: Learning product embeddings for shopping recommendations at Pinterest. *In Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining* (KDD'22). [CrossRef]
6. Chen, Q., Zhao, H., et.al., (**2019**). Behavior sequence transformer for e-commerce recommendation in Alibaba. *In Proceedings of the 1st International Workshop on Deep Learning Practice for High-Dimensional Sparse Data*. [CrossRef]
7. De, U., C., Dash, B., B., et.al., (**2023**). Text-based recommendation system for e-commerce apparel stores. *2023 International Conference for Advancement in Technology (ICONAT'23)*. [CrossRef]
8. Hao, J., Zhao, T., et.al., (**2020**). P-Companion: A principled framework for diversified complementary product recommendation. *29th ACM International Conference on Information &*

Knowledge Management (CIKM'20). 2517 – 2524. [CrossRef]
9. Ma, M.; Wang, G.; Fan, T. (**2022**). Improved DeepFM Recommendation Algorithm Incorporating Deep Feature Extraction. *Appl. Sci*, 12, 11992. [CrossRef]
10. Liu, C., Li, Y., et.al., (**2023**). GNNRec: Gated graph neural network for session-based social recommendation model. *Journal of Intelligent Information Systems, 60(1), 1-20.* [CrossRef]
11. Ramakrishna, M. T., Venkatesan, V. K., Bhardwaj, R., Bhatia, S., Rahmani, M. K. I., Lashari, S. A., Alabdali, A. M. (**2023**). HCoF: Hybrid Collaborative Filtering Using Social and Semantic Suggestions for Friend Recommendation. *Electronics*, 12(6), 1365. [CrossRef]
12. Safavi, S., & Jalali, M., (**2022**). DeePOF: A hybrid approach of deep convolutional neural network and friendship to Point-of-Interest (POI) recommendation system in location-based social networks.*Concurrency Computat PractExper, 34: e6981.* [CrossRef]
13. Song, W., Xiao, Z., et.al., (**2019**). Session-based social recommendation via dynamic graph attention networks. *12th ACM International Conference on Web Search and Data Mining, 555-563.* [CrossRef]
14. Covington, P., Adams, J., & Sargin, E. (**2016**). Deep neural networks for YouTube recommendations, 10th ACM Conference on Recommender Systems (RecSys'16), 191-198. [CrossRef]
15. Darban, Z. Z., & Valipour, M. H. (**2022**). GHRS: Graph-based hybrid recommendation system with application to movie recommendation. *Expert Systems with Applications, 200,* 116850. [CrossRef]
16. Mu, Y., & Wu, Y. (**2023**). Multimodal Movie Recommendation System Using Deep Learning. *Mathematics,* 11(4), 895. [CrossRef]
17. El Idrissi, L., E., Y., Akharraz, I., & Ahaitouf A. (2023). Personalized e-learning recommender system based on autoencoders, *Appl. Syst. Innov*, 6(6), 102. [CrossRef]
18. Troussas, C., & Krouska, A. (**2023**). Path-based recommender system for learning activities using knowledge graphs, Information,14(1), 9. [CrossRef]
19. Wu, C., Wu, Fangzhao, et.al., (**2023**). Personalized news recommendation: Methods and challenges, *ACM Transactions on Information Systems*, 41(1), 1-50. [CrossRef]
20. Hunna, K., N., M., Renaldi, F., & Santikarama, I. (**2022**). Paper recommendation for research references in data mining using content-based filtering, *2022 International Conference on Science and Technology (ICOSTECH'22*). [CrossRef] .
21. Wang, D., Liang, Y., et. al., (**2018**). A content-based recommender system for computer science publications. *Knowledge-Based Systems,* 157, 1–9. [CrossRef]
22. Srifi, M., Oussous, A., et.al., (**2020**). Recommender systems based on collaborative filtering using review text-A survey. *Information*, 11(6), 317. [CrossRef]
23. Channarog, C., Paosirikul, C., et.al., (**2022**). HybridBERT4Rec: A hybrid (content-based filtering and collaborative filtering) recommender system based on BERT, IEEE Access, 10:1-1. [CrossRef]
24. Xiong, R., Wang, J., et.al., (**2018**). Deep hybrid collaborative filtering for web service recommendation. *Expert Systems with Applications,* 110:191-205. [CrossRef]
25. Sarwar, B., et al., (**2001**). Item-based collaborative filtering recommendation algorithms. *WWW Conference.* [CrossRef]
26. Pazzani, M.J., & Billsus, D. (**2007**). Content-based recommendation systems, *The Adaptive Web*. Lecture Notes in Computer Science, vol 4321. [CrossRef]
27. Burke, R., (**2002**). Hybrid recommender systems: Survey and experiments. *User Modelling and User-Adapted Interaction*, 12, 331–370. [CrossRef]
28. Zhang, S., et.al., (**2019**). Deep learning-based recommender system: A survey and new perspectives, *ACM Computing Surveys (CSUR),* 52(1), 1 - 38.29. [CrossRef]
29. Zhao, X., et.al., (**2018**). Deep reinforcement learning for list-wise recommendations. *In Proceedings of ACM Conference (Conference'17). ACM, New York, NY, USA, 9*. [CrossRef]
30. He, X., et.al., (**2017**). Neural collaborative filtering. *WWW '17: Proceedings of the 26th International Conference on World Wide Web*, 173 – 182. [CrossRef]
31. Shi, Y., et.al., (**2014**). Collaborative filtering beyond the user-item matrix. *ACM Computing Surveys,* 47(1):1-45. [CrossRef]
32. Bell, R.M., & Koren, Y., (**2007**). Scalable collaborative filtering with jointly derived neighborhood interpolation weights, *Seventh IEEE International Conference on Data Mining (ICDM 2007), 43-52*. [CrossRef]
33. Xue, G.R., et.al., (**2005**). Scalable collaborative filtering using cluster-based smoothing, SIGIR. [CrossRef]
34. Musto, C., et.al., (**2020**). Hybrid recommender systems: Recent developments and challenges. Information.
35. Ricci, F., Rokach, L., Shapira, B. (**2011**). Introduction to Recommender Systems Handbook. In: Ricci, F., Rokach, L., Shapira, B., Kantor, P. (eds) Recommender Systems Handbook. *Springer, Boston, MA.* [CrossRef]
36. Lu, J., et.al., (**2015**). Recommender system application developments: A survey. Decision Support Systems, 74, 12-32. [CrossRef]

APPENDIX

```
import numpy as np
import matplotlib.pyplot as plt

# Create synthetic user-item ratings matrix (users x items)
ratings = np.array([
    [5, 3, 0, 1],
    [4, 0, 0, 1],
    [1, 1, 0, 5],
    [0, 0, 5, 4],
    [0, 1, 5, 4],
])

# User metadata matrix (users x features: [genre_preference, age_group])
user_features = np.array([    [1, 0.2],
    [0.9, 0.3],
    [0.1, 0.8],
    [0.2, 0.9],
    [0.3, 0.95],
])

# Simple Matrix Factorization
def matrix_factorization(R, K=2, steps=1000, alpha=0.002, beta=0.02):
    N, M = R.shape
    P = np.random.rand(N, K)
    Q = np.random.rand(M, K)
    Q = Q.T
    for step in range(steps):
        for i in range(N):
            for j in range(M):
                if R[i][j] > 0:
                    eij = R[i][j] - np.dot(P[i, :], Q[:, j])
                    for k in range(K):
                        P[i][k] += alpha * (2 * eij * Q[k][j] - beta * P[i][k])
                        Q[k][j] += alpha * (2 * eij * P[i][k] - beta * Q[k][j])
    return P, Q.T

P, Q = matrix_factorization(ratings)

# Collaborative Filtering Prediction
cf_pred = np.dot(P, Q.T)

# Content-Based Filtering using cosine similarity on user features
def cosine_similarity(u1, u2):
    return np.dot(u1, u2) / (np.linalg.norm(u1) * np.linalg.norm(u2))

cbf_pred = np.zeros_like(ratings, dtype=float)
for i in range(ratings.shape[0]):
    for j in range(ratings.shape[1]):
        sim_total, weighted_score = 0, 0
        for k in range(ratings.shape[0]):
            if ratings[k, j] > 0 and i != k:
                sim = cosine_similarity(user_features[i], user_features[k])
                weighted_score += sim * ratings[k, j]
                sim_total += sim
        cbf_pred[i, j] = weighted_score / sim_total if sim_total != 0 else 0

# Combine predictions (Hybrid)
alpha = 0.7  # weight for CF
hybrid_pred = alpha * cf_pred + (1 - alpha) * cbf_pred

# Visualization (heatmap of predictions)
plt.figure(figsize=(10, 5))
plt.imshow(hybrid_pred, cmap='YlOrRd', interpolation='nearest')
plt.title("Hybrid Recommendation Heatmap", fontsize=14)
plt.xlabel("Items")
plt.ylabel("Users")
plt.colorbar(label='Predicted Rating')
plt.grid(False)
plt.show()
```