Volume 9, Issue 4, April – 2024 International Journal of Innovative Science and Research Technology
ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24APR1249

Recommender System in E-Commerce

Abhinav Sharma[1]; Preksha Agrawal[2]; Surendra Kumar Keshari[3]
Information Technology KIET Group of Institutions Delhi-NCR, Ghaziabad, India

Abstract:- In the realm of e-commerce, recommendation systems play a pivotal role in guiding users towards relevant products. However, existing systems often grapple with inefficiencies in handling large datasets and fail to deliver personalized recommendations tailored to individual preferences. Addressing these challenges, the study introduces an innovative approach leveraging graph databases to enhance the performance of e-commerce recommendation systems. Through comprehensive analysis, the study delves into four critical aspects: database comparison, user exploration frequency across product categories, the diversity of available category types, and user browsing history analysis. This investigation unveils Neo4j's superior efficiency over MySQL in managing extensive datasets, laying the groundwork for more robust recommendation engines. By scrutinizing user behaviour patterns, the recommender system predicts preferences with precision, promising a tailored and gratifying shopping experience for users. Moreover, with support for a diverse array of category types, users gain flexibility in exploring products based on varied criteria, addressing a crucial need in the market for personalized shopping experiences. Leveraging insights gleaned from user browsing history, the system delivers refined recommendations, poised to elevate user satisfaction and engagement within the competitive landscape of e-commerce. In conclusion, the study highlights the significance of recommendation systems in enhancing the e- commerce experience. By leveraging graph databases, particularly Neo4j, over traditional systems like MySQL, significant improvements in managing extensive datasets are demonstrated.

Keywords:- *E-commerce, Recommendation Systems, Neo4j, Personalized Recommendations, User Engagement, Data Analysis.*

I. INTRODUCTION

In the ever-evolving realm of online retail, recommendation systems play a central role in shaping user experiences and influencing purchasing behaviours. As e-commerce platforms strive to deliver tailored and compelling content, there exists a pressing need for novel solutions to enhance the efficiency and effectiveness of recommendation engines [1]. This research endeavours to tackle the inherent challenges faced by traditional recommendation systems, with a keen focus on enhancing user satisfaction and refining the precision of product suggestions.

This project aims to leverage advanced technology, particularly graph databases such as Neo4j, to augment the capabilities of these systems. Graph databases offer a unique approach to organizing and querying interconnected data, offering a promising avenue for developing more sophisticated and personalized recommendation algorithms [2].

At the core of this study lies the objective of conducting a comprehensive evaluation and comparison of Neo4j and MySQL databases within the context of an e-commerce recommender system. By scrutinizing their performance in handling query parameters and response times, the aim is to determine the most suitable database solution for managing extensive datasets, thereby ensuring timely and accurate information retrieval. Furthermore, study delves into specific metrics pivotal to the effectiveness of the discussed system, including the frequency of user visits to different grouping types, the diversity of available grouping types, and patterns in user traversal history [3]. Fig 1 discuss about how the recommendation works on abstract level. This Fig 1. tell more about factors that are used in discussed recommendation system.

Observations seeks to bridge the gap between traditional recommendation systems and the evolving needs of online shoppers. Through this endeavour, the aspiration to enhance the overall shopping experience, ultimately driving user engagement and satisfaction in the ever-expanding landscape of e-commerce.

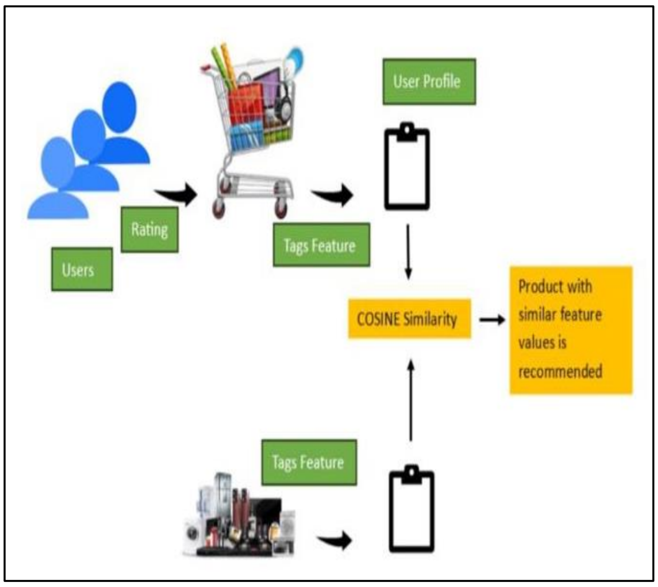

Fig 1: Conceptual Diagram for Recommendation System

II. LITERATURE REVIEW

The section discusses the proposed recommender engine encompasses a comprehensive understanding of existing recommendation systems, particularly collaborative filtering methods and the challenges they face. Collaborative filtering (CF) methods, including memory-based and model-based approaches, have been extensively studied and utilized in various online platforms to provide personalized recommendations based on past user- item interactions.

A. Collaborative Filtering (CF) Recommendation Methods

The recommendation systems field encompasses Collaborative Filtering (CF), Content-Based Filtering (CBF), and hybrid methods. CF strategies focus on computing similarities between users and items using distance measures, with variations between user-based and item-based approaches.

Recommender systems commonly leverage two CF techniques: memory-based and model-based methods as mentioned by Breese et al. [4]

Clustering techniques are instrumental in model- based CF approaches, discussed by Sarwar et al. [5]. Recent studies explore model-based clustering approaches to enhance efficiency.

Traditionally, data collection in business was primarily for specific purposes. However, the realization of data's value as a substantial asset has transformed its treatment. For instance, the IBM and Twitter partnership on data analytics aimed at selling analytical information to corporate clients, enabling real-time decision-making [6].

Big data represents a recent evolution in decision support data management [7], significantly impacting various business domains, including Customer Relationship Management (CRM), Enterprise Resource Planning (ERP), and Supply Chain Management (SCM).

The variety aspect involves handling various data types, including structured and unstructured data, from diverse channels like audio, video, images, and location data (Google Maps, web pages, text) [8].

Customer Relationship Management or CRM serves as a pivotal tool and strategy for managing customer interactions, encompassing sales, marketing, and customer service activities [9].

The emergence of Web 2.0 has revolutionized collaboration platforms beyond traditional email and information retrieval [10]. Social networks embedded within Web 2.0 have transformed organizations by enabling peer-to-peer collaboration and real-time communication [11]. However, this rapid growth in Web 2.0 has necessitated a shift in CRM techniques, leading to the advent of Social CRM (CRM 2.0).

B. Enhanced User Profiling through Tagging Behaviour Analysis

To improve recommendation accuracy, De Gemmis et al. [12] emphasized the importance of user tagging behaviour in constructing more accurate user profiles. Gedikli et al. [13] expanded on this notion, proposing methodologies to exploit context- specific tag preferences within recommendation processes.

Some attempts by Du et al.[14] aimed to reduce misinterpretation of user preferences derived solely from tags.

C. Challenges and Innovations in Addressing Data Sparsity through Model-Based Approaches

Matrix Factorization (MF) methods, a prominent model-based technique, aim to predict ratings by decomposing the user-item matrix into latent feature vectors. SVD++ and probabilistic matrix decomposition models by Koren et al. [15] initially struggled with data sparsity, limiting their ability to extract comprehensive user-item feature information.

Subsequently, Enrich et al. [16] and Bao et al. [17] proposed advanced algorithms integrating tag information to improve rating predictions and alleviate overfitting issues in matrix decomposition.

D. Leveraging Temporal and Tagging Data Integration for Enhanced Recommendations

Recent studies, such as Wang et al. [18] in TV program recommendations, have explored temporal information to enhance group recommendations. This study aims to delve deeper into integrating and exploiting additional tagging and temporal data sources, aiming to better understand user tag preferences and deliver more precise recommendations.

E. Utilizing Graph Databases in recommendations

The proposed recommender engine represents a paradigm shift in recommendation systems, leveraging graph databases to overcome existing limitations. Unlike conventional methods, it amalgamates graph databases' strengths to address various loopholes prevalent in the current landscape.

One primary shortcoming of memory-based collaborative filtering methods is their susceptibility to sparsity issues, especially in large-scale applications. Leveraging graph structures allows for meaningful representation of user-item interactions, addressing the sparsity problem.

The engine utilizes graph-based models to enhance recommendation accuracy and scalability.

Furthermore, the engine's use of graph databases enables the hybridization of recommendation techniques within a unified structure, offering a more robust recommendation mechanism to users.

Its architecture allows for real-time data processing, leveraging the speed of graph databases for adaptive learning, ensuring recommendations remain relevant and up-to-date.

III. METHODOLOGY

The methodology encompasses defining user and tracker nodes, collecting tracker node values, creating and updating relationships based on user interactions, storing grouping information, generating binary arrays, calculating cosine similarity, sorting products, and presenting top recommendations. By leveraging collaborative filtering concepts and frequency-based considerations, the system provides personalized and accurate recommendations tailored to individual user preferences, ensuring comprehensive ranking and presentation of top recommendations.

- **Define User and Tracker Nodes**: The creation of user and tracker nodes is a foundational step, initializing the graph to represent users and tracker attributes.
- **Collect Tracker Node Values**: The function Collect Tracker Node Values (products, tracker Nodes) gathers values associated with tracker nodes for each product, facilitating a detailed understanding of user preferences based on tracking attributes.
- **Store Grouping Information**: The part involves storing grouping information based on frequency considerations. The function Store Grouping Information (user Nodes, tracker Nodes, interactions) identifies and stores the most frequently visited grouping types and values, contributing to a richer understanding of user preferences.
- **Generate Binary Array**: The part involves generating binary arrays based on the frequency of grouping values. The function Generate Binary Array(products, grouping Info) emphasizes the importance of frequently visited values in user preferences by representing them as binary arrays.
- **Calculate Cosine Similarity**: The formula involves applying collaborative filtering concepts in the cosine similarity calculation.

The cosine similarity formula is used in the **Calculate Cosine Similarity** function to measure the similarity between the searched product and other products. Here's the cosine similarity formula in Equation (1):

Given two vectors **A** and **B**, the cosine similarity ($\cos(\theta)$) is calculated as:

$$\cos\theta = \frac{A.B}{||A||.||B||} \qquad (1)$$

In the context of the recommender system methodology:

$$\frac{\sum_{i-1}^{n} searched\ Product[i].product[i]}{\sqrt{\sum_{i-1}^{n}(searchedProduct[i])^2}.\sqrt{\sum_{i-1}^{n}(product[i])^2}} \qquad (2)$$

In Equation (2), searched Product[i] and product[i] are the binary values (0 or 1) for the ith grouping type in the binary arrays of the searched product and another product, respectively.

-n is the total number of grouping types.

This formula computes the cosine similarity between the binary arrays of the searched product and another product, providing a measure of how similar their grouping type preferences are. The resulting similarity score is used in sorting and ranking products to present personalized recommendations.

- **Present Top Recommendations**: The formula involves presenting the top recommendations to the user based on the sorted products. The function Present Top Recommendations (sorted Products, top N) incorporates collaborative filtering and frequency-based considerations to deliver personalized and accurate recommendations.

The Flowchart discuss the methodology and flow of control. Fig 2. discusses about the flow of algorithm and data in recommender system.

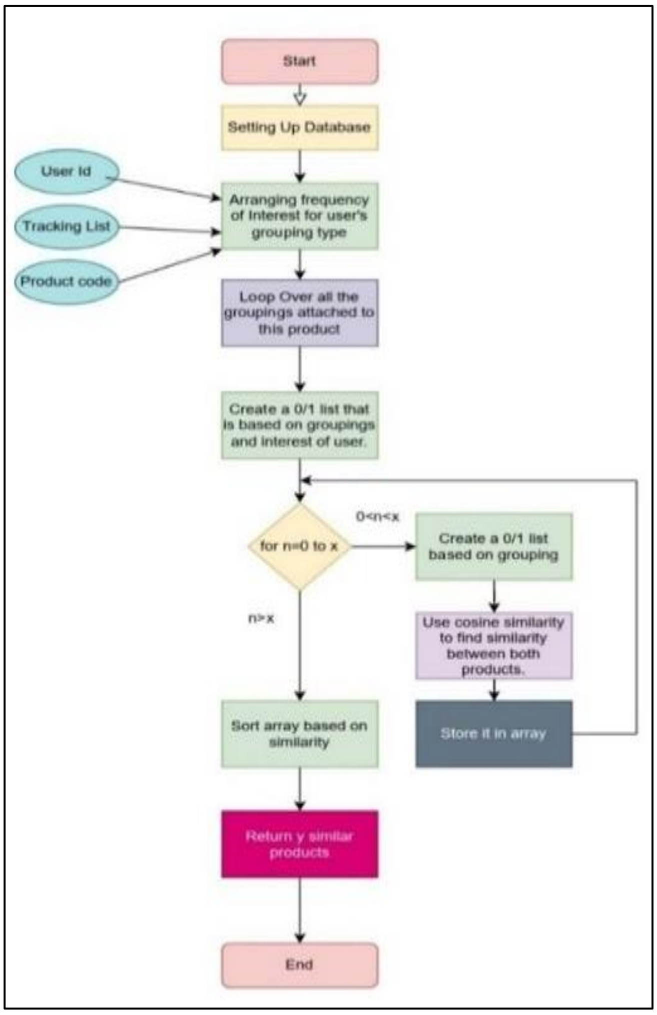
Start
Setting Up Database
User Id
Tracking List
Product code
Arranging frequency of Interest for user's grouping type
Loop Over all the groupings attached to this product
Create a 0/1 list that is based on groupings and interest of user.
for n=0 to x
0<n<x
Create a 0/1 list based on grouping
Use cosine similarity to find similarity between both products.
Store it in array
n>x
Sort array based on similarity
Return y similar products
End

Fig 2: Flowchart of Recommender System

IV. PROPOSED PRODUCT RECOMMENDATION USING GRAPH DATABASE

To generate accurate recommendations, the products and their grouping data should be represented in graph database in Neo4j. After the data, has been represented in graph database, customers can get product recommendations using cypher queries. The Fig 3. and Fig 4. shows outputs of cypher queries.

A. Dataset Description

In the context of their e-commerce product recommendation project, data organization occurs within a graph database framework. The dataset construction follows a sequence of distinct steps, each characterized by specific queries and relational structures.

- Grouping Data: For the creation of groupings based on the store's grouping data, utilize API calls to receive the necessary information.
- Product Creation: Following the establishment of groupings, products are created in alignment with the grouping and grouping type data associated with each product.
- Cosine Similarity: Cosine similarity plays a pivotal role in the recommendation system. The similarity between products is calculated using feature arrays (val and val1). Here's a working query demonstrating the computation of cosine similarity:
- WITH gds.similarity.cosine(val, val1) as res, p2 as p2 RETURN p2.code, p2.name, p2.thumbnail, res ORDER BY res DESC LIMIT 10.

B. Dataset Structure Establishment

Project begins by creating a structured database for housing product-related data, employing a graph database model. Initiate this process by establishing a main node in database with predefined categories, including "Apparel," "Footwear," "Personal Care," and "Sporting Goods". Subsequently, Forge relationships between these categories and the main database to signify their inclusion within the overarching dataset structure.

```
MATCH (a:Category),(b:Database) WHERE NOT EXISTS((b)-[:Include]->(a)) CREATE (b)-[:Include]->(a) RETURN a,b
```

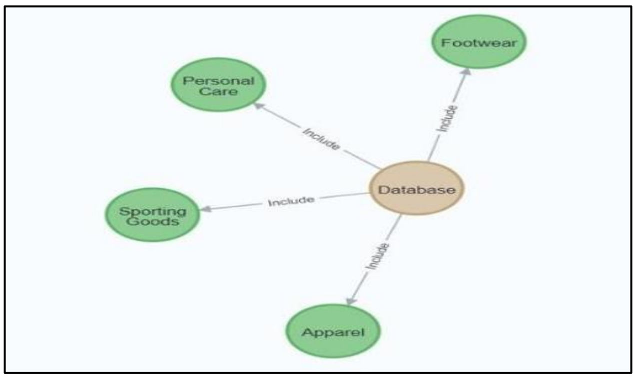

Fig 3: Output for Relationship between Categories

C. Attribute and Feature Integration

With the foundational structure in place, the project delves into the creation of attributes and their associations with respective categories. For instance, create an attribute named "Brand" for the "Apparel" category:

```
CREATE (:Product{name:"Nike Air Max", id:101, brand:'Nike'})
MATCH (a:Product),(b:Value) WHERE a.id=101 AND a.brand=b.name CREATE (a)- [:Related{type:'Brand'}]->(b) RETURN b
```

D. Attribute Value Creation

The dataset incorporates attribute values that can be universally accessible across multiple categories and attributes. Establish relationships between attributes and their corresponding values, allowing attributes like "Brand" to be associated with specific values:

```
MATCH (a:Attribute),(b:Value) WHERE a.isAttributeOf="Footwear" AND a.name="Brand" AND b.name="Adidas" CREATE (a)-[:ItemValue]->(b) RETURN a,b
```

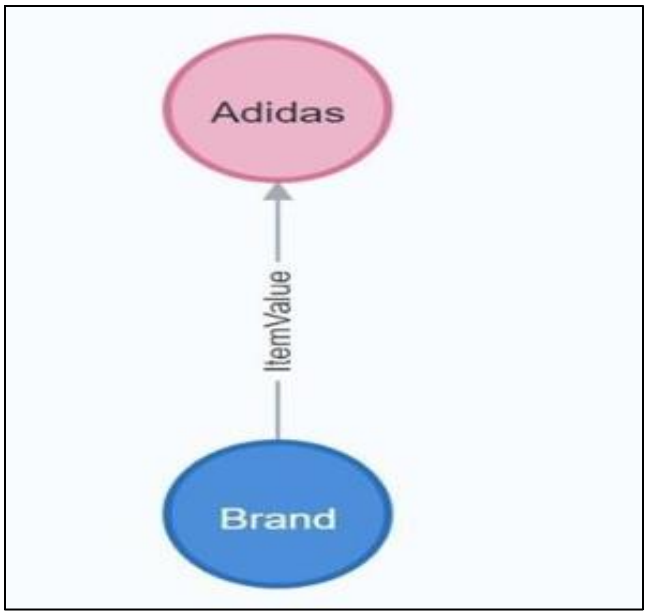

Fig 4. Output for Relationships between Attributes and their Values

E. Product Creation and Attribute Linkage:

To complete the dataset, product nodes are created and directly relate attributes like "Brand" to their corresponding values. For instance, create a product named "Nike Air Max" and establish its brand as "Nike".

```
CREATE (:Product{name:"Nike Air Max", id:101, brand:'Nike'})
MATCH (a:Product),(b:Value) WHERE a.id=101 AND a.brand=b.name CREATE (a)- [:Related{type:'Brand'}]->(b) RETURN b
```

F. *Attribute Value Count*

Finally, the dataset is leveraged to determine the count of attribute values linked to a specific category, relying on the product's relation type. For instance, the counting process encompasses values associated with the "Brand" attribute for the product "Nike Air Max."

```
MATCH (a:Product)-[r:Related]->(b:Value)<-[v:ItemValue]-(c:Attribute)-[:ItemValue]->(f:Value)
WHERE a.id=101 AND b.name=a.brand AND c.name=r.type RETURN COUNT(f) + 1.
```

This comprehensive dataset construction approach facilitates the creation of a rich and interconnected database, paving the way for effective product recommendations within the e-commerce platform.

V. RESULT AND DISCUSSION

In this investigation, the research focuses on the comprehensive evaluation of the recommender system, investigating its performance across key parameters essential for efficient recommendation generation in e-commerce platforms. Specifically, examine and compare the system's behaviour concerning the database comparison, frequency of user visits, the diversity of grouping types available within the system, and the history of user traversal patterns to ascertain its ability to deliver tailored and effective recommendations.

A. *Database Efficiency and Comparison with traditional SQL Database*

The aim is to see which database, Neo4j or MySQL, worked better for e-commerce project. Same Questions were asked to both databases five times and averaged how long it took to get answers. The main goal is to find out which database could handle a lot of data and provide quick answers.

For e-commerce work, store important information in Neo4j, a special type of database. The study used it to ask questions and get recommendations for products. At the same time, store the same information in MySQL to compare how fast both databases worked.The results, shown in Table 1, highlight Neo4j's quicker performance compared to MySQL. Times are measured in milliseconds (ms).

Table 1: Comparison between Neo4j and MySQL

Metric	**Neo4j**	**MySQL**
Query Complexity (Out of 10)	4.5	7.2
Indexing Efficiency (Out of 10)	8.2	5.9
Response Time (ms)	12.1	18.5
Scalability	Excellent	Good

Neo4j's superior indexing efficiency over MySQL is deeply rooted in its native graph structure, purpose- built to excel in graph databases. Its support for native full-text indexing and composite indexes further enhances efficiency, offering a compelling advantage over MySQL in scenarios prioritizing relationships, adaptability, and tailored indexing strategies.

Neo4j outshines MySQL in query complexity due to its native graph database architecture. Neo4j's strength lies in traversing relationships, making it inherently suited for complex graph queries. In contrast, MySQL, may require complex joins and subqueries to achieve similar graph-based operations, resulting in increased query complexity.

The label-based indexing in Neo4j contributes significantly to its favorable query complexity. This indexing mechanism, designed for graph-oriented searches, aligns seamlessly with the structure of the graph database, simplifying the formulation of queries based on node labels. MySQL, with its traditional B-tree indexes, may face challenges in achieving the same level of simplicity in queries involving relationships, potentially leading to higher query complexity.

Moreover, Neo4j's Cypher query language supports graph traversal operations with a natural syntax, making it easier to express and understand queries that involve navigating through nodes and relationships. In essence, Neo4j's native support for graph queries and its optimized query language position it as a more effective solution for scenarios where query complexity is a critical factor.

B. *Frequency of Time Each Grouping Type is Visited:*

In the context of online platforms, e-commerce, and search engines, the term "frequency" refers to the number of times a product or grouping has been searched. This metric is used in analytics and business intelligence to understand user behaviour.

The frequency of a product or grouping is measured by how often users enter specific search queries related to that product or category. This information can be gathered through search logs and analytics tools provided by search engines or e-commerce platforms. High search frequency for a particular product or grouping may indicate strong user interest and demand. Products with high search frequency are often considered popular, and businesses may use this information to optimise their inventory, marketing campaigns, and overall product offerings.

Users can leverage information about search frequency to plan targeted marketing campaigns and promotions. Promoting products that have a high search frequency can be a strategic way to capture user attention and drive sales.

For a user, who often visits website and searches for various products. So system may save frequency for a grouping type here "Brand" as represented in Table II.

Table 2: Working of Frequency Metric in System

GROUPING VALUE	FREQUENCY
Casio	1
Titan	3
FastTrack	0
Esprit	1
Timex	0

So it can be easily predicted that Titan is most visited by user

C. Grouping Types in the System:

Grouping type refers to the manner in which elements or entities are categorised, organised, or classified based on shared characteristics, properties, or criteria. grouping types play a pivotal role in organising and managing information efficiently. There are several types of grouping. The commonly used grouping type are color, gender, age, brand, etc.

Each grouping type serves a unique purpose, providing a flexible framework for structuring data and enabling efficient retrieval and analysis, tailored to the specific needs of the problem domain. The choice of grouping type is a critical design decision, influencing the ease of data manipulation, search, and overall system performance.

In Table III. some grouping types are shown with some groupings, these grouping types will define how accurately the recommendation is working.

Table 3: Working of Grouping Metric in System

GROUPING TYPE	GROUPING VALUE
Brand	Casio
.	Titan
.	FastTrack
Age Group	Kids-unisex
.	Adults-men
.	Kids-girls

D. History of Traversal

In Table IV each user's search history will be maintained and also what grouping values they landed to. This will help us to give specific recommendations about user's choice.

Table 4: Working of History Metric in System

SER ID	PRODUCT VISITED	GROUPING VALUES
U1	#U764	Casio
	.	Adults-men
	.	Black
	.	Accessories
U1	#U892	Kids-unisex
	.	Titan
	.	FastTrack
	.	Accessories

The combination of high visit frequency in specific groupings, a diverse range of grouping types, and a deep understanding of user traversal patterns positions the recommender system to deliver tailored and effective recommendations. Leveraging these crucial parameters enhances the system's ability to meet user preferences and improve overall user satisfaction.

VI. CONCLUSION

The shortcomings of conventional recommendation systems, including slow execution times and limited personalization. By leveraging innovative database technologies such as Neo4j, they've demonstrated significant improvements in handling large datasets, addressing a critical bottleneck in system efficiency. Moreover, insights gleaned from analysing user visit frequency, grouping type diversity, and historical traversal patterns have enabled the delivery of tailored suggestions, meeting individual preferences and enhancing overall satisfaction in online shopping experiences.

Looking ahead, the integration of efficient databases and insightful metric analysis offers promising avenues for further advancements in personalized recommendation systems. This underscores the importance of aligning technology with user needs, paving the way for a future where product recommendations resonate more authentically with individual preferences. As the pursuit of innovation continues, the potential for enhancing online shopping experiences remains substantial, offering exciting avenues for future exploration and development in this domain.

REFERENCES

[1]. Charles Ntumba, Samuel Aguayo, Kamau Maina "Revolutionizing Retail: A Mini Review of E-commerce Evolution" (2023).
[2]. Devi Sunuwar; Monika Singh " Comparative Analysis of Relational and Graph Databases for Data Provenance: Performance, Queries, and Security Considerations " (2023).
[3]. Pablo Sánchez , Alejandro Bellogín " Point-of-Interest Recommender Systems Based on Location-Based Social Networks: A Survey from an Experimental Perspective" (2022).
[4]. Breese, John S., David Heckerman, and Carl Kadie. "Empirical analysis of predictive algorithms for collaborative filtering." arXiv preprint arXiv:1301.7363 (2013).
[5]. Sarwar, B. M., Karypis, G., Konstan, J., & Riedl, J. (2002). Recommender systems for large-scale e-commerce: Scalable neighborhood formation using clustering. In Paper presented at the Proceedings of the fifth international conference on com- puter and information technolog.
[6]. D. Clark, IBM and Twitter forge partnership in data analytics, Wall Street J. Retrieved April 15, 2016 from http://www.marketwatch.com/story/ibm-andtwitter-forge- partnership-on-data-analytics-2014-10-29.

[7]. H.J. Watson, O. Marjanovic, Big data: the fourth data management generation, Bus. Intelligence J. 18(3) (2013) 4–8 (Chicago)

[8]. E. Dumbill, Making sense of big data, Big Data 1 (1) (2013)1–2.

[9]. M. Anshari, Y. Alas, N. Yunus, N.I. Sabtu, M.H. Hamid, Social customer relationship management and student empowerment in online learning systems, Int. J. Electronic Customer Relat. Manage. 9(2/3) (2015) 104–121.

[10]. H. Hinchcliffe, The state of Web 2.0, 2006. Retrieved 12thMay,2012;from http://web2.socialcomputingmagazine.com/the_state_of_web_2 0.htm

[11]. P. Greenberg, CRM at the Speed of Light: Social CRM 2.0 Strategies, Tools, and Techniques for Engaging yOur Customers, fourth ed., McGraw-Hill Osborne Media, 2009.

[12]. De Gemmis, M., Lops, P., Semeraro, G., & Basile, P. (2008). Integrating tags in a semantic content-based recommender. In Proceedings of the 2008 ACM conference on Recommender systems (pp. 163–170). ACM.

[13]. Gedikli, F., & Jannach, D. (2013). Improving recommendation accuracy based on item-specific tag preferences. ACM Transactions on Intelligent Systems and Technology (TIST), 4, 11.

[14]. Liu, J., Wang, W., Chen, Z., Du, X., & Qi, Q. (2012). A

[15]. novel user-based collaborative filtering method by inferring tag ratings. ACM SIGAPP Applied Computing Review, 12 , 48–57

[16]. Koren, Y., Bell, R., & Volinsky, C. (2009). Matrix factorization techniques for recom- mender systems. Computer, 42(8), 30–37.

[17]. Enrich, M., Braunhofer, M., & Ricci, F. (2013). Cold-start management with cross-domain collaborative filtering and tags. In International Conference on Electronic Commerce and Web Technologies (pp. 101–112). Springer.

[18]. Bao, T., Ge, Y., Chen, E., Xiong, H., & Tian, J. (2012).

[19]. Collaborative filtering with user ratings and tags. In Proceedings of the 1st International Workshop on Context Discovery and Data Mining (p. 1). ACM.

[20]. Wang, Z., & He, L. (2016). User identification for enhancing ip-tv recommendation. Knowledge-Based Systems, 98, 68–75.