Advanced Recommendation Engine: Enhancing E-Commerce Recommendations with Database Optimization and Collaborative Filtering

Aniruddha Chakravarty, Sai Mounika Peteti, Sumukh Naveen Aradhya,
Department of Computer Science
San Jose State University (SJSU), San Jose, CA, USA.
{aniruddhaprabhash.chakravarty, saimounika.peteti, sumukhnaveen.aradhya}@ sjsu.edu

***Abstract*—This project presents a comprehensive approach to designing and implementing a recommendation system tailored for e-commerce platforms. The system combines diverse recommendation algorithms, including collaborative filtering, K-Nearest neighbors (KNN) with means, Singular Value Decomposition (SVD), and Alternating Least Squares (ALS), to provide personalized product recommendations to users. The architecture prioritizes high availability and low latency, ensuring real-time responses to user interactions while acknowledging the trade-offs between consistency and availability. The project leverages a robust tech stack, encompassing data visualization using matplotlib and seaborn, data processing with pandas and scipy.stats (z-score), big data processing through PySpark and SparkContext, and additional tools like ydata-profiling and scikit-learn. The MySQL database, along with UI frameworks such as Streamlit, Flask, and Tornado, plays a vital role in handling data and user interactions efficiently.**

This project also integrates the original table with unstructured data like review comments data for the purchase transaction. Performing these algorithms are computationally heavy and including comment data, adds an additional layer of complexity, but these offer more realistic trends of the actual purchase. This offers valuable insights into algorithm selection, system design, and real-time processing trade-offs, advancing recommendation systems in the dynamic landscape of online retail. It serves as a practical guide for organizations seeking to enhance user experiences and drive business growth through intelligent recommendation systems.

***Keywords*—E-Commerce Recommendation Systems, Collaborative Filtering Models, Python, scikit-learn, PySpark Data Analytics, Alternating Least Squares (ALS) Algorithm, K-Nearest Neighbour (KNN), Singular Value Decomposition (SVD), Popularity-Based Recommendation, SQL Server Database Management, Data Preprocessing Automation, User Behavior Insights, Real-Time Recommendation Engine, Advanced Data Visualization Techniques.**

I. INTRODUCTION

In the ever-evolving realm of e-commerce, the integration of advanced data analysis and machine learning technologies has ushered in a new era of enhancing the online shopping journey. As the e-commerce landscape expands, the intricate nature of consumer preferences and behaviors necessitates the development of more sophisticated and personalized recommendation systems.

This project addresses this evolving requirement by harnessing the power of Python, scikit-learn, and PySpark to construct an advanced collaborative filtering model, underpinned by the robust Alternating Least Squares (ALS) Algorithm.

Our approach is rooted in the belief that a data-driven understanding of user interactions can markedly elevate the precision and relevance of product recommendations, thereby transforming the way consumers engage with e-commerce platforms. This innovation is driven by the escalating volume of online transactions and the extensive array of available products, often overwhelming consumers. Our system is meticulously designed to streamline this complexity, offering a refined and personalized shopping experience that caters to individual preferences. Through automated data collection, preprocessing, and a focus on key user metrics, we aim to extract profound insights that power our recommendation engine.

A. *Project Goals & Motivation:*

Our primary objective in this project is to revolutionize the e-commerce landscape through an advanced recommendation system, addressing key challenges in online shopping to boost user satisfaction and business efficiency. Our goals include:

• Sophisticated Algorithms: Implementing a suite of advanced recommendation algorithms such as Collaborative Filtering, KNN with means, Singular Value Decomposition, and Popularity-Based Recommendation. These will enable us to offer highly personalized product suggestions by analyzing user behaviors and preferences.

• Optimized Data Processing: With the immense volume of data in e-commerce, efficient data processing is crucial. We focus on handling large-scale datasets effectively, utilizing techniques like data partitioning, parallel processing with PySpark, and efficient storage methods to ensure swift and relevant recommendations.

• Merging Structured and Unstructured Data: A key strategy involves merging structured data (such as purchase histories) with unstructured data (like customer reviews). This approach allows us to run our sophisticated algorithms on a richer dataset, leading to more accurate and personalized recommendations.

• User-Friendly Interface: Developing an intuitive interface using Streamlit to make our sophisticated algorithms accessible. We prioritize simplicity and clarity, enabling users to easily interact with our system, provide feedback, and refine their preferences.

• Scalability & Flexibility: Ensuring our system is scalable and flexible to adapt to the dynamic e-commerce sector. It will be designed to efficiently manage increasing data volumes and user queries, with a modular structure for seamless updates and integration of new features."

B. *Background Information:*

The e-commerce sector, a significant part of today's global economy, presents unique challenges and opportunities. Here's an overview:

• Product Range & User Experience: E-commerce platforms offer an overwhelming range of products, posing challenges for traditional recommendation systems. Our project addresses the need for more personalized recommendations, ensuring that niche but relevant products are not overshadowed by popular ones.

• Technological Advancements: We leverage advanced machine learning techniques like Collaborative Filtering, KNN with means, Singular Value Decomposition, Popularity Based Recommendation and ALS to overcome recommendation challenges. These algorithms harness user data and preferences to deliver tailored suggestions.

• Handling Big Data: Efficient data processing is critical due to the sheer volume of e-commerce data. We harness big data technologies to process and analyze large datasets efficiently, essential for real-time recommendations.

• Sort-merge join is a powerful database operation used to combine two or more datasets, particularly effective when dealing with large volumes of data. In the context of our project, we employed sort-merge join within MySQL to merge structured transaction data with unstructured data, such as customer reviews and feedback.

• Future-Oriented Approach: Our project takes a forward-looking approach, ensuring scalability to handle growing data volumes and flexibility to integrate emerging technologies.

Section II & III contrast our methods with existing literature, emphasizing the uniqueness of our approach. Section IV outlines our methodology, including data collection, analysis, and key algorithms. Section V presents a concise overview of our system architecture and its components, supplemented by UML Diagrams. Implementation details and additional UML diagrams are discussed in Section VI.

Our evaluation strategy and the impactful results it yielded are covered in Section VII. Section VIII showcases the system in action, while Section IX analyzes the results and addresses challenges. Section X reflects on the project's learnings and potential improvements. Finally, Section XI concludes the report, summarizing our key findings, followed by a comprehensive reference list in Section XII.

II. RELATED WORK

This section examines the existing body of research relevant to our work, highlighting key contributions, limitations, and potential gaps that motivate our proposed approach.

The authors S. Shekhar et.al [10] have elucidated the concept of a join-index, a data structure used to process join queries in databases. This structure leverages precomputation techniques to accelerate query processing, particularly beneficial for tables that are not used or updated frequently. It highlights how the I/O cost of join computation using a join-index can be influenced significantly by the page-access sequence for fetching pages from base relations. The paper introduces a set of methods to compute joins using a join-index.

In their work [1], Zhao et al. presented a collaborative filtering algorithm implemented on a cloud computing platform using the MapReduce framework. The algorithm comprises three main phases: data partitioning, map, and reduce. In the data partitioning phase, user IDs are organized into files to optimize computing time and load distribution. The map phase involves initializing mappers to calculate user similarities, identify nearest neighbors, and predict ratings for items. These recommendations are sorted and stored. Finally, the reduce phase collects and sorts the recommendations, outputting them to the distributed file system. This approach harnesses the power of parallel processing and distributed computing to efficiently compute recommendations, particularly well-suited for

handling substantial datasets. However, this approach may face challenges in terms of data partitioning and storage.

The proposed approach, presented by Hafed Zarzour et al. in [2], focuses on constructing an effective recommender system with two main phases: offline model creation and online model utilization. In the offline phase, user clustering using the k-means algorithm and dimensionality reduction via Singular Value Decomposition (SVD) are employed to create a recommendation model. The online phase utilizes this model to provide accurate recommendations for active users. This approach is designed to handle datasets of varying sizes efficiently. Experimental results using MovieLens datasets demonstrate the superiority of the k-means-SVD-based recommendation method over k-means-based and k-nearest neighbor-based methods in terms of prediction accuracy, as measured by Root Mean Square Error (RMSE). The proposed approach leverages the benefits of k-means clustering and SVD techniques to enhance recommendation performance. Lastly, while this approach excels in prediction accuracy, especially for large datasets, it may require significant computational resources for offline model creation.

In [3], Liu Xiaojun proposed an innovative clustering-based collaborative filtering algorithm that addresses traditional limitations in the domain. This algorithm incorporates user and item clustering, time decay modeling for user interests, and improved similarity measurements. Clustering reduces the search space for nearest neighbors, enhancing recommendation accuracy, while time decay functions capture changing user interests. Experiments on a MovieLens dataset demonstrate the algorithm's superiority, yielding lower Mean Absolute Error (MAE) and outperforming traditional collaborative filtering algorithms (P, NMF, PS) in recall rate and accuracy rate. This represents a significant advancement in the field of recommendation systems by providing a comprehensive solution to the challenges faced by traditional collaborative filtering algorithms.

In his comprehensive research on recommendation systems, Hongjiao Liu [4] has made significant contributions to the field. His work delves into the evaluation of various recommendation algorithms using diverse datasets, including MovieLens and Coat Shopping datasets, which vary in size. These datasets encompass MovieLens, consisting of 100,000 ratings by 943 users for 1,682 movies, and the Coat Shopping dataset, featuring 290 users rating 300 coats. The algorithms under scrutiny include UserCF and ItemCF, which calculate user and item similarities to provide movie recommendations. Additionally, Matrix Factorization (MF) and Neural Collaborative Filtering (NCF) are examined. Notably, ItemCF demonstrates commendable precision but is challenged by low coverage, while MF outperforms NCF in terms of NDCG results. The study highlights the suitability of UserCF and ItemCF for specific scenarios based on user and item counts. Future research may explore combining MF and NCF to enhance recommendation accuracy.

In e-commerce recommender systems, algorithms can focus on either users or products. User-centric approaches find similar users and recommend products based on their preferences, while product-centric methods suggest items similar to what users have purchased. Common algorithms include Collaborative Filtering, Content-based Filtering, and User Clustering Models. Collaborative Filtering identifies similar users and recommends products they've purchased but faces challenges like data sparsity and scalability. Content-based Filtering suggests

items based on attributes and quickly adapts to user preferences. Cluster Models segment users and recommend products based on similar users. Handling big data requires distributed processing like Hadoop and adapting to changing user behavior, as discussed by Xuesong Zhao [5].

Approach [2] by Hafed Zarzour et al. excels due to its well-structured two-phase model, providing superior prediction accuracy on various dataset sizes, ensuring efficient handling of fluctuating data volumes. On the other hand, approach [5] highlighted by Xuesong Zhao emphasizes the flexibility of choosing between user-centric and product-centric algorithms, addressing challenges like data sparsity and scalability through distributed processing. Combining elements from both approaches, a hybrid recommendation system can be developed that leverages the predictive accuracy and scalability of approach [2] while incorporating adaptability to changing user behaviors and data volumes, as emphasized in [5]. This hybrid approach aims to provide accurate real-time recommendations tailored to specific e-commerce use cases.

III. DATABASE SYSTEM CHALLENGES

In the realm of our project, several notable challenges emerged within the database systems domain. The foremost challenge encompassed the management of an extensive dataset, boasting approximately 8 million records. This encompassed various facets, including the intricacies of data storage, retrieval efficiency, and the overarching performance of the system. Furthermore, delving into the intricacies of data preprocessing unveiled a complexity that demanded meticulous attention. This included the intricate task of handling missing data and outliers. In developing our real-time e-commerce recommendation system, we faced challenges in rapidly processing data and integrating diverse algorithms like Collaborative Filtering and Popularity-Based Recommendation. Combining structured transaction data with unstructured data such as customer reviews introduced complexities in data compatibility and integrity. However, by utilizing streaming clustering algorithms like k-means and ALS, we effectively merged these data types. Sophisticated SQL queries and tools like PySpark enabled us to overcome these challenges, ensuring our system's adaptability to user behaviors and market trends while maintaining high availability and partition tolerance for real-time, personalized recommendations.

In their work [6], Subasish Gosh et al. employed matrix factorization and the Alternating Least Squares (ALS) algorithm to enhance personalized recommendations in real-time. Addressing challenges faced by E-Commerce companies, they used user ratings data in a sparse matrix to predict missing ratings, boosting recommendation accuracy. Their system architecture harnessed Apache Spark for efficient large dataset processing and Hadoop clusters for data management. Evaluation metrics like Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and Mean User Gain (MUG) were applied. Their Apache Spark-based prototype demonstrated quicker execution compared to non-distributed systems. Conclusively, their approach significantly influenced our system-level challenges, propelling the development of an adept and efficient recommendation system tailored for the dynamic E-Commerce sector.

IV. METHODOLOGY

This section details the methodological approach employed in the development of our e-commerce recommendation system, including data acquisition, pre-processing, algorithm selection, and model evaluation techniques.

A. *Detailed Methodology Description:*

Our e-commerce system architecture, as illustrated in the Use Case Style Diagram in Figure 1, is adept at collecting and processing user interaction data, including ratings and reviews, from various sources like website logs and user feedback mechanisms. After initial gathering (Step 0), the raw data is preprocessed for accuracy using Python libraries such as pandas (Step 1) and then stored in a MySQL database for detailed analysis. In the e-commerce industry, harnessing real-time transaction information is crucial. Our system architecture is designed to pass this information to streaming clustering algorithms like k-means or collaborative filtering methods such as ALS. This enables us to dynamically blend structured transaction data with unstructured data sources like customer comments or product reviews.

This integration is a key strategy in our approach, allowing us to continuously refine and adapt our recommendations based on new trends and user interactions. The system utilizes advanced collaborative filtering algorithms, including KNN with Means, SVD, and ALS, for generating personalized recommendations, while PySpark manages big data scalability challenges. The use of sophisticated SQL queries facilitates the efficient merging of these diverse data types, enhancing the accuracy and relevance of the recommendations. The entire analysis process, supported by Python libraries such as SciPy.stats, Matplotlib, and Seaborn, culminates in a user-friendly presentation on a Streamlit dashboard. Throughout, our system's architecture maintains a focus on availability and partition tolerance, ensuring it delivers real-time, personalized recommendations even in demanding network conditions, thereby improving user engagement and satisfaction on the e-commerce platform

B. *Design Goals: -*

Our database design for the e-commerce recommendation engine focuses on key goals to enhance functionality and user experience:

In our product recommender system for e-commerce, we've incorporated the principles of the CAP Theorem by prioritizing availability and partition tolerance. This ensures the system remains responsive and operational in real-time, even during network disruptions. Although this means occasionally sacrificing strong consistency, it's a deliberate decision to enhance the overall user experience, aligning with the high-paced nature of online shopping.

- Real-Time Processing: Aligning with the rapid pace of e-commerce transactions to ensure timely data handling and response.
- Availability: Prioritizing high availability to ensure quick and consistent responses from the recommendation engine, enhancing user interaction.
- Partition Tolerance: Designing for resilience against network issues or failures, maintaining continuous operation.
- Consistency vs. Availability: Balancing between strong consistency and high availability, accepting minor data staleness in certain scenarios.
- Scalability: Preparing for the increasing volume of e-commerce data to maintain efficiency as the system grows.
- Integration of Data Types: Merging structured transaction data with unstructured data like customer reviews, enriching the dataset for more precise recommendations.
- Efficient Data Processing and Analysis: Utilizing tools such as Spark/Flink, Kafka, and

MySQL for robust data analysis and insight extraction.

These goals guide our database design to support the Advanced Recommendation Engine, ensuring it's well-equipped to meet the dynamic demands of the e-commerce sector.

V. SYSTEM ARCHITECTURE

This section presents the system architecture, outlining the major components, their interactions, and the data flow between them.

A. *System Architecture Overview:*

Our system architecture, as illustrated in the Use Case Style Diagram in Figure 1, efficiently collects and processes user interaction data from an e-commerce platform. This data includes ratings and reviews acquired through website logs, user profiles, and feedback mechanisms. Initially, the raw data undergoes preprocessing using Python libraries like pandas for accuracy (Step 1), and is then stored in a MySQL database. Here, we implement real-time transaction processing, integrating structured data with unstructured data sources like customer comments or product reviews. This integration is essential for adapting recommendations to evolving trends and user preferences.

The system employs advanced collaborative filtering algorithms such as KNN with Means, SVD, and ALS for personalized recommendations, with PySpark ensuring scalability with big data. Our database system includes sophisticated SQL queries that facilitate the merging of these diverse data sources, using methods such as sort-merge-join for efficient data integration. SQL queries are utilized for data merging, enhancing the accuracy and relevance of our recommendations.

Data analysis and visualization are supported by Python libraries like SciPy.stats, Matplotlib, and Seaborn, with results displayed on a Streamlit dashboard to improve the shopping experience. The architecture prioritizes availability and partition tolerance, ensuring robust, real-time recommendations even in challenging network conditions, ultimately aiming to enhance user engagement and satisfaction on the e-commerce platform.

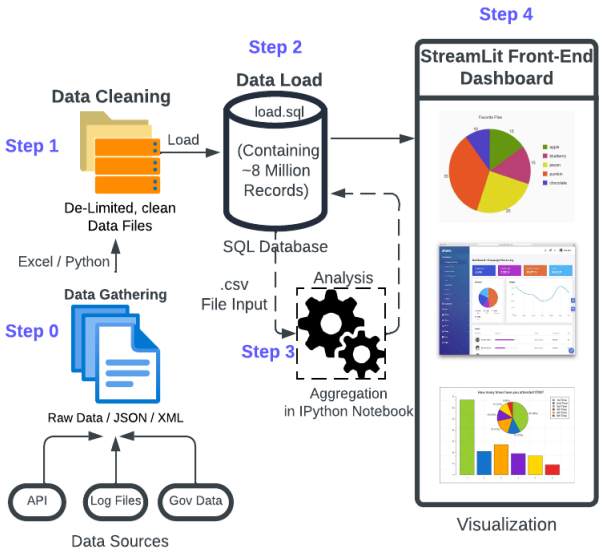

Figure 1: System Architecture Overview

The UML Deployment Diagram in Figure 2 illustrates the system architecture, where the client device interacts with a web server hosting the Streamlit application. This server communicates with an application server that runs the recommendation engine code. Both servers access a centralized database server, which contains the datasets and database schema, forming a multi-tiered structure facilitating user requests and data processing.

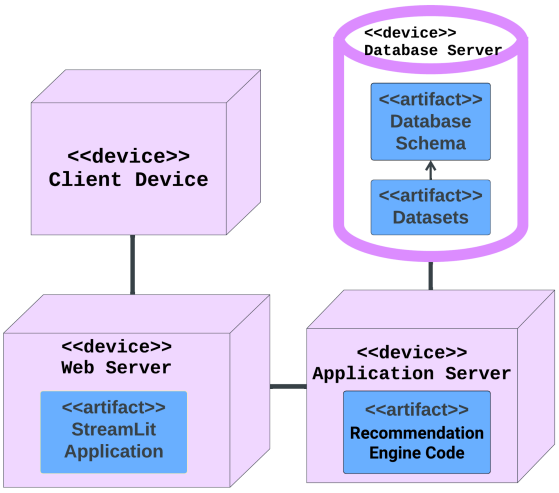

Figure 2: UML Deployment Diagram

B. *Component Interaction:*

In the system's architecture, class StreamlitInterface serves as the primary user interaction layer, orchestrating the flow between user inputs and the visualization of data. It leverages the “DataVisualization" class to render insightful plots and charts, which are crucial for analyzing the performance of the recommendation models. StreamlitInterface Class then calls upon the popularity_based_recommender_model & collab_filtering_based_recommender_model Classes to generate product recommendations based on user data. These recommender classes utilize collaborative filtering and popularity-based algorithms to process and predict user preferences. They depend on the DataVisualization class to visually summarize the results, enabling a cohesive workflow from data processing to user presentation. The interplay between these components ensures a seamless, dynamic, and user-centric recommendation experience within the E-Commerce platform dataset.

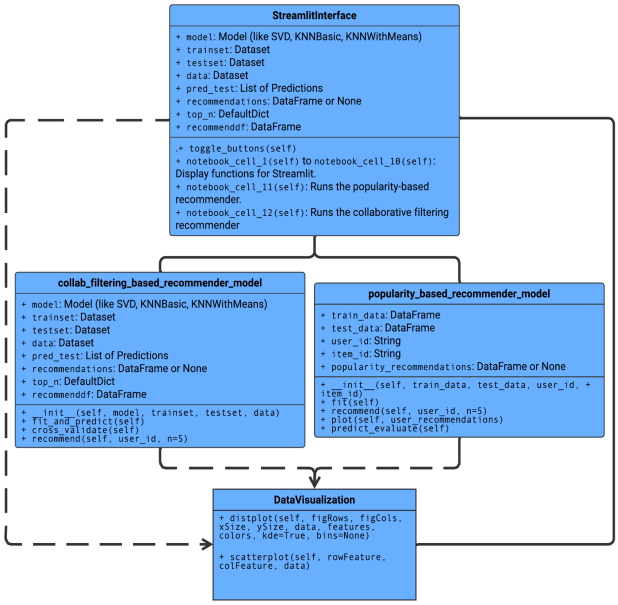

Figure 3: UML Class Diagram

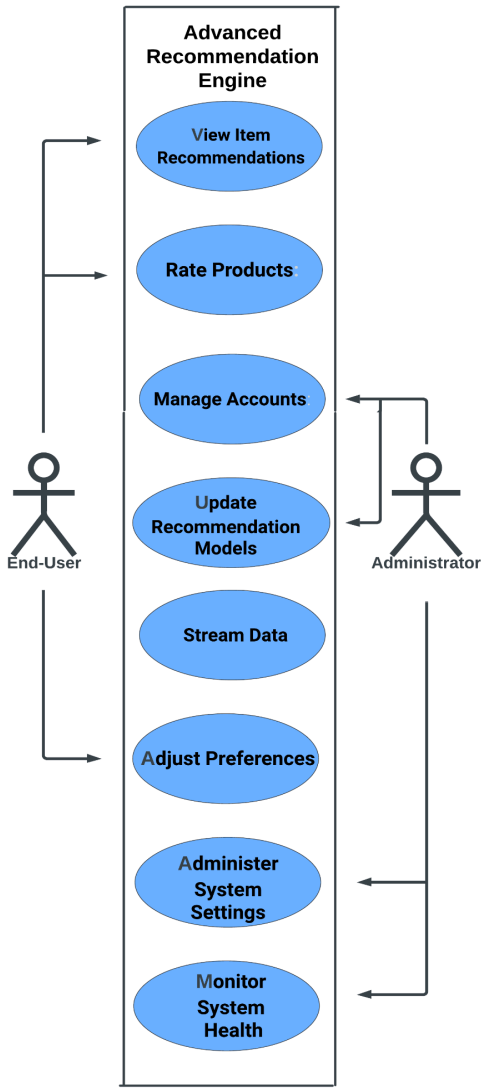

Figure 4: UML Use Case Style Diagram

The UML Use Case Diagram in Figure 2 showcases the interaction between end-users and administrators with the Advanced Recommendation Engine. End-users can view item recommendations, rate products, manage accounts, adjust preferences, and stream data. Administrators can update recommendation models, administer system settings, and monitor system health, indicating a comprehensive system that caters to various user needs and administrative functions.

VI. IMPLEMENTATION DETAILS

This section details the technical implementation of our Advanced Recommendation System, outlining the chosen programming language, frameworks, and libraries utilized to construct a robust and scalable system.

In [7], Xiangpo Li introduced the user-based collaborative filtering algorithm, which relies on user behavior, especially item ratings, to measure similarities among users and generate recommendations. The algorithm involves stages of data acquisition, calculating user similarities, and formulating recommendations based on evaluations from 'nearest neighbors', determined through cosine similarity between users' rating vectors in a multidimensional item space. Li's approach utilized the Mean Absolute Error (MAE) metric for performance validation, where a lower MAE signifies higher accuracy in predicting user ratings. The research explored the impact of user demographics on the algorithm's performance, demonstrating its effectiveness across diverse user groups. Li's enhanced collaborative filtering algorithm was integrated into our Advanced Recommender system, enhancing its ability to predict user preferences accurately, address user demographics, and improve overall recommendation quality and personalization.

In this research, paramount emphasis was placed on implementing a stable and robust database system capable of efficiently handling multiple requests. The system utilizes a MySQL database, selected for its attributes in terms of availability and partition tolerance. To enhance query performance, the implementation incorporates the strategic use of composite indexes. Additionally, the use of temporary

tables is employed to streamline complex queries, thereby reducing computational overhead and improving overall efficiency. A major aspect of the database design involves the integration of Sort-Merge-Join algorithm. This methodology ensures the delivery of accurate and anticipated results, reinforcing the system's efficacy in handling diverse datasets. This comprehensive approach to database management underscores the research's commitment to optimizing data processing and retrieval, vital for achieving the objectives of this study.

In the system illustrated in Figure 3's Sequence Diagram, user interaction initiates the recommendation process through the streamlitUI, the graphical user interface. The userInputHandler component handles user preferences' acquisition and initial processing. These preferences are then conveyed to the dataManager, which coordinates data retrieval by interacting with the databaseConnector, ensuring relevant data retrieval. Subsequently, the popularityRecommender uses advanced algorithms to combine user-processed preferences with the retrieved data to generate personalized recommendations.

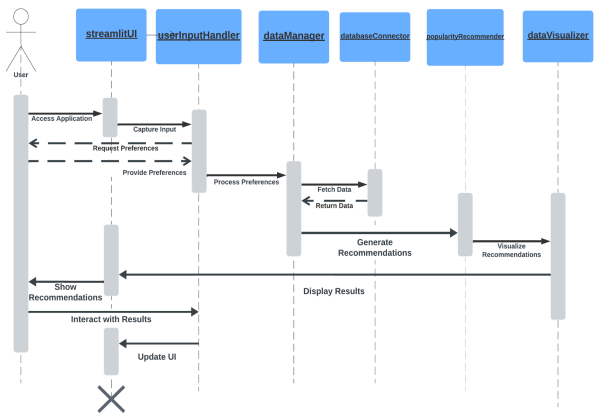

Figure 5: UML Sequence Diagram

These recommendations from popularityRecommender are then presented to the user through the dataVisualizer component, which employs advanced visualization techniques to represent the recommendations intuitively and interactively within the streamlitUI. Users can engage in a feedback loop by providing further input after reviewing the visualized recommendations, prompting the system to update and refine the recommendations dynamically. This iterative interaction highlights the system's adaptability and real-time responsiveness. The technical architecture emphasizes a modular design, facilitating system maintenance and scalability.

VII. EVALUATION

This section meticulously evaluates the performance of our E-Commerce recommendation system, utilizing metrics and benchmarks to assess its effectiveness, efficiency, and scalability.

A. *Evaluation Methodology:*

In [8], Weiwei Zhang tested the collaborative filtering algorithm's efficacy in a Spark cluster, focusing on its accuracy in reflecting user preferences. The algorithm was evaluated using Mean Absolute Error (MAE) across varying training and test set ratios. Results indicated enhanced precision with larger training sets, with the algorithm outperforming traditional methods. This validation confirmed their algorithm's robust performance and its practical applicability in providing accurate e-commerce recommendations.

In accordance with the System Architecture presented in the UML Communication Diagram in Figure 4, the Advanced Recommendation System's evaluation involves a series of rigorous tests to validate each component's functionality and seamless integration. Interface testing

ensures user interaction fidelity in the streamlitUI component, while unit testing focuses on the precision of the userInputHandler. Data integrity and performance tests are conducted on the dataManager and databaseConnector components, and the recommendationModel is assessed for algorithmic efficiency and accuracy. Additionally, the dataVisualizer's capacity to render intuitive and accurate recommendations is tested. System integration testing verifies data flow, followed by end-to-end and user acceptance testing to meet specifications and user needs. Performance metrics, including response time and system reliability, are monitored to ensure a robust user engagement platform.

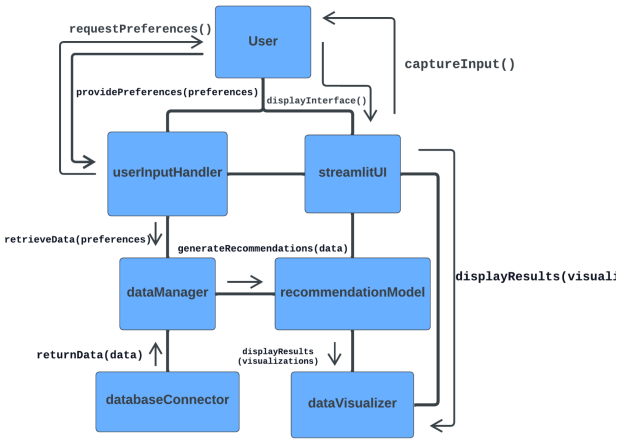

Figure 6: UML Communication Diagram

The evaluation methodology for our e-commerce recommendation system is meticulously designed to address the dynamic and vast nature of online shopping data, employing a sophisticated mix of data visualization, processing, and big data management tools.

B. *Significant Results*

The results of our evaluation indicate that the recommendation system performs exceptionally in the e-commerce domain. It showcases remarkable accuracy in predicting user preferences, as evidenced by the low RMSE values. This precision is critical in the context of recommendation systems and is indicative of the robustness of our methodologies and technological choices. Nicolas Hug, et al. in [9] underscore the importance of accuracy in such systems, particularly when dealing with diverse and large datasets. The integration of qualitative and quantitative data analysis through our technological stack, including the use of big data processing tools like pyspark and data analysis libraries such as pandas and Scipy.stats, enhances the system's capability to provide comprehensive and tailored recommendations.

This approach not only improves the user experience in e-commerce platforms but also exemplifies the effective application of advanced data processing and visualization tools in developing efficient recommendation systems.

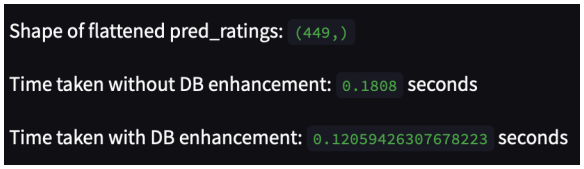

Figure 7: Time difference observed with and without database joins (UI)

VIII. SYSTEM DEMONSTRATION

In this section, we showcase the implementation and functionality of our user-centric e-commerce recommendation system with an intuitive interface. Unlike traditional methods that rely on datasets like Movielens, our approach integrates Product Rating and Product Review datasets, offering a more e-commerce-centric analysis. This integration provides a comprehensive view of user preferences by combining quantitative ratings with qualitative reviews. Our technology stack, including pandas, Scipy.stats, pyspark, SparkContext, ydata-profiling, scikit-learn, MySQL, and Streamlit, plays a crucial role in

efficiently managing the data's complexity and volume, ultimately optimizing the recommendation process.

In addition to accuracy, our system exhibits significant improvements in access times and efficiency. This enhancement is attributed to the strategic implementation of efficient join operations, specifically the sort-merge join technique within our database management practices. The sort-merge join, known for its efficiency in processing large-scale data, has been utilized that contributes significantly to the system's performance.

Furthermore, we employed composite indexes on SQL tables, which played a pivotal role in optimizing query execution times. By indexing multiple columns used frequently in queries, the system has been able to rapidly retrieve and process relevant data, thereby reducing the response time significantly. These indexes are particularly beneficial in accelerating the look-up processes, making the system more responsive and agile.

The combination of these database optimization strategies, along with the utilization of big data processing tools like PySpark and data analysis libraries such as Pandas and Scipy.stats, has significantly enhanced the system's capacity for providing tailored recommendations.

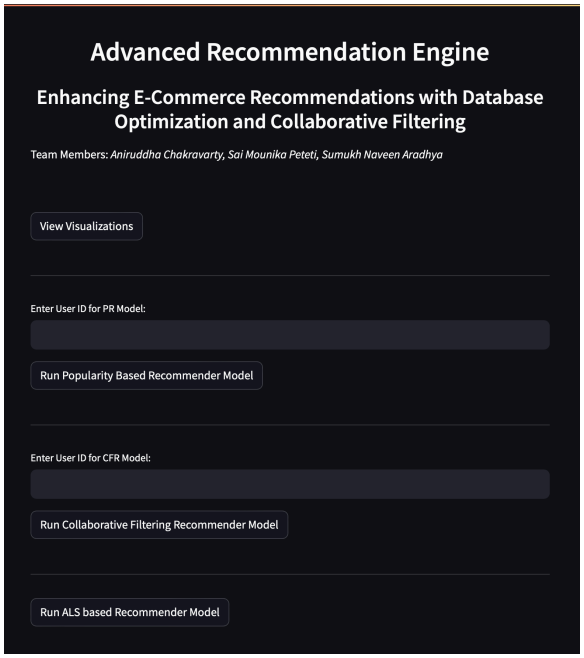

Figure 8: Front-End User Interface (UI)

```
mysql> LOAD DATA INFILE '/var/lib/mysql-files/ratings_Electronics.csv'
    -> INTO TABLE electronics
    -> FIELDS TERMINATED BY ','
    -> OPTIONALLY ENCLOSED BY '"'
    -> LINES TERMINATED BY '\n';


Query OK, 7824482 rows affected (37.62 sec)
Records: 7824482  Deleted: 0  Skipped: 0  Warnings: 0

mysql>
mysql>
mysql>
mysql>
mysql> select * from electronics limit 5;
+----------------+------------+--------+------------+
| userId         | productId  | Rating | timestamp  |
+----------------+------------+--------+------------+
| AKM1MP6P0OYPR  | 0132793040 |   5.00 | 1365811200 |
| A2CX7LUOHB2NDG | 0321732944 |   5.00 | 1341100800 |
| A2NWSAGRHCP8N5 | 0439886341 |   1.00 | 1367193600 |
| A2WNBOD3WNDNKT | 0439886341 |   3.00 | 1374451200 |
| A1GI0U4ZRJA8WN | 0439886341 |   1.00 | 1334707200 |
+----------------+------------+--------+------------+
5 rows in set (0.03 sec)

mysql> select count(*) from electronics;
+----------+
| count(*) |
+----------+
|  7824482 |
+----------+
1 row in set (1.62 sec)
```

Figure 9: SQL Querying

A. *Popularity Based Recommender Model:*

As seen below in Figure 5, the Popularity-Based Recommender Model serves as a fundamental component of the Advanced Recommender Engine, providing a reliable benchmark for trending products through its user-score aggregation and ranking system.

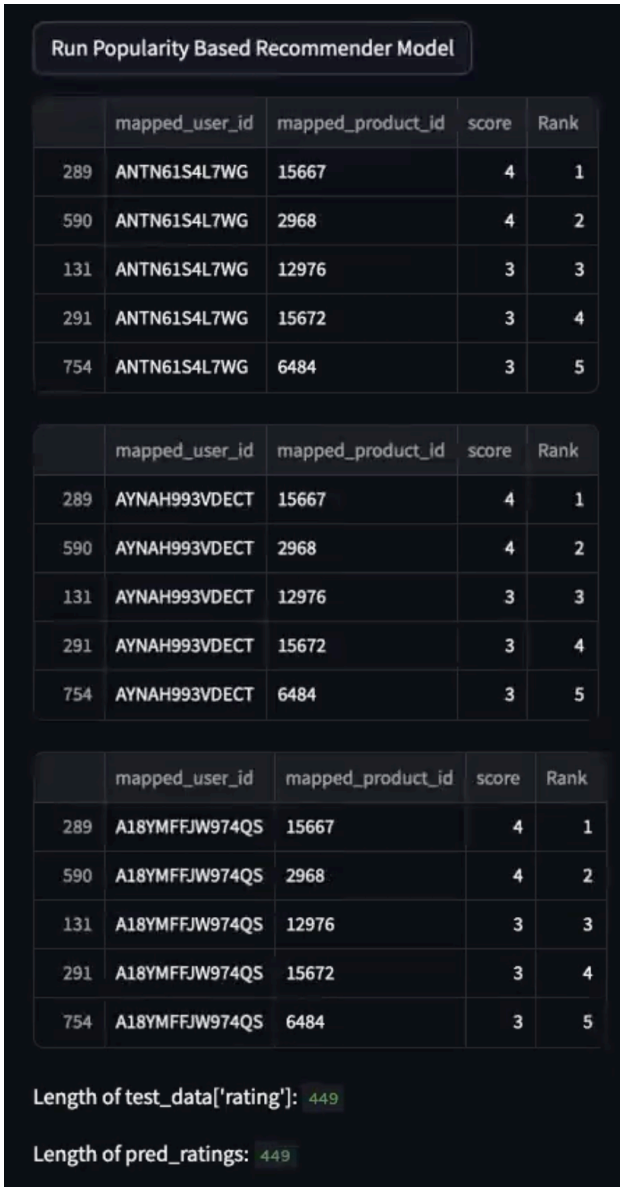

Figure 10: Popularity Based Recommender Model

B. *KNN with Means - Memory based Collaborative Filtering*

As depicted in Figure 6, the KNN with Means model, a memory-based collaborative filtering approach, is pivotal for the Advanced Recommender Engine. It enhances personalization by adjusting for user bias, refining recommendations to align closely with individual user preferences and historical ratings.

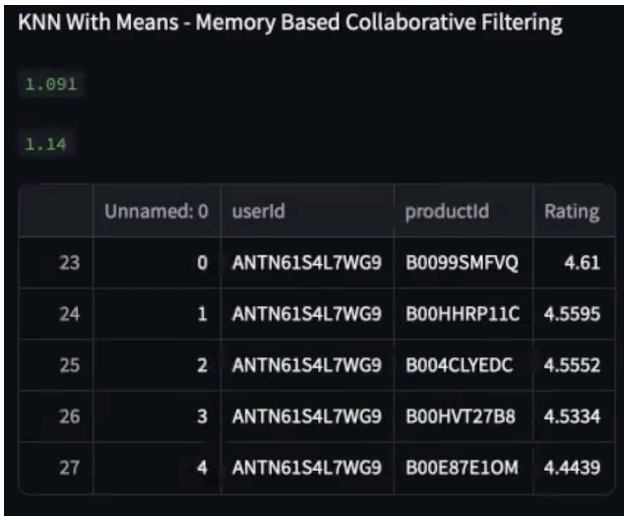

Figure 11: KNN with Means - Memory based Collaborative Filtering

C. SVD - Model Based Collaborative Filtering

As seen below in Figure 7, the SVD model-based collaborative filtering method is a cornerstone for the Advanced Recommender Engine, offering precise user-item interaction analysis for recommendation accuracy. Significantly, the integration of database (DB) enhancements yields a noticeable efficiency gain, with the time taken for computations reduced from 1.139 seconds without DB enhancement to 0.987 seconds with it, underscoring the model's effectiveness and the impact of optimization strategies.

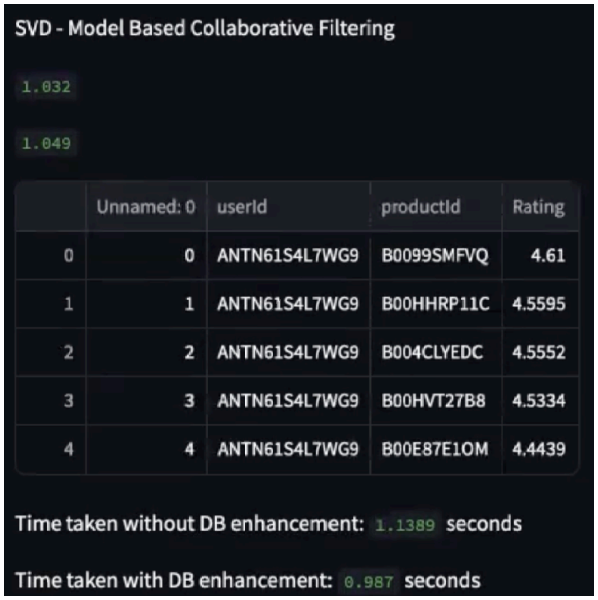

Figure 12: SVD - Model Based Collaborative Filtering

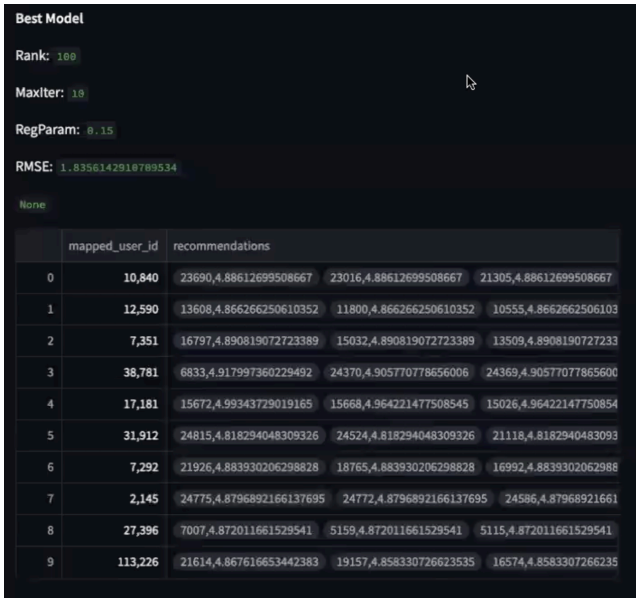

Figure 13: Alternating Least Squares (ALS) based Recommender Model

D. *Alternating Least Squares (ALS) based Recommender Model:*

As seen below in Figure 8, the Alternating Least Squares (ALS) based Recommender Model is integral to the Advanced Recommender Engine, adeptly handling sparse data to yield personalized recommendations. The model's effectiveness is quantified by a Root Mean Square Error (RMSE) of 1.835, with a defined rank of 100 and regularization parameter (RegParam) of 0.15, indicating a balanced approach between accuracy and overfitting, essential for maintaining the integrity of user-specific recommendation quality.

IX. DISCUSSION

This section delves into the implications of our e-commerce recommendation system's findings and assesses its contributions to the field, addressing potential limitations and future research directions.

A. *Results Analysis:*

The Results Analysis of the implemented Advanced Recommendation Engine, incorporating the components outlined in the Activity Diagram in Figure 9, indicates a profound impact on personalized shopping experiences. The system begins with "User provides preferences," a critical step where user-specific data is collected. This step reflects how tailored inputs are crucial in shaping the recommendation outcomes. In the Capture input in streamlitUI phase, the system demonstrates its user-friendly interface capabilities, which is pivotal in ensuring user engagement and accurate preference logging. As preferences are processed within the “Process input in userInputHandler," the system's ability to interpret and transform user data into actionable

insights is showcased. The "Retrieve data from database" component, involving the dataManager and databaseConnector, emphasizes the system's robustness in fetching relevant data, which is a cornerstone for generating reliable recommendations. The subsequent "Generate recommendations" phase, utilizing models like KNN with means, SVD, and ALS, underlines the system's versatility and analytical strength in creating personalized suggestions. "Visualize data," executed by the dataVisualizer, illustrates the system's capability to present recommendations in an engaging and comprehensible manner, enhancing user decision-making. The final step, "Display results to user," where results are presented back through streamlitUI, reinforces the system's effectiveness in delivering a personalized e-commerce experience. Collectively, these components—user preference collection, input processing, data retrieval, recommendation generation, and visualization—contribute to a recommendation system that not only accommodates large-scale and sparse datasets but also maintains the agility to provide individualized recommendations, solidifying the system's implications and effectiveness in the realm of e-commerce personalization.

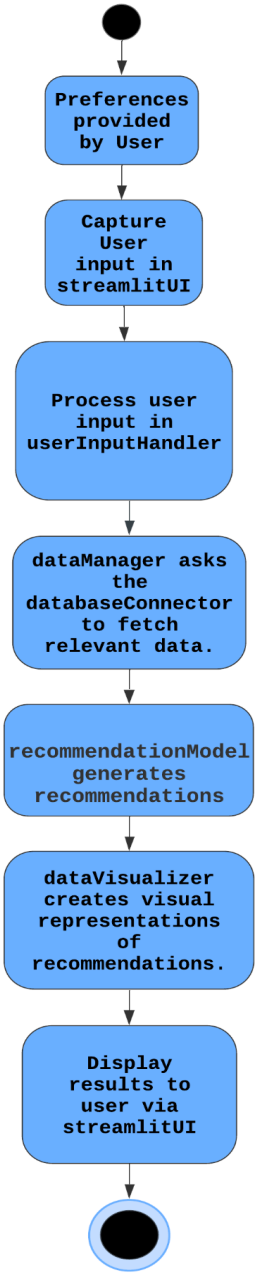

Figure 14 `: UML Activity Diagram

B. *Challenges and Limitations:*

In developing the recommendation system, we faced challenges inherent to collaborative filtering, such as capturing rapidly changing user preferences and the computational intensity of models like KNN with means, which also struggled with the cold-start issue. The popularity-based model, while effective for trending items, often lacked personalized

recommendations. Scalability, addressed by the ALS model, came with the risk of overfitting. Additionally, the vast and sparse nature of e-commerce datasets presented challenges in balancing recommendation diversity and algorithm efficiency. Adding to these complexities was the integration of structured and unstructured data, which required sophisticated data merging techniques and careful handling to ensure the integrity and relevance of the combined dataset, further emphasizing the need for ongoing system refinement and optimization.

X. LESSONS LEARNED & FUTURE SCOPE

This section reflects upon key lessons learned during the development of our e-commerce recommendation system and outlines promising avenues for future exploration and enhancement.

A. *Lessons Learned:* -

The development of our e-commerce recommendation system offered profound insights into various critical aspects: algorithmic complexity, data quality, user-centric design, scalability, adaptability, and collaborative approaches. The integration of complex algorithms required meticulous attention to efficiency to ensure real-time responsiveness. The significance of data quality became evident, as data cleaning, processing, and management were pivotal to the system's effectiveness. Our commitment to user-centric design principles led to significant usability enhancements through iterative development, placing a strong emphasis on user experience and feedback.

We addressed scalability concerns early, gaining valuable insights into their long-term impact on functionality and maintenance. Adaptability was identified as essential in the dynamic e-commerce environment, allowing our system to seamlessly integrate new features and adapt to evolving market needs. Additionally, the project highlighted the importance of collaboration. Working within a cross-functional team, we combined diverse expertise to create a more robust and comprehensive solution.

Importantly, this project made us realize that real-life data is often more complex than anticipated. This revelation underscored the criticality of sound database principles. Effective database design, organization, and management are crucial in handling the intricacies and scale of real-world data, further emphasizing the need for robust database systems to support complex e-commerce environments

B. *Future Scope*:

In the future, our emphasis will be on enhancing database systems to enrich e-commerce experiences. We plan to integrate real-time browsing patterns with historical purchase data, using advanced database management to personalize recommendations. Key efforts will include database optimization to reduce query latency for faster, more relevant data retrieval. This will involve techniques like effective indexing and optimized query design.

We will also explore advanced technologies like distributed databases and NoSQL to support complex data structures for machine learning models. Additionally, integrating Natural Language Processing for sentiment analysis from user reviews will further refine our recommendations, combining it with behavioral data for a nuanced shopping experience. These steps will enable us to efficiently handle diverse, large-scale data, setting the stage for advanced, personalized e-commerce solutions.

XI. CONCLUSION

Shortened conclusion:

In our project, we've harnessed a diverse tech stack to elevate the e-commerce shopping experience. Incorporating tools for data

visualization like matplotlib and seaborn, data processing via pandas and Scipy.stats, and big data handling with pyspark and SparkContext, our system integrates sophisticated algorithms such as Collaborative Filtering, KNN, SVD, and Popularity-Based Recommendation. Central to our architecture is a robust MySQL database, with a user-friendly interface created using Streamlit, Flask, and Tornado. We've structured our operations around two key data tables—one for product and purchase data, the other for review comments—merged efficiently using sort-merge join techniques for enhanced data analysis and recommendation accuracy. In line with the CAP Theorem, our system prioritizes availability and partition tolerance, ensuring resilient and scalable performance.

Additionally, our system's depth is augmented by sentiment analysis on review comments, melding this with user ratings to enrich the recommendation system. By combining both quantitative and qualitative data, we offer a more personalized shopping experience. Adhering to our project vision, we leverage real-time transaction data and streaming clustering algorithms like K-Means and ALS, integrating these with unstructured data sources such as customer comments and product reviews. This approach ensures our system's continuous evolution and adaptation to new trends, successfully enhancing the e-commerce experience while significantly improving user satisfaction and business efficiency, meeting our primary objectives.

XII. REFERENCES

[1] Zhao, Zhi-Dan, and Ming-Sheng Shang. "User-based collaborative-filtering recommendation algorithms on hadoop." In 2010 third international conference on knowledge discovery and data mining, pp. 478-481. IEEE, 2010.

[2] Zarzour, Hafed, Ziad Al-Sharif, Mahmoud Al-Ayyoub, and Yaser Jararweh. "A new collaborative filtering recommendation algorithm based on dimensionality reduction and clustering techniques." In 2018 9th international conference on information and communication systems (ICICS), pp. 102-106. IEEE, 2018.

[3] Xiaojun, Liu. "An improved clustering-based collaborative filtering recommendation algorithm." Cluster computing 20 (2017): 1281-1288.

[4] Liu, Hongjiao. "Implementation and Effectiveness Evaluation of Four Common Algorithms of Recommendation Systems-User Collaboration Filter, Item-based Collaborative Filtering, Matrix Factorization and Neural Collaborative Filtering." In *2022 International Conference on Cloud Computing, Big Data Applications and Software Engineering (CBASE)*, pp. 224-227. IEEE, 2022.

[5] Zhao, Xuesong. "A study on e-commerce recommender system based on big data." In 2019 IEEE 4th international conference on cloud computing and big data analysis (ICCCBDA), pp. 222-226. IEEE, 2019.

[6] Ghosh, Subasish, Nazmun Nahar, Mohammad Abdul Wahab, Munmun Biswas, Mohammad Shahadat Hossain, and Karl Andersson. "Recommendation system for e-commerce using alternating least squares (ALS) on apache spark." In *International Conference on Intelligent Computing & Optimization*, pp. 880-893. Cham: Springer International Publishing, 2020.

[7] Li, Xiangpo. "Research on the application of collaborative filtering algorithm in mobile e-commerce recommendation system." In *2021 IEEE Asia-Pacific Conference on Image Processing, Electronics and Computers (IPEC)*, pp. 924-926. IEEE, 2021.

[8] Zhang, Weiwei, Lingming Cao, and Yuanting Lu. "Personalized and Stable Recommendation Algorithm of E-commerce Commodity Information Based on Collaborative Filtering." In *2022 Global Reliability and Prognostics and Health Management (PHM-Yantai)*, pp. 1-6. IEEE, 2022.

[9] Hug, Nicolas. "Surprise: A Python library for recommender systems." *Journal of Open Source Software* 5, no. 52 (2020): 2174.

[10] S. Shekhar, Chang-Tien Lu, S. Chawla and S. Ravada, "*Efficient join-index-based spatial-join processing: a clustering approach,*" in IEEE Transactions on Knowledge and Data Engineering, vol. 14, no. 6, pp. 1400-1421, Nov.-Dec. 2002, doi: 10.1109/TKDE.2002.1047776