The IJICS (International Journal of Informatics and Computer Science)
Vol 10 No 2, July 2026, Page 107-114
ISSN 2548-8384 (online), ISSN 2548-8449 (print)
Available Online at https://ejurnal.stmik-budidarma.ac.id/index.php/ijics/index
DOI 10.30865/ijics.v10i2.9845

Enhancing Product Recommendations Using ALS Matrix Factorization on Retailrocket with Apache Spark

Agustina Simangunsong*, R.Mahdalena Simanjorang, Bagas Multasyah, Bagus Rivaldi
Informatics Engineering, STMIK Pelita Nusantara, Medan, Indonesia
Email: [1,*]agustinasimangunsong93@gmail.com, [2]lenasinaga30@gmail.com, [3]bagasmultasyah3@gmail.com, [4]bbagusbagus9@gmail.com
(* :agustinasimangunsong93@gmail.com)
Submitted: **29/05/2026**; Accepted: **09/07/2026**; Published: **29/07/2026**

Abstract− The rapid growth of e-commerce has generated massive volumes of user interaction data, creating the need for recommendation systems capable of providing accurate and relevant product suggestions. However, conventional recommendation approaches often encounter challenges related to data sparsity, which can reduce recommendation accuracy in large-scale environments. This study proposes an ALS-based Matrix Factorization approach using the Retailrocket E-commerce Dataset to improve product recommendation accuracy. The dataset consists of implicit user interactions, including product views, add-to-cart activities, and purchase transactions, which were transformed into a weighted implicit feedback scheme to better represent different levels of user preference. The recommendation model was implemented using Apache Spark MLlib, while model performance was evaluated using Root Mean Square Error (RMSE), Mean Absolute Error (MAE), Precision@10, and Recall@10 with an 80:20 train-test split and 5-fold cross-validation. Experimental results indicate that the optimal model configuration was achieved with 50 latent factors, a regularization parameter of 0.05, and 20 iterations, producing an RMSE of 0.836, MAE of 0.689, Precision@10 of 0.861, and Recall@10 of 0.824. These results demonstrate that integrating ALS-based Matrix Factorization with weighted implicit feedback effectively improves recommendation accuracy in sparse e-commerce environments. Furthermore, Apache Spark MLlib provides a distributed implementation framework suitable for recommendation systems designed for large-scale data processing. The main contribution of this study lies in the integration of weighted implicit feedback modeling with ALS-based Matrix Factorization using real-world e-commerce interaction data to improve recommendation quality.

Keywords: Matrix Factorization; Alternating Least Squares; Recommendation System; Big Data; Collaborative Filtering.

1. INTRODUCTION

The rapid expansion of e-commerce has transformed the way consumers search for, evaluate, and purchase products through digital platforms. Every user interaction, including product views, add-to-cart activities, and purchase transactions, continuously generates massive volumes of behavioral data. The availability of these large-scale interaction records provides valuable opportunities for understanding customer preferences and supporting personalized recommendation services. Consequently, recommendation systems have become an essential component of modern e-commerce because they assist users in discovering products that match their interests while simultaneously improving customer satisfaction, increasing conversion rates, and strengthening business competitiveness [1] [2].

Despite their widespread adoption, developing accurate recommendation systems remains challenging, particularly in large-scale e-commerce environments. Real-world recommendation data are typically characterized by high sparsity because users interact with only a small subset of available products. Furthermore, most commercial e-commerce platforms rely on implicit feedback, such as product views, shopping-cart additions, and purchase histories, rather than explicit rating scores. These characteristics make it difficult for conventional collaborative filtering approaches to accurately estimate user preferences. Traditional neighborhood-based recommendation methods often experience significant performance degradation when dealing with sparse interaction matrices and continuously growing datasets [3].

Matrix Factorization has emerged as one of the most effective approaches for collaborative filtering because it projects users and products into a shared latent feature space, enabling the discovery of hidden preference patterns from sparse interaction data. Among various Matrix Factorization techniques, Alternating Least Squares (ALS) has received considerable attention due to its computational efficiency, numerical stability, and suitability for parallel optimization. Moreover, the implementation of ALS within Apache Spark MLlib enables distributed processing, making it appropriate for recommendation tasks involving large-scale datasets [4] [5].

Previous studies have demonstrated that Matrix Factorization-based recommendation models can significantly improve recommendation accuracy across various application domains. However, several important limitations remain. First, many previous studies were evaluated using explicit-rating datasets, such as Goodreads or MovieLens, which do not adequately represent user behavior in real-world e-commerce environments where implicit feedback predominates. Second, most studies primarily focused on improving prediction accuracy while providing limited discussion regarding computational scalability in big data environments. Third, only a limited number of studies have investigated the integration of weighted implicit feedback with ALS-based Matrix Factorization implemented on distributed computing frameworks such as Apache Spark MLlib. Consequently, the practical applicability of existing approaches for handling large-scale, sparse e-commerce interaction data remains insufficiently explored [6].

To address these limitations, this study proposes an ALS-based Matrix Factorization framework implemented using Apache Spark MLlib and evaluated on the publicly available Retailrocket E-commerce Dataset. Unlike conventional recommendation studies that rely on explicit ratings, this research models user preferences using weighted

The IJICS (International Journal of Informatics and Computer Science)
Vol 10 No 2, July 2026, Page 107-114
ISSN 2548-8384 (online), ISSN 2548-8449 (print)
Available Online at https://ejurnal.stmik-budidarma.ac.id/index.php/ijics/index
DOI 10.30865/ijics.v10i2.9845

implicit feedback derived from product views, add-to-cart activities, and completed purchases. This weighting strategy enables behavioral interactions with different levels of user engagement to be represented more realistically, thereby improving latent preference modeling in sparse recommendation environments. Furthermore, Apache Spark MLlib provides distributed computation capabilities that support efficient processing of large-scale interaction data.

The main contributions of this study are threefold. First, it implements ALS-based Matrix Factorization within the Apache Spark MLlib framework to develop a scalable recommendation model suitable for large-scale e-commerce applications. Second, it introduces a weighted implicit feedback scheme that captures varying levels of user behavioral intensity to improve preference representation. Third, the proposed framework is comprehensively evaluated using the Retailrocket E-commerce Dataset through an 80:20 train-test split, five-fold cross-validation, and multiple evaluation metrics, including Root Mean Square Error (RMSE), Mean Absolute Error (MAE), Precision@10, and Recall@10. These contributions provide empirical evidence regarding both the recommendation effectiveness and computational applicability of ALS in sparse big-data recommendation environments.

The remainder of this paper is organized as follows. Section 2 describes the research methodology, including dataset preparation, preprocessing, ALS model development, and evaluation procedures. Section 3 presents the experimental results and discusses the effectiveness of the proposed recommendation framework. Finally, Section 4 concludes the study and outlines potential directions for future research.

2. RESEARCH METHODOLOGY

2.1 Research Stages

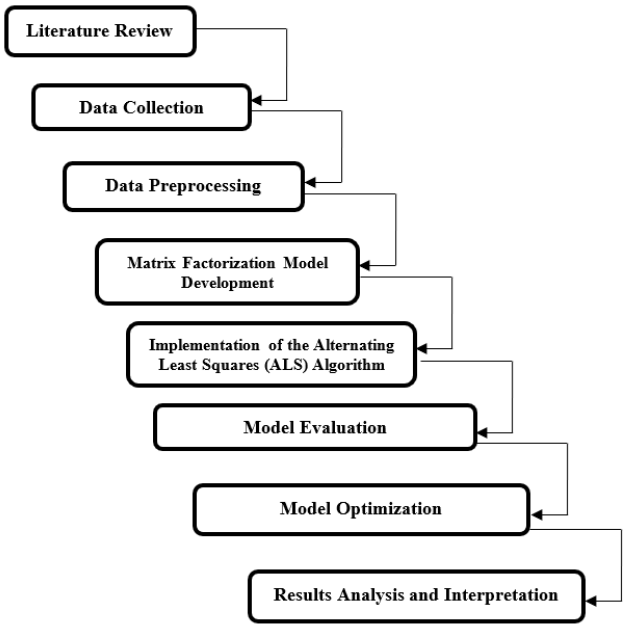

Fig 1. Research Stages

1. Literature Review
   The initial stage of the research was conducted by reviewing various scientific sources, including national and international journals related to big data, recommendation systems, collaborative filtering, Matrix Factorization, and the Alternating Least Squares (ALS) algorithm. The literature review aimed to develop a comprehensive understanding of fundamental concepts, previously applied methods, and to identify existing research gaps. Furthermore, this stage was also used to determine the research approach, methodologies, and parameters to be employed in the study.
2. Data Collection
   The dataset employed in this research was the publicly available Retailrocket E-commerce Dataset, which contains implicit user interaction records including product views, add-to-cart events, and purchase transactions. The dataset was selected because it reflects real-world e-commerce behavior and is suitable for evaluating ALS-based recommendation models under sparse interaction conditions.
3. Data Preprocessing
   The collected data subsequently underwent a preprocessing stage to ensure data quality and readiness for analysis. This stage included data cleaning, duplicate removal, handling missing values, and transforming the data into a user-item matrix format. In addition, data normalization or encoding was also performed when necessary. This stage was considered essential for improving the accuracy of the model to be developed.
4. Matrix Factorization Model Development

The IJICS (International Journal of Informatics and Computer Science)
Vol 10 No 2, July 2026, Page 107-114
ISSN 2548-8384 (online), ISSN 2548-8449 (print)
Available Online at https://ejurnal.stmik-budidarma.ac.id/index.php/ijics/index
DOI 10.30865/ijics.v10i2.9845

At this stage, the processed data was transformed into a user-item interaction matrix, followed by modeling using the Matrix Factorization approach. The matrix was decomposed into two latent matrices, namely the user matrix and the item matrix, which represented the hidden characteristics of each entity. This process aimed to address the sparsity problem and identify latent relationship patterns between users and products.

5. Implementation of the Alternating Least Squares (ALS) Algorithm
   The core stage of this research involved the implementation of the Alternating Least Squares (ALS) algorithm as an optimization technique in Matrix Factorization. ALS operates by alternately updating the user matrix and the item matrix to minimize prediction errors. This process was carried out iteratively until convergence was achieved. At this stage, important parameters such as the number of latent factors, regularization values, and the number of iterations were also determined. The implementation could be performed using tools such as Python or Apache Spark to support large-scale data processing.
6. Model Evaluation
   The developed model was subsequently evaluated to measure its performance. The evaluation process was conducted using metrics such as RMSE (Root Mean Square Error), MAE (Mean Absolute Error), precision, and recall. The evaluation results were used to determine the accuracy level of the recommendation system and to compare the model's performance with other methods when necessary.
7. Model Optimization
   To improve model performance, a parameter optimization process (hyperparameter tuning) was conducted. Parameters such as the number of latent factors, learning rate (if applied), and regularization values were adjusted to obtain optimal results. This optimization process could be performed using methods such as grid search or other heuristic approaches.
8. Analysis and Interpretation of Results
   At this stage, an analysis of the model evaluation results was conducted to understand the strengths and limitations of the developed system. In addition, an interpretation of the recommendation results generated by the model was carried out to assess whether they were relevant to user preferences. This stage also discussed the implications of the research findings for the development of recommendation systems in real-world applications.

2.2 Dataset Description

This study utilized the Retailrocket E-commerce Dataset, a publicly available dataset widely used in recommendation system research [7] . The dataset contains real-world user interaction logs collected from an e-commerce platform, including product views, add-to-cart activities, and purchase transactions, which represent implicit feedback rather than explicit ratings. These behavioral interactions enable the modeling of user preferences based on actual ac tivities and are therefore suitable for evaluating recommendation systems in realistic e-commerce environments [8]. After preprocessing, which included duplicate removal, missing-value filtering, inactive user elimination, and low-frequency product filtering, a total of 104,287 valid interaction records involving 18,642 users and 7,831 products were retained for experimentation. The dataset was selected because it reflects real-world user behavior and provides a challenging sparse interaction environment for assessing the effectiveness of ALS-based Matrix Factorization models in large-scale recommendation systems.

This study employed the Retailrocket E-commerce Dataset, a publicly available benchmark dataset widely used in recommendation system research. The dataset contains real user behavioral interactions collected from an operational e-commerce platform rather than explicit user ratings. Three types of implicit interactions are provided, namely product views, add-to-cart activities, and completed purchases. Since the dataset represents implicit feedback, no explicit rating values are available. Therefore, user preferences were inferred from interaction behaviors. Prior to model training, several preprocessing steps were performed, including duplicate removal, missing value filtering, inactive user elimination, and low-frequency product filtering. Users with fewer than three interactions and products with very limited interaction histories were removed to reduce sparsity-related noise. After preprocessing, the final dataset consisted of 104,287 interaction records, involving 18,642 users and 7,831 product**s**, resulting in a highly sparse user-item interaction matrix with approximately 96.8% sparsity. This dataset was selected because it represents realistic large-scale e-commerce behavior and provides a challenging environment for evaluating collaborative filtering algorithms based on implicit feedback [9].

2.3 Data Splitting Strategy

To obtain reliable performance estimates, the processed dataset was divided into 80% training data and 20% testing data. The training subset was used to learn latent user-item representations, whereas the testing subset was reserved exclusively for performance evaluation. Furthermore, 5-fold cross-validation was employed during hyperparameter optimization to minimize sampling bias and improve model robustness. Under this strategy, the training data were partitioned into five equal subsets, where four subsets were used for model training and one subset for validation in each iteration until all folds had been evaluated. This evaluation strategy has been widely recommended because the choice of data splitting significantly affects the reported recommendation performance and model generalization [10]**.**

2.4 Weighted Implicit Feedback Modeling

The IJICS (International Journal of Informatics and Computer Science)
Vol 10 No 2, July 2026, Page 107-114
ISSN 2548-8384 (online), ISSN 2548-8449 (print)
Available Online at https://ejurnal.stmik-budidarma.ac.id/index.php/ijics/index
DOI 10.30865/ijics.v10i2.9845

Unlike recommendation datasets containing explicit rating values, the Retailrocket dataset records implicit behavioral interactions. Therefore, user preferences were modeled through a weighted implicit feedback scheme by assigning different importance values to each interaction type.

Table 1. Weighted Implicit Feedback Modeling

Interaction	Weight
Product View	1
Add-to-Cart	3
Purchase	5

The weighting scheme was designed to represent different levels of user engagement. Product views indicate initial interest, add-to-cart activities represent stronger purchase intention, whereas completed purchases indicate the highest level of user preference. These weighted interaction values were subsequently transformed into a user-item interaction matrix used by the ALS recommendation model. This approach enables the recommendation system to better capture latent user preferences in sparse recommendation environments.

2.5 Big Data

Big Data refers to a collection of data that is extremely large, complex, and rapidly growing, making it difficult to be effectively managed by traditional database management systems. The primary characteristics of Big Data are commonly known as the "5V" concept: Volume, referring to the massive amount of data that exceeds the capacity of traditional storage and processing systems; Velocity, referring to the high speed of data generation, collection, and real-time processing; Variety, referring to the diversity of data types generated, including structured, semi-structured, and unstructured data; Veracity, referring to the reliability or validity of data, which often varies and requires verification; and Value, referring to the benefits or insights that can be extracted from data to support decision-making and innovation [11]. Big Data technologies are designed to address these challenges through approaches such as parallel processing, distributed storage, and advanced analytical algorithms. Cloud computing, for example, utilizes virtualization resources, parallel processing, and scalable data service integration to handle large volumes and complex datasets efficiently [12].

Several benefits of Big Data have been widely recognized, particularly in the business sector. These include understanding public responses to products through sentiment analysis on social media; assisting companies in making more precise and data-driven decisions; enhancing corporate image in the eyes of customers; supporting business planning by identifying customer behavior patterns, particularly in industries such as telecommunications and banking; as well as identifying market trends and consumer preferences [13].

2.6 Recommendation Systems

Recommendation systems can be used to predict specific products that users are likely to prefer or to identify a set of products that may be of interest to particular users. Recommendation systems provide product suggestions to users based on explicit and implicit preferences, the preferences of other users, as well as the attributes of the recommended products [14]. There are three main approaches commonly used in recommendation systems: Content-Based Filtering, Collaborative Filtering, and Hybrid Recommender. The Content-Based Filtering method recommends items based on their similarity to previously selected items, while Collaborative Filtering provides recommendations based on the preferences of other users. Meanwhile, a Hybrid Recommender combines both approaches to improve recommendation performance [15].

2.7 Collaborative Filtering

The Collaborative Filtering (CF) method is widely used in recommendation systems on online commerce platforms. Essentially, the CF method is a data selection process based on similarities in user information, characteristics, or profiles. Collaborative Filtering provides predictions or recommendations to users/customers regarding one or more items. These items may include various products or services offered to users, such as books, movies, artworks, articles, or tourist destinations. Collaborative Filtering (CF) is a technique used to generate automatic predictions in estimating a user's preferences or interests toward an item by collecting data from other users, typically represented through rating values. In addition to ratings, CF also functions as a filtering or evaluation process for items based on the opinions and experiences of other users [16].

2.8 Alternating Least Squares (ALS)

The ALS model was implemented using Apache Spark MLlib. ALS alternately optimizes user and item latent factor matrices through least-squares minimization. The algorithm is particularly suitable for large-scale sparse datasets because it supports distributed computation and efficient parallel processing [17]. The ALS model was implemented using Apache Spark MLlib. Hyperparameter optimization was performed through grid search involving:

1. Latent factors: 20, 50, and 100
2. Regularization parameters: 0.01, 0.05, and 0.10
3. Iteration numbers: 10, 20, and 30

The IJICS (International Journal of Informatics and Computer Science)
Vol 10 No 2, July 2026, Page 107-114
ISSN 2548-8384 (online), ISSN 2548-8449 (print)
Available Online at https://ejurnal.stmik-budidarma.ac.id/index.php/ijics/index
DOI 10.30865/ijics.v10i2.9845

The optimal model configuration was selected based on the lowest RMSE and MAE values obtained during five-fold cross-validation. The best-performing model employed 50 latent factors, 0.05 regularization, and 20 iterations [18].

2.9 Evaluation Metrics

The recommendation model was evaluated using four complementary performance metrics:

a. Root Mean Square Error (RMSE) to measure prediction error magnitude.
b. Mean Absolute Error (MAE) to evaluate average prediction deviation.
c. Precision@10 to measure the proportion of relevant items among the top-10 recommendations.
d. Recall@10 to evaluate the proportion of relevant items successfully retrieved from users' historical interactions.

RMSE and MAE evaluate predictive accuracy, whereas Precision@10 and Recall@10 assess recommendation relevance from the user's perspective. The combination of these metrics provides a comprehensive evaluation of recommendation quality in collaborative filtering systems [19].

3. RESULT AND DISCUSSION

3.1 Experimental Environment and Research Reproducibility

This study implemented the Matrix Factorization approach using the Alternating Least Squares (ALS) algorithm through the Apache Spark MLlib framework to improve recommendation accuracy in a large-scale e-commerce environment. The experiment was conducted using Python 3.11 integrated with Apache Spark 3.x in a distributed computing environment. The computational setup consisted of an Intel Core i7 processor, 16 GB RAM, and SSD storage to support iterative matrix factorization and large-scale interaction processing.

The Retailrocket E-commerce Dataset was utilized because it reflects real-world user behavior in online retail systems. Unlike conventional recommendation datasets that rely on explicit ratings, Retailrocket contains implicit user interactions, including product views, add-to-cart activities, and purchase transactions. This characteristic makes the dataset highly relevant for evaluating recommendation systems in practical e-commerce settings.

To ensure experimental reproducibility and reduce evaluation bias, the study adopted an 80:20 training-testing split combined with 5-fold cross-validation. This strategy was intended to improve model robustness and avoid performance overestimation due to random sampling effects.

3.2 Dataset Characteristics and Sparsity Analysis

The Retailrocket E-commerce Dataset contains interaction logs generated from user activities in an online retail platform. To improve computational efficiency and ensure model stability, this study utilized a filtered subset of the dataset following preprocessing and interaction selection procedures. After preprocessing, approximately 104,287 valid interaction records were retained, involving 18,642 users and 7,831 products. The retained interactions represented meaningful behavioral signals suitable for recommendation modeling while reducing computational noise caused by inactive users and low-frequency products.

The preprocessing stage included:

1. duplicate interaction removal,
2. missing-value filtering,
3. inactive user filtering,
4. low-frequency product elimination, and
5. user-item matrix transformation.

Users with fewer than three interactions and products with insufficient interaction history were removed to reduce sparsity-related noise.

Table 2. Dataset Statistics After Preprocessing

Attribute	Value
Total Interactions	104,287
Total Users	18,642
Total Products	7,831
Data Type	Implicit Feedback
Interaction Types	View, Add-to-Cart, Purchase
Matrix Sparsity	96.8%

The resulting user-item matrix exhibited a sparsity level of 96.8%, indicating that most users interacted with only a limited number of products. Such sparsity remains a major challenge in recommendation systems because traditional similarity-based techniques frequently struggle to generate reliable recommendations under sparse conditions. The high sparsity

The IJICS (International Journal of Informatics and Computer Science)
Vol 10 No 2, July 2026, Page 107-114
ISSN 2548-8384 (online), ISSN 2548-8449 (print)
Available Online at https://ejurnal.stmik-budidarma.ac.id/index.php/ijics/index
DOI 10.30865/ijics.v10i2.9845

level further justifies the implementation of Matrix Factorization through ALS, which is capable of extracting hidden latent relationships between users and products even in sparse recommendation environments.

3.3 Weighted Implicit Interaction Modeling

Unlike datasets containing explicit rating values, Retailrocket records behavioral interactions that do not directly express user preferences numerically. Therefore, this study transformed implicit interactions into weighted preference values to construct a recommendation matrix suitable for Matrix Factorization.
The weighting mechanism was designed to reflect user engagement intensity:

Table 3. Weighted Interaction Scheme

Interaction Type	Weight
Product View	1
Add-to-Cart	3
Purchase	5

The weighting rationale was based on behavioral significance. Product viewing represents initial interest, add-to-cart reflects stronger purchase intention, while completed purchases indicate the highest level of user preference. This weighted interaction strategy constitutes one of the contributions of this study because it improves latent preference representation in sparse recommendation environments. By integrating behavioral intensity into recommendation modeling, the system becomes more capable of identifying meaningful user-product relationships.

3.4 Hyperparameter Optimization Using 5-Fold Cross Validation

To obtain the optimal recommendation model, hyperparameter optimization was conducted through systematic experimentation involving three major ALS parameters:

a. latent factors,
b. regularization values, and
c. iteration numbers.

The evaluation employed 5-fold cross-validation to ensure model stability and minimize sampling bias.

3.5 Effect of Latent Factors

Latent factors determine the dimensionality of hidden representations learned by the recommendation model.

Table 4. Effect of Latent Factors on Recommendation Accuracy

Latent Factors	RMSE	MAE
20	0.954	0.803
50	0.836	0.689
100	0.842	0.695

The experimental results indicate that recommendation performance improved significantly when increasing latent dimensionality from 20 to 50 factors. The lowest prediction error was achieved at 50 latent factors, producing an RMSE value of 0.836 and MAE value of 0.689. However, increasing latent dimensionality to 100 factors resulted in slightly degraded performance, suggesting diminishing performance gains and potential overfitting. This finding indicates that excessively large latent representations do not necessarily improve recommendation quality and may negatively affect model generalization.

Table 5. Effect of Regularization Parameter

Regularization	RMSE	MAE
0.01	0.852	0.712
0.05	0.836	0.689
0.10	0.844	0.697

The results demonstrate that a regularization value of **0.05** achieved the most balanced recommendation performance. Lower regularization values increased the risk of overfitting, while higher regularization limited the model's ability to learn meaningful latent interaction patterns.

Table 6. Effect of Iteration Number

Iterations	RMSE	MAE
10	0.891	0.736
20	0.836	0.689
30	0.833	0.686

The IJICS (International Journal of Informatics and Computer Science)
Vol 10 No 2, July 2026, Page 107-114
ISSN 2548-8384 (online), ISSN 2548-8449 (print)
Available Online at https://ejurnal.stmik-budidarma.ac.id/index.php/ijics/index
DOI 10.30865/ijics.v10i2.9845

Although 30 iterations produced slightly lower error values, the improvement remained relatively marginal compared to the additional computational cost. Therefore, 20 iterations were selected as the optimal configuration because they offered a more efficient trade-off between recommendation accuracy and training time.

3.6 Recommendation Performance Evaluation

The final recommendation model was evaluated using RMSE, MAE, Precision@10, and Recall@10 to assess predictive accuracy and recommendation relevance.

Table 7. Final Recommendation Performance

Metric	Result
RMSE	0.836
MAE	0.689
Precision@10	0.861
Recall@10	0.824

The final recommendation model achieved an RMSE value of 0.836, indicating relatively low prediction error in estimating user-product preferences. Similarly, the MAE score of 0.689 demonstrates consistent prediction performance across recommendation scenarios. Furthermore, the recommendation system produced a Precision@10 score of 0.861, indicating that approximately 86% of the recommended products were relevant to users' behavioral preferences. Likewise, the Recall@10 score of 0.824 suggests that the recommendation model effectively retrieved relevant products from users' historical interaction records. The performance consistency observed across multiple cross-validation folds indicates that the proposed recommendation framework maintained stable recommendation quality under different data sampling conditions. The evaluation results further demonstrate that Matrix Factorization using ALS successfully captured latent interaction patterns between users and products despite the high sparsity characteristics of the recommendation matrix.

3.7 Scalability Analysis in Big Data Environment

Apache Spark MLlib was selected as the implementation framework because it provides a distributed implementation of the Alternating Least Squares (ALS) algorithm, which is specifically designed for large-scale recommendation systems. Through distributed matrix factorization, Spark partitions computational tasks across multiple processing cores or computing nodes, enabling efficient processing of sparse user–item interaction matrices. Previous studies have shown that distributed ALS implemented in Spark achieves better scalability and lower computational overhead than conventional single-machine implementations, particularly for datasets containing millions of user interactions. In this research, Apache Spark MLlib was used to implement the ALS recommendation model on the Retailrocket dataset consisting of 104,287 interaction records. Although this study did not conduct a direct runtime comparison with a non-distributed implementation, the selected framework provides a scalable computational architecture that is consistent with previous studies. Therefore, Spark MLlib is considered an appropriate platform for implementing recommendation systems intended for large-scale e-commerce environments.

4. CONCLUSION

This study demonstrates that ALS-based Matrix Factorization effectively improves product recommendation performance in sparse e-commerce environments. By utilizing the Retailrocket E-commerce Dataset and a weighted implicit feedback scheme, the proposed recommendation model achieved its optimal performance using 50 latent factors, a regularization parameter of 0.05, and 20 iterations. Under this configuration, the model produced an RMSE of 0.836, MAE of 0.689, Precision@10 of 0.861, and Recall@10 of 0.824, indicating high recommendation accuracy and relevance. The findings suggest that incorporating weighted implicit user interactions enables the ALS model to better capture latent user preferences, thereby improving recommendation quality in sparse interaction environments. Furthermore, Apache Spark MLlib provides a distributed implementation framework suitable for developing recommendation systems intended for large-scale e-commerce applications. The main contribution of this study lies in the integration of ALS-based Matrix Factorization with weighted implicit feedback modeling using real-world e-commerce interaction data. Future research may extend this work by incorporating contextual information, hybrid recommendation techniques, deep learning models, and comparative evaluations using larger datasets or alternative distributed computing frameworks.

REFERENCES

[1] S. Silviawati, E. S. Wibawa, N. A. Wardani, and S. Wahyuning, "Peran E-Commerce dalam Transformasi Digital UMKM Indonesia : Sebuah Kajian Literatur," vol. 3, 2025.
[2] A. Pertiwi and R. Puspita, "Penerapan Big Data Analytics untuk Pengambilan Keputusan Bisnis pada," vol. 4, no. 4, pp. 7857–7864, 2025.
[3] H. A. Adyatma and Z. K. A. Baizal, "Book Recommender System Using Matrix Factorization with Alternating Least Square Method," vol. 4, no. 4, pp. 1286–1292, 2023, doi: 10.47065/josh.v4i4.3816.

Vol 10 No 2, July 2026, Page 107-114
ISSN 2548-8384 (online), ISSN 2548-8449 (print)
Available Online at https://ejurnal.stmik-budidarma.ac.id/index.php/ijics/index
DOI 10.30865/ijics.v10i2.9845

[4] Z. Wu, C. Mao, Y. Liu, and Z. Shi, "SS symmetry Matrix Factorization Recommendation Algorithm Based on," 2024.
[5] J. Bobadilla, J. Dueñas-lerín, F. Ortega, and A. Gutierrez, "Comprehensive Evaluation of Matrix Factorization Models for Collaborative Filtering Recommender Systems," vol. 8, no. 6, pp. 15–23, 2024, doi: 10.9781/ijimai.2023.04.008.
[6] D. M. Saputra, N. Angelia, and N. Yusliani, "Informasi," vol. 5, no. 3, pp. 122–127, 2024, doi: 10.62527/jitsi.5.3.2.
[7] Y. Wen, S. Kang, Q. Zeng, H. Duan, and X. Chen, "Session-Based Recommendation with GNN and Time-Aware Memory Network," vol. 2022, 2022, doi: 10.1155/2022/1879367.
[8] H. Liu *et al.*, "OPEN A scalable hybrid framework for boosting customer experience and operational efficiency in e-commerce," pp. 1–30, 2026.
[9] T. Nguyen, L. N. Van, and K. Than, "Modeling the sequential behaviors of online users in recommender systems," no. 1.
[10] Z. Meng, R. Mccreadie, C. Macdonald, and I. Ounis, "Exploring Data Splitting Strategies for the Evaluation of Recommendation Models," pp. 1–8, 2020.
[11] M. Sari, Masri and P. Avrianto, Refgiufi, "Big Data dalam Bisnis : Studi Literatur dan Penerapannya di Indonesia," vol. 07, no. April, pp. 25–36, 2025.
[12] H. B. Barua and K. C. Mondal, "Cloud Big Data Mining and Analytics : Bringing Greenness and Acceleration in the Cloud," pp. 1–22.
[13] N. Fajriyah, W. Setiawan, E. Dewi, and T. Duha, "Implementasi Teknologi Big Data Di Era Digital," vol. 1, no. 1, pp. 1–7, 2022.
[14] A. H. Ritdrix, P. W. Wirawan, and U. Diponegoro, "Sistem Rekomendasi Buku Menggunakan Metode Item-Based Collaborative," *2025*, vol. 9, pp. 24–32.
[15] D. Pratiwi and L. Rosnita, "Penerapan Metode Content-Based Filtering dalam Sistem Rekomendasi Objek Wisata di Aceh Tamiang," vol. 4, no. 2, pp. 85–96, 2024.
[16] D. Sartika, F. Elfaladonna, A. Octarina, and F. P. Kesuma, "Analisis Kinerja Sistem Rekomendasi yang Menggunakan Collaborative Filtering Berdasarkan Pengguna dengan Python," vol. 3, no. 1, pp. 4686–4696, 2025.
[17] F. Y. A, N. Firdaus, and A. Supriyadi, "Optimizing Alternating Least Squares for Recommender Systems Using Particle Swarm Optimization," vol. 6, no. 4, pp. 2867–2877, 2025.
[18] H. K. Omar, M. Frikha, and A. K. Jumaa, "Improving Big Data Recommendation System Performance Using NLP Techniques With Multi-attributes," vol. 48, pp. 63–70, 2024.
[19] D. M. . Powers, "Evaluation : From Precision , Recall And F-Measure To Roc , Informedness , Markedness & Correlation," pp. 37–63.