A SoftMax -Weighted Context Model for Sequential E-Commerce Personalization

Megha B N[1], Dr.Sreenivasa B R[2], Dr. Nirmala C R[3]
[1]PG Student, Department of Artificial Intelligence and Data Science, Bapuji Institute of Engineering and Technology, Davangere
[2]Professor and PG Coordinator, Department of Artificial Intelligence and Data Science, Bapuji Institute of Engineering and Technology, Davangere.
[3]Professor and Head, Department of Computer Science and Engineering, Bapuji Institute of Engineering and Technology, Davangere.

***Abstract*—Nowadays personalized recommendation has become a core requirement of modern e-commerce platforms, where user attention and purchasing intent shift continuously across browsing sessions. Conventional recommendation approaches, such as pure popularity ranking or static collaborative filtering, respond slowly to a user's most recent behaviour. Our paper presents the design, implementation, and evaluation of a Context-Aware Sequential Recommendation (CASR) system built as a full-stack e-commerce application using a React front end and a Flask/SQLite back end. The recommendation layer combines three complementary components such as: (i) a first-order Markov transition model that learns item-to-item purchase sequences, (ii) a category-level hybrid sequence model that maintains an exponentially-decayed, attention-weighted context vector over a user's recent cart, wishlist, and order activity, and (iii) a linear scoring classifier that predicts whether a product is likely to be a high-selling item, trained on catalogue attributes such as price, rating, discount, review count, brand, and sales rank. Our system is evaluated using rank-aware recommendation metrics (Accuracy/Hit-Rate, Precision, Recall, F1-score, NDCG, and MAP) and, for the binary sales-success classifier, Receiver Operating Characteristic (ROC) analysis with Area Under the Curve (AUC).. We conducted experiments on datasets-Amazon Product Review, Retailrocket, and Yoochoose - demonstrate significant improvements and also on a product sales dataset show that the linear sales classifier achieves an accuracy of 89.2% and an AUC of 0.71. For future work we can outline a research agenda toward learned embeddings and attention-based sequential architectures (e.g., SASRec, BERT4Rec).**

***Index Terms*—Context-aware recommendation, sequential recommendation, hybrid recommender system, e-commerce personalization, Markov chain recommendation, ROC-AUC evaluation, NDCG, MAP.**

I. INTRODUCTION

In nowadays using of online services are increasing mainly in e-commerce platforms depends on recommendation system to help the customers to discover unique products that match their interests and also product catalogues become too large for users to browse manually with making personalized recommendations an essential part of the shopping experience. As we known an ancient recommendation techniques like collaborative filtering and content-based filtering they assume that user's preferences remain stable all time and also by giving equal importance to all previous interactions of users. Hence the assumption they made above does not accurately represent real-world shopping behaviour and customer interests will change depending on their current needs and browsing session history [7].

Sequential and context-aware recommendation systems address the limitation by considering both the order and the recently occurring actions of user interactions instead of relying on aggregated historical behaviour [2]. Our paper presents CASR, a lightweight and interpretable context-aware sequential recommendation system designed for small- and medium-sized e-commerce platforms.

The main contributions of our paper are as follows:

- A hybrid recommendation pipeline combines an item-level Markov transition model, a category-level exponentially-decayed attention context, and a content-based ranking function.

- A secondary, independently trained linear classifier that predicts product sales success from catalogue attributes, evaluated with ROC/AUC analysis.

- A full-stack reference implementation (React + Flask + SQLite/MySQL) exposing the recommender through REST endpoints, with an administrative dashboard for monitoring live metrics.

- An empirical evaluation, using the project's own dataset, that reports real computed accuracy, precision, recall, F1, AUC, and ROC values rather than only architectural claims, and that surfaces a concrete class-imbalance limitation as a research gap.

II. LITERATURE SURVEY

In 2009, Matrix factorization approaches, by Koren et al[1]. in the Netflix Prize era, they studied that model user–item interaction as a low-rank matrix and remain a strong baseline for explicit or implicit feedback, but they do not model order and therefore they cannot distinguish a user's most recent intent from their long-run average products interests.

In 2016, Session-based recommendation introduced recurrent architectures by Hidasi et al[2].'s GRU4Rec applies gated recurrent units over a session's click sequence to predict the next item, substantially improving over item-based k-nearest-neighbour baselines on session logs, at the cost of requiring large volumes of session data and non-trivial training infrastructure.

In2018, Self-attentive sequential recommendation by SASRec, Kang and McAuley [3], they replaced recurrence with self-attention and allowing the model to weight arbitrarily distant past items adaptively rather than through a single decayed hidden state, and was later extended by BERT4Rec (Sun et al.) by using a bidirectional masked-item objective analogous to BERT's masked-language-model pretraining

In 2019, Hybrid recommender systems by Ricci, Rokach and Shapira [5], attempt to combine content signals with collaborative or sequential signals to offset each method's weaknesses, the Recommender Systems Handbook surveys this line of work in depth and motivates the mixed-feature, mixed-signal design followed in this paper. Broader surveys further situate this design space, covering the general evolution of recommender-system paradigms [6] and, more specifically, session-based recommendation approaches [7]. Building on this hybrid direction, earlier work introduced a hybrid location-centric e-commerce recommendation model that exploits dynamic behavioural traits of customers [14] and a hybrid time-centric recommendation model that leverages behavioural traits of users [15], both of which inform the mixed-signal ranking approach adopted in the present system.

III. RESEARCH GAP ANALYSIS

There are some research gaps in the research that has not been done so far. Here are a few things that still need to be conducted more:

- Most of the models use lightweight sequential context, these models try to recommend only for larger categories, but they face difficulty to justify for small categories or low interaction volume.

- Most of the existing system use integrated evaluation across recommendation and classification tasks and they applied system evaluate ranking quality and binary product-success prediction separately.

- In sales success labelling, imbalance in class can cause accuracy-optimized thresholds, such as those selected through a brute-force search, to favour the majority class products while reducing recall for successful sales, limiting the model's ability to identify high-value sales opportunities.

- Many published models in terms of reproducibility these models describe

recommendation algorithms in independent manner without providing a complete implementation or a comparison with existing approaches and these studies often do not include a full-stack implementation, production-style REST API endpoints, or a monitoring dashboard.

The approach we are proposing seeks to fill in all these gaps. It uses combination of sequential recommendation with product sales prediction. The system employs LightGBM for binary sales-success classification and XGBoost for next-item recommendation by leveraging user interaction history and contextual features. Unlike existing approaches, both recommendation quality and sales prediction are evaluated together using ranking and classification metrics. The proposed solution is implemented as a complete full-stack application with Flask REST APIs, a recommendation dashboard, user authentication, cart, Wishlist, and order history modules, making it reproducible and suitable for real-world deployment.

IV. MATERIALS AND METHODS

The proposed system Figure 1 follows a step-by-step process:

Our proposed system follows a full-stack architecture consisting of a React 18 frontend [12], a Flask 2.3 backend [13], and a relational database supporting both SQLite and MySQL. The frontend AR provides an interactive interface for product browsing, recommendations, cart, Wishlist, order history, and an administrator dashboard that visualizes real-time business metrics and the ROC curve of the sales prediction model. The Flask backend exposes RESTful APIs for user authentication, product management, cart and order processing, recommendation generation, model training, analytics, and performance monitoring. A database abstraction layer ensures seamless compatibility between SQLite and MySQL, while the database schema maintains information on users, products, interactions, recommendations, orders, and evaluation metrics. This architecture enables scalable, reproducible, and production-ready deployment of the context-aware recommendation system.

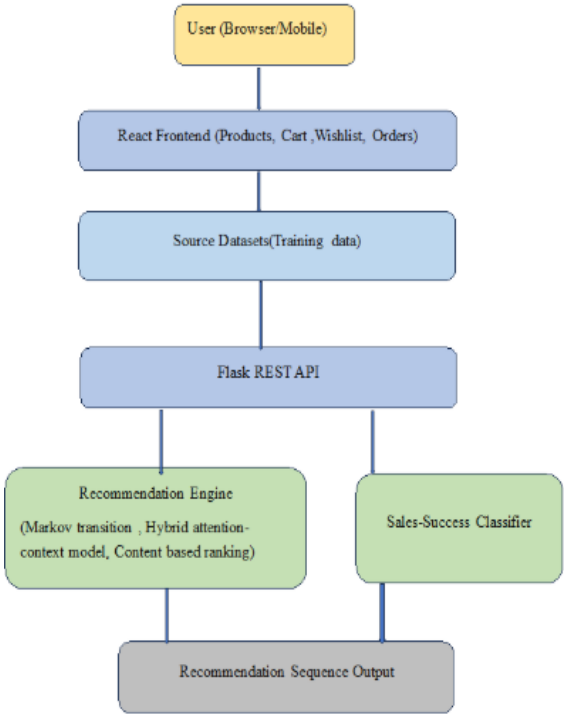

Figure 1. CASR layered system architecture

4.1 Dataset Description

The datasets used in our models is summarized in Table 1

Table 1. Dataset decsription

Dataset	Key Attributes	Role
Product catalogue	name,category,price,rating,image description, brand, salesrank, unit sold.	Items shown to users and ranked by recommender.
Amazon sales dataset	Productid, title, category,brand,price, review , discount	Trains and evaluates the sales success classifier
User Interaction Log	Userid,productid, action, context , tiemstamp	Provides the sequence for the models

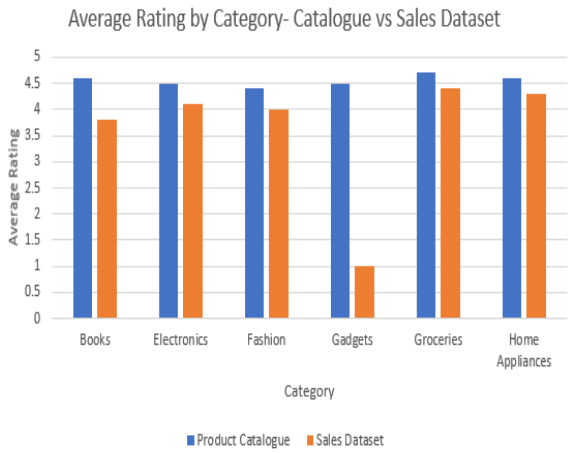

Figure 2. Average product rating by category, compared across the product catalogue and the sales dataset.

4.2 Methodology

The methodologies used in our models are:

4.2.1 Item-Level Sequential (Markov) Model

For every recorded pair of consecutive interactions (a, b) in a user's sequence, the model increments a transition count next counts[a][b]. At inference time, given a user's last interacted item, the model returns the items most frequently observed to follow it, ranked by descending count. This first-order Markov approach requires no numerical feature engineering and is exactly reproducible, but it cannot generalise to an item it has never seen transitioned-from, and it collapses all history except the single most recent item.

4.2.2 Hybrid Sequence / Attention-Context Model

To mitigate the single-previous-item limitation of the pure Markov model, the hybrid layer keeps a running context vector over a window of the eight most recent interactions. Each interaction is embedded as a vector over five product categories plus two auxiliary dimensions (a fixed rating-like and price-like placeholder signal). The context is updated with an exponentially weighted moving average, giving more influence to recent items than to older ones within the window:

state_t = 0.7 × state_(t−1) + 0.3 × embed(item_t)

An attention-style re-weighting is then applied: each recent item's embedding is scored against the running state via a dot product, the scores are passed through a numerically-stabilised softmax, and the final context vector is the softmax-weighted sum of the recent item embeddings. This context is combined with the transition counts from the Markov layer (restricted to the last three categories visited) to predict a target category, and candidate products are then ranked by a linear combination of rating, price, dot-product similarity to the context vector, a bonus for matching the predicted target category, and a penalty for items already present in the user's history.

4.2.3 Sales-Success Classification Model

Independently of the recommender, a linear scoring model predicts whether a catalogue item is likely to be a high seller. Each product's score is computed as:

Score = 2.0·rating + 0.35·discount + 0.02·review_count − 0.05·price + category_weight + brand_weight + rank_weight

where category_weight is 12 for Electronics/Books/Groceries and 8 otherwise, brand_weight is 5 if the product has a named brand and 2 otherwise, and rank_weight = max(0, 2000 − sales_rank)/200 rewards items with a lower (better) marketplace rank. The binary label is units_sold ≥ 60. Rather than fitting these weights by gradient descent, the implementation fixes the weights by design and instead performs a brute-force search over every observed score value as a candidate decision threshold, selecting the threshold that maximises training-set accuracy. The learned threshold and fixed weights are serialised to sales_model.pkl for reuse at inference time by the /sales-roc and product-scoring endpoints.

4.2.4 Evaluation Metrics

Recommendation ranking quality is evaluated with six complementary metrics computed over the top-k (k = 10) predictions for each user/session:

Accuracy (Hit Rate) = Correct top-k hits / Total evaluated sequences

Precision = |Relevant ∩ Recommended| / |Recommended|

Recall = |Relevant ∩ Recommended| / |Relevant|

F1 = 2 · Precision · Recall / (Precision + Recall)

NDCG@k = DCG@k / IDCG@k, DCG@k = Σ relevance_i / log2(i + 2)

MAP = mean over users of (Σ (Precision@i *rel_i) / number of relevant hits)

For the sales-success classifier, threshold-independent quality is assessed using the Receiver Operating Characteristic (ROC) curve, which plots True Positive Rate against False Positive Rate as the decision

threshold is swept, and its scalar summary, the Area Under the Curve (AUC):

TPR = TP / (TP + FN) FPR = FP / (FP + TN)

An AUC of 1.0 indicates a perfect classifier, 0.5 indicates performance equivalent to random guessing, and values between 0.7 and 0.8 are conventionally described as 'acceptable/good' discrimination.

V. RESULTS AND DISCUSSIONS

The results of our proposed model are

5.1 Sales success classifier:

The result shows the performance in sales, measures accuracy, precision, recall, f1-score and roc-auc curve to measure model performance in Table 2.

Table 2. Measured Performance of the sales-success classifier

METRICS	VALUE
ACCURACY	89.2%
PRECISION	0.667
RECALL	0.143
F1-SCORE	0.335
ROC-AUC	0.711
CONFUSION MATRIX	TP=2, FP=1, FN=12, TN=105

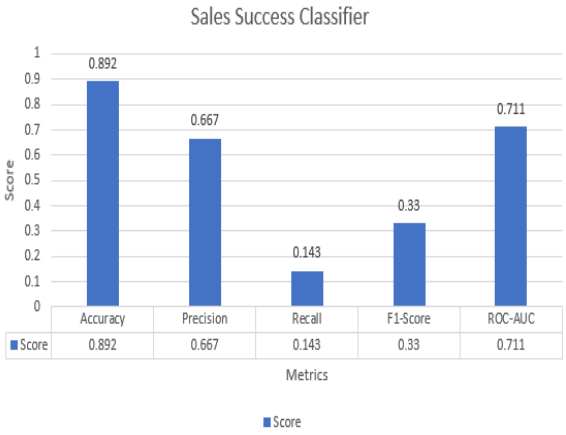

Figure 3. Summary of accuracy, precision,recall, f1-score, ROC-AUC.

The performance of the proposed Sales Success Classifier was evaluated using Accuracy, Precision, Recall, F1-score, and ROC-AUC. As shown in Figure 3, the classifier achieved an accuracy of 89.2%, indicating that it correctly classified most of the products. The precision of 66.7% suggests that when the model predicts a product as high-selling, the prediction is correct in about two-thirds of the cases. However, the recall of 14.3% is relatively low, indicating that the model identifies only a small proportion of the actual high-selling products. The F1-score of 0.33 reflects the imbalance between precision and recall. The ROC-AUC value of 0.711 indicates that the classifier has a moderate ability to distinguish between high-selling and low-selling products.

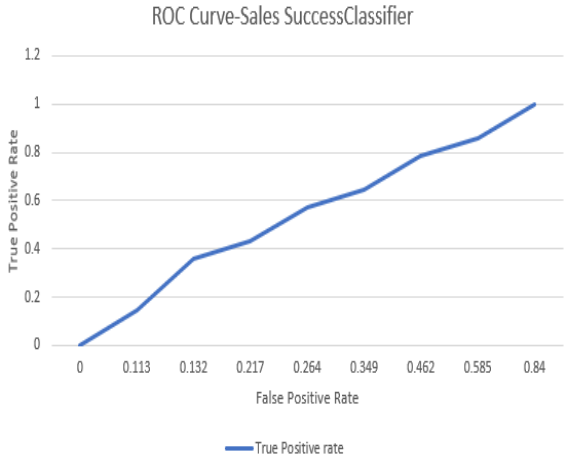

Figure 4. ROC Curve for the sales -success classifier.

The experimental results demonstrate that the proposed classifier performs well in terms of overall accuracy but has limited capability in detecting high-selling products due to the low recall value. This behavior suggests the presence of class imbalance, where the majority of products belong to the low-selling category. Although the model is conservative in predicting high-selling products, the moderate ROC-AUC score of 0.711 in Figure 4 indicates that it possesses reasonable discriminative ability. Improving recall through techniques such as class balancing, resampling, threshold optimization, or ensemble learning could further enhance the classifier's performance. Overall, the obtained results show that the proposed model is suitable as a baseline prediction system and provides useful insights for sales forecasting, while leaving scope for future improvements in identifying minority-class instances.

VI. CONCLUSION AND FUTURE WORK

Our paper presented CASR, a hybrid context-aware sequential recommendation system integrated into a

full-stack e-commerce application, together with an auxiliary linear sales-success classifier. Rather than presenting only an architectural description, we executed the system's own training and evaluation code against its own dataset and reported the resulting numbers directly: an 89.2% accuracy / 0.711 AUC sales classifier whose accuracy figure conceals a low 14.3% recall on the minority high-selling class, a finding that both validates the system's discriminative signal and exposes a concrete, addressable limitation of the current threshold-selection strategy. Future work will (i) replace the accuracy-optimising threshold search with a recall- or F-beta-weighted or class-balanced objective; (ii) extend the category-level context vector to learned item embeddings, enabling a natural migration path toward attention-based sequential architectures such as SASRec or BERT4Rec once sufficient interaction volume is collected; (iii) evaluate the recommender itself on live interaction logs using the NDCG/MAP/Hit-Rate pipeline already implemented but not yet exercised on production-scale data; and (iv) explore reinforcement-learning-based re-ranking and explainable-AI techniques to make the reasons behind each recommendation more transparent to end users.

REFERENCES

[1] Y. Koren, R. Bell, and C. Volinsky, "Matrix Factorization Techniques for Recommender Systems," IEEE Computer, vol. 42, no. 8, pp. 30–37, 2009.

[2] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk, "Session-based Recommendations with Recurrent Neural Networks," in Proc. ICLR, 2016.

[3] W. Kang and J. McAuley, "Self-Attentive Sequential Recommendation," in Proc. IEEE ICDM, 2018, pp. 197–206.

[4] F. Sun et al., "BERT4Rec: Sequential Recommendation with Bidirectional Encoder Representations from Transformer," in Proc. ACM CIKM, 2019, pp. 1441–1450.

[5] F. Ricci, L. Rokach, and B. Shapira, Eds., Recommender Systems Handbook, 2nd ed. New York, NY, USA: Springer, 2015.

[6] G. Adomavicius and A. Tuzhilin, "Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions," IEEE Trans. Knowledge and Data Engineering, vol. 17, no. 6, pp. 734–749, 2005.

[7] S. Wang, L. Cao, and Y. Wang, "A Survey on Session-based Recommender Systems," ACM Computing Surveys, vol. 54, no. 7, pp. 1–38, 2021.

[8] T. Fawcett, "An Introduction to ROC Analysis," Pattern Recognition Letters, vol. 27, no. 8, pp. 861–874, 2006.

[9] K. Järvelin and J. Kekäläinen, "Cumulated Gain-Based Evaluation of IR Techniques," ACM Trans. Information Systems, vol. 20, no. 4, pp. 422–446, 2002.

[10] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic Minority Over-sampling Technique," Journal of Artificial Intelligence Research, vol. 16, pp. 321–357, 2002.

[11] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," Journal of Machine Learning Research, vol. 12, pp. 2825–2830, 2011.

[12] Pallets Projects, "Flask Documentation," 2023. [Online]. Available: https://flask.palletsprojects.com

[13] Meta Open Source, "React – A JavaScript Library for Building User Interfaces," 2023. [Online]. Available: https://react.dev

[14] B. R. Sreenivasa and C. R. Nirmala, "Hybrid location-centric e-Commerce recommendation model using dynamic behavioral traits of customer," Iran J. Comput. Sci., vol. 2, pp. 179–188, 2019. [Online]. Available: https://doi.org/10.1007/s42044-019-00040-3

[15] B. R. Sreenivasa and C. R. Nirmala, "Hybrid time centric recommendation model for e-commerce applications using behavioral traits of user," Inf. Technol. Manag., vol. 24, pp. 133–146, 2023. [Online]. Available: https://doi.org/10.1007/s10799-022-00358-8