

Research Article

Enhancing Dropshipping Performance Through Hybrid Recommendation Engines

Mehmet Shala , Zana Shala Krasniqi* , Edmond Hajrizi 

Computer Science and Engineering, University for Business and Technology, Pristina, Kosovo.

*zs64619@ubt-uni.net

Abstract

Small dropshipping stores often present the same popular products to all customers rather than providing personalised recommendations. Personalisation, however, represents one of the few aspects of the shopping experience that such stores can directly control. This study presents a lightweight hybrid recommendation engine that combines content-based filtering, item-based collaborative filtering, and trend signals under the budget and infrastructure constraints of small businesses. The system was evaluated using the public Amazon Beauty dataset, comprising 1.35 million ratings from 883,753 users across 23,838 products, and compared with tuned ItemKNN, PureSVD, and Bayesian Personalized Ranking (BPR) baselines using a temporal train-test split and full-catalogue ranking. Introducing item-based collaborative filtering increased NDCG@10 from 0.0029 to 0.0418, while the best-performing hybrid configuration, with weights of 10% content, 70% collaborative, and 20% trend, achieved an NDCG@10 of 0.0425 ± 0.0013 and significantly outperformed the individual components. Although statistically reliable, the resulting improvement was small in practical terms, corresponding to approximately one additional successful recommendation per 862 recommendation lists. The relative advantage increased to 52% when only one-quarter of the training data was available but reversed for users with longer interaction histories. Content-based recommendations derived from product attributes provided the weakest signal, challenging the assumption that attribute similarity is particularly effective for beauty products. This finding was consistent across the metadata configurations evaluated, including the product-type field, which was the only fully reliable product attribute available in the dataset. The collaborative neighbourhood model achieved approximately twice the ranking performance of the latent-factor baselines while requiring approximately one-fiftieth of their training time. On commodity hardware, the deployed system generated recommendations with a median latency of 34 ms and required approximately 25 s for an offline rebuild. Six propositions with explicit boundary conditions are formulated to characterize the observed relationships between data sparsity, evidence distribution, and recommendation accuracy. The central proposition is that accuracy under sparse conditions depends more strongly on the alignment between model parameters and the locations of available evidence than on model sophistication alone. As the evaluation is limited to offline experiments, future work should assess whether these improvements translate into changes in customer engagement and conversion through live A/B testing.

Keywords: Recommendation Systems; Dropshipping; Collaborative Filtering; Content-Based Filtering; E-Commerce; Hybrid Recommendations; Low-Resource Deployment.

INTRODUCTION

Personalisation has become a standard expectation in online retail. McKinsey reports that 71% of consumers expect personalised interactions when shopping online [1], and other industry studies show that personalisation can influence purchasing decisions [2]. Recommendation systems are widely used by large online retailers, and both a peer-reviewed account of a production system [3] and industry reporting [4] associate them with improved conversion rates and higher average order value. At the scale of major platforms, even small improvements can generate substantial financial benefits.

However, small dropshipping businesses face challenges when trying to adopt similar technologies. Unlike traditional retailers, dropshipping stores do not manage their own inventory. Instead, customer orders are passed to suppliers, who ship products directly to buyers. This model reduces the cost of starting an online store, but it also creates several limitations. Product information is usually collected from multiple supplier feeds, which can vary in quality and consistency. Products may frequently enter or leave the catalogue, and retailers often have limited knowledge about the products they sell. In addition, these businesses operate with technology budgets orders of magnitude smaller than those of major platforms, typically without dedicated infrastructure or engineering staff. This makes large-scale recommendation platforms, such as those used by major e-commerce companies, difficult to adopt. This paper investigates whether a recommendation system that performs better than a simple best-seller approach can be developed within these practical constraints. We design, implement, and evaluate a lightweight recommendation engine specifically for this setting. The system combines content-based filtering, item-based collaborative filtering, and trend-based scoring through a weighted hybrid approach. It is implemented as a full-stack web application using Python/Flask, React, and MySQL, and evaluated offline using more than 1.3 million real product ratings.

Research Gap and Contributions

Most recommender-system research focuses on settings that are different from small-scale dropshipping in two key ways. First, existing studies often assume access to large amounts of user interaction data, such as the data available to major e-commerce platforms where item-to-item collaborative filtering was first widely adopted [5, 6]. Other studies focus on models that require substantial computational resources, including neural networks [7, 8], graph-based approaches [9, 10], and sequential transformer models [11, 12].

Second, although hybrid recommendation methods have been widely studied since Burke's survey [13] and continue to receive attention in e-commerce research [14], they are usually evaluated using standard datasets without considering deployment costs or resource limitations. Reproducibility studies have also shown that carefully tuned neighbourhood and linear models can perform as well as, or better than, more complex neural approaches on common benchmarks [15, 16]. However, these findings have not been directly connected to the practical question of which recommendation approach is suitable for small online retailers.

This creates three research gaps:

- G1: Density asymmetry is not captured by aggregate sparsity alone. Sparsity is typically reported as a single figure for the entire interaction matrix, and models are compared at that single operating point. The two margins of the matrix can nevertheless differ by orders of magnitude, with every recommendation model allocating its parameters to one or the other. Which margin carries the evidence, and the density at which each paradigm ceases to be viable, are rarely measured. Our setting makes this asymmetry extreme: 99.994% of the matrix is empty, yet the median user contributes a single interaction while the average product carries 57. Small-business deployments are where this matters most, because they combine this asymmetry with infrastructure budgets that rule out the models typically proposed for sparse data.
- G2: Lack of dropshipping-specific analysis. Recommendation research rarely considers conditions that are common in dropshipping, such as inconsistent supplier metadata, frequent catalogue changes, and the absence of first-party product quality signals. The business model also makes the asymmetry in G1 structural rather than incidental, since a store can add supplier catalogues far faster than it can add customers. Existing academic work on dropshipping mainly focuses on the business model itself rather than recommendation systems [17].
- G3: The value of combining signals under information scarcity is assumed rather than measured. In attribute-rich categories such as beauty, the relative contribution of product attributes and user behaviour is often taken for granted. Hybrid weights are likewise reported as study findings rather than as functions of how much evidence each component has to work with. As a result, it remains unclear whether the benefit of combining signals grows or shrinks as a store accumulates data.

To address these gaps, this paper makes five contributions:

- C1: A reproducible evaluation framework. We introduce an offline evaluation protocol using a public beauty-product dataset containing 1.35 million ratings. The evaluation includes per-user temporal splitting, full-catalogue ranking, NDCG, MAP, and hit-ratio metrics, together with beyond-accuracy measures, five repeated seeded runs, and paired non-parametric statistical tests.
- C2: A controlled comparison of recommendation approaches. We compare content-based filtering, popularity-based ranking, ItemKNN, PureSVD, and BPR under extreme sparsity. After validation-based hyperparameter tuning, the item-neighbourhood model achieves the highest-ranking accuracy among the five approaches and has the lowest training cost. These results are based on one dataset and one product category, so we do not generalise them to other methods or domains.
- C3: Analysis of hybrid model components. We perform component ablation and a 66-point weight-sensitivity analysis of the hybrid model. It has been measured the

impact of a common shortcut used in lightweight implementations, where the collaborative score is replaced with a fixed value instead of being calculated from item similarities. Replacing it with item-based collaborative filtering improves NDCG@10 by fourteen times. The best-performing weight combination is collaborative-dominant (10/70/20), while performance decreases consistently as more weight is given to the content component.

- C4: A deployable and reproducible implementation. It has been provided a complete implementation with a median online recommendation latency of 34 ms on commodity hardware and an offline model rebuild time of approximately 25 seconds.
- C5: A propositional account of the regime. We generalise the measurements into six propositions, each stated with the mechanism we believe produces it, the boundary conditions under which we expect it to hold, and the observation that would falsify it. They concern the alignment between where a model places its parameters and where the data places its evidence, the ordering of paradigm degradation thresholds, the declining marginal utility of combining signals, and the ceiling that metadata resolution places on any content component.

LITERATURE REVIEW

Collaborative Filtering

Collaborative filtering (CF) is based on a simple idea: if users agreed in the past, they would probably agree again in the future. This idea first appeared in Goldberg et al.'s Tapestry system [18] and later developed into two main approaches. User-based CF looks for users with similar rating histories, while item-based CF focuses on items that are liked by the same users [5]. The item-based approach became more useful at scale and is used in systems like Amazon's "customers who bought X also bought Y" [6], mainly because relationships between items are more stable and can be worked out in advance. Its main strengths and weaknesses are already well known [19, 20]. CF can suggest items that are not obviously related, which can be useful, but it has some clear problems. It does not work well for new users or new items (the cold-start problem), struggles when the data is sparse, and depends on storing user activity. Matrix factorisation [21] later changed CF by modelling users and items using hidden factors, and methods like Bayesian Personalised Ranking [22] made it work better with implicit feedback, which is common in e-commerce.

Content-Based Filtering

Content-based filtering recommends items that are similar to ones a user has already interacted with, using item features instead of other users' behaviour [23]. In cases where there is a lot of text, TF-IDF with cosine similarity is still commonly used because it is simple, cheap, and easy to understand [24]. These methods work well for new items and can explain why something was recommended. However, they often end up recommending very similar items again and again [25], which can limit variety. Their

performance also depends on the quality of the metadata, which can be inconsistent when it comes from different suppliers.

Trend-Based Recommendations and Hybrid Systems

Trend-based methods focus on what is popular rather than what is similar or personalised. Instead of asking what a user might like based on past behaviour, they look at what is currently being bought or rated highly. These methods do not need user history, so they are often used in cold-start situations [26]. In practice, many small online stores rely on simple best-seller lists. Hybrid systems combine different recommendation methods so that one can make up for the weaknesses of another. Burke [13] organised this area into categories like weighted, switching, mixed, and cascade approaches, and this categorisation is still widely used. Reviews of hybrid systems in e-commerce [14] show that weighted combinations are the most common, mainly because they are simple to use and adjust. However, choosing the right weights is usually done by trial and error, and this study finds that the best choice depends on how strong each signal is.

Modern Directions

In the last decade, recommendation systems have moved beyond these traditional approaches. Neural collaborative filtering replaces the simple inner product used in matrix factorisation with learned interaction functions [7], and deep learning is now used widely in this area [8, 27]. Graph neural networks model the user-item graph directly; LightGCN [9] showed that even simpler versions of these models can work well, and surveys describe many different variations [10]. Sequential recommendation models look at the order of user actions, including methods like SASRec [11], BERT4Rec [12], and transformer-based systems such as Transformers4Rec [28], with more survey papers appearing in this area [29]. Self-supervised learning tries to deal with limited labelled data using contrastive methods [30], and large language models have introduced a new way of generating recommendations without task-specific training [31].

However, these methods are not always practical in this context. They usually require large amounts of data, detailed user interactions, and significant computing power, which are not available in dropshipping settings. Also, some studies suggest that complex models do not always perform better in practice. The study in [15] showed that many neural recommenders can be matched by well-tuned simpler methods, and authors in [16] found that properly tuned dot-product models can outperform more complex ones. Whether this is still true in very sparse, low-budget environments is something this study aims to test.

Dropshipping and E-Commerce Analytics

There is not much academic research specifically on dropshipping. One recent study looks at its ecosystem and highlights how easy it is to enter [17], but recommendation systems in this area have not been studied in detail. Related areas provide more information. Research in database systems shows the importance of efficient indexing and avoiding repeated calculations in large e-commerce systems [32]. Work on text normalisation deals with combining product data from different suppliers [33]. More

generally, the business-analytics literature and industry forecasting associate effective use of data with competitive advantage [34-36]. Cold-start problems, which are especially important for low-traffic stores, have also been widely studied [37], and there are reviews focused on specific areas like fashion and beauty [38]. However, there is still not much evidence on which recommendation methods are actually worth using in dropshipping, especially considering their computational cost. This study aims to address that gap.

PROBLEM STATEMENT AND HYPOTHESES

A typical small dropshipping store relies on just one recommendation method, most often a simple best-seller list, sometimes filtered by category. There are a few practical reasons for this. First, the cold-start problem is very strong, since most visitors are new and there is little prior data to use [26]. Second, the data is extremely sparse. In the catalogue used in this study, 99.994% of the user-item matrix is empty, and the median user has interacted with only one product, which means there is almost no usable signal for personalisation. Third, real dropshipping catalogues are inconsistent, a condition the public dataset used here does not capture. The same product can appear multiple times under different names or categories because it comes from different suppliers, which makes straightforward content matching unreliable.

Based on this setting, the study evaluates three hypotheses.

- H1: A weighted hybrid that combines content-based, collaborative, and popularity signals performs better than any single method in terms of ranking accuracy in a sparse dropshipping catalogue.
- H2: In a category with strong product attributes, such as beauty, content-based recommendations are more accurate than simple popularity-based ones.
- H3: When data is extremely sparse, a lightweight neighbourhood-based CF model can achieve similar ranking performance to latent-factor models while using much less computational power.

These hypotheses are operational: each is decided by a measurement on this dataset and each is given an explicit verdict in the Results. They are not, however, the theoretical claim of the study. The measurements that decide them also reveal a more general regularity: namely, that the side of the interaction matrix on which evidence accumulates, rather than the sophistication of the model consuming it, determines which paradigm performs better and by how much. The Discussion states this regularity as six propositions (P1-P6), each with its boundary conditions. The hypotheses are therefore best read as the instruments through which the propositions were derived, and the propositions as what the study offers beyond the dataset on which they were derived.

Figure 1 summarises the framework tested in this study. It is not just a processing diagram: each arrow represents a causal claim that the experiments can support or refute. The first claim is that the amount of evidence available on each side of the interaction matrix, rather than the sophistication of the model using it, determines which signal

dominates. Sparsity is therefore treated as having two separate margins, user-side and item-side, which the Results examine separately. The second is that the value of a component in a weighted hybrid depends on the strength of the components alongside it, so mixing weights are a property of a given dataset rather than a transferable constant. The third is that the metadata a small retailer actually has limits what any content-based component can contribute, regardless of the similarity function used. The measured outcomes are relevance, diversity, novelty, and coverage, along with the deployment costs of latency, memory and rebuild time. Effects on business outcomes are not tested here and are left to future live experiments.

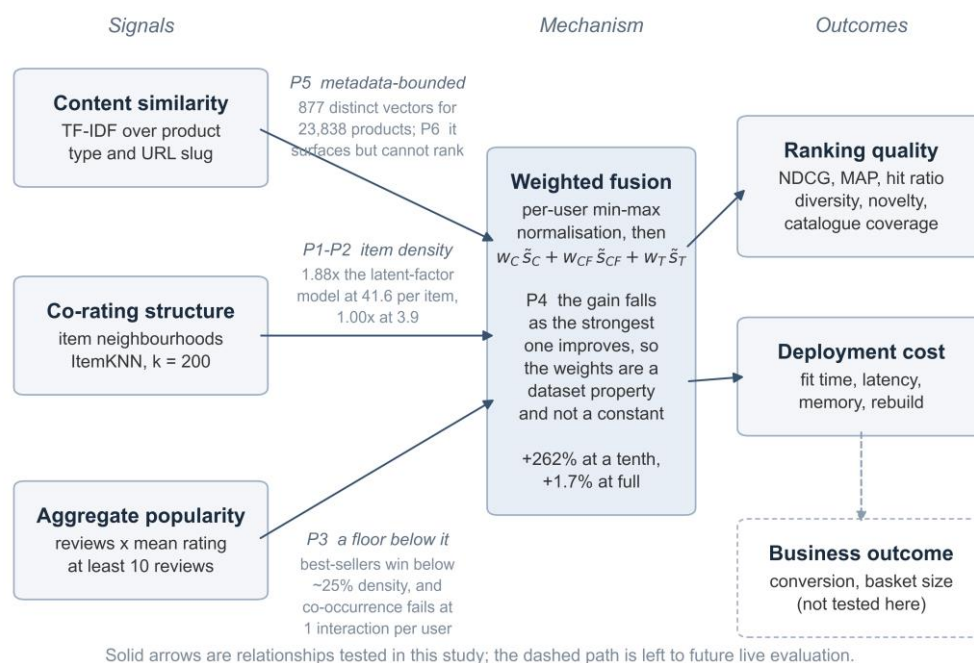


Figure 1. Conceptual Framework

Each input path is labelled with the proposition that governs it (P1-P3 and P5, stated in the Discussion), the condition that determines its strength, and the measurement establishing it. P4 governs the fusion step and is stated in the fusion box; P6 qualifies the content path and appears with P5. Solid paths are tested in this study; the dashed path is left for future live evaluation.

METHODOLOGY

Dataset

Our experiments use the public “Amazon Beauty Products” dataset available on Kaggle [39], which is similar in size and category focus to a typical dropshipping beauty catalogue. The dataset contains 1,348,246 ratings (with no duplicate user–item pairs) from 883,753 users across 23,838 products, identified by ASIN, and covering 22 product categories. Nail Polish (28%) and Shower Gel (15%) make up the largest shares. Ratings are given on a 1–5

scale and are skewed towards higher values, with 77% being either 4 or 5. The data spans a long period, from 1999 to 2014. Sparsity is the property that constrains which methods can work here: around 99.994% of the user–item matrix has no entries, the median user has only one rating, and just 28,040 users (3.2%) have five or more ratings. In this study, sparsity is not treated as a limitation but as the main focus, since it reflects the conditions typically faced by a new dropshipping store (gap G1). Table 1 provides a summary of the dataset and how it is split.

Table 1. Dataset and Split Statistics

Statistic	Value
Ratings	1,348,246
Users/Products/Product types	883,753/23,838/22
Sparsity	99.994%
Positive threshold q	rating ≥ 4.0
Training interactions (all users)	1,289,576
Held-out test interactions	58,670 (45,279 positives)
Evaluation users (≥ 5 interactions, ≥ 2 train positives, ≥ 1 test positive)	23,952

Pre-processing

The pipeline checks for missing identifiers and duplicate user–item pairs; the published dataset contains neither, so all 1,348,246 records are kept. Product categories are taken directly from the product-type field, which provides 22 top-level categories. As a result, no manual annotation is needed, and inter-annotator agreement is not relevant. The product text used for content-based filtering is the product type concatenated with the text of the product URL slug. Large-scale attribute normalisation methods, such as those used in e-commerce datasets [33], are also unnecessary here because the data already includes a single, consistently formatted category field.

An audit of that URL field shows that it contains much less product information than its name suggests. No product identifier in the dataset matches the ASIN in its own URL, and the ASINs that do appear belong to a later product generation than the interaction data, which ends in July 2014. The 23,838 products resolve to only 844 distinct URL slugs, so a median of fifteen products share a slug, while 18.9% have none at all; within a single product type, there are between 11 and 61 distinct slugs. This has a direct effect on the content component. Its TF-IDF representation contains only 877 distinct vectors for 23,838 products, and the average product is indistinguishable from 141 others, meaning the component cannot rank products within such a group. The product-type field, by contrast, is present and consistent for every item. We therefore report the content-based results using both feature sets, so that the weakness of the content signal can be separated from the weakness of the metadata it is based on.

Recommendation Algorithms

Product descriptions are vectorized using the Term Frequency–Inverse Document Frequency (TF-IDF) method, as defined in equation (1), to represent the textual content of products as numerical feature vectors.

$$\begin{aligned} \text{TF-IDF}(t, d, D) &= \text{TF}(t, d) \times \text{IDF}(t, D) \\ \text{IDF}(t, D) &= \ln\left(\frac{N}{df(t)}\right) \end{aligned} \quad (1)$$

and compared by cosine similarity, equation (2):

$$\cos(A, B) = \frac{A \cdot B}{\|A\| \|B\|} \quad (2)$$

An item's content score is defined as the highest cosine similarity between that item and any item the user has rated positively, i.e. "similar to something you bought." This matches how the item-to-item similarity panel works in the deployed system. The deployed engine additionally applies a minimum similarity threshold of 0.2, selected because higher values reduce the candidate pool too much while lower values allow items that appear similar but differ significantly in price or brand, and restricts candidates to a price range of $\pm 50\%$ around the reference item. The offline evaluation omits both filters so that all components are scored over the full catalogue under identical conditions.

The collaborative component uses an item-based k-nearest-neighbor (ItemKNN) model constructed from a binary matrix of positive ratings, with ratings of 4.0 or higher classified as positive interactions. For each item, we calculate cosine similarity with all other items, keep the top 200 most similar ones, and then estimate a user's interest in item i by adding up the similarities between item i and the items that user has rated positively, see equation (3).

$$s_{CF}(u, i) = \sum_{j \in I_u^+ \cap N_k(i)} \text{sim}(i, j) \quad (3)$$

Trend Since there is no live purchase stream, the trend component approximates product popularity using accumulated popularity, as defined in Equation (4):

$$\text{trend}_{score}(i) = \text{review}_{count}(i) \times \text{avg}_{rating}(i) \quad (4)$$

Only items with at least ten reviews and an average rating above 4.0 are included, which prevents single-review items from dominating.

Each component score is min–max normalized for each user across the candidate set and then combined linearly to produce the hybrid recommendation score, as defined in Equation (5):

$$\begin{aligned} \text{score}(u, i) &= w_C \tilde{s}_C(u, i) + w_{CF} \tilde{s}_{CF}(u, i) + w_T \tilde{s}_T(i) \\ w_C + w_{CF} + w_T &= 1 \end{aligned} \quad (5)$$

The choice of weights is treated as an empirical question and determined through sensitivity analysis. When the behavioral components provide no signal, as in the case of a new product or user, the system falls back to pure content-based ranking rather than returning an empty panel.

Evaluation Protocol

Each user's interactions are sorted by timestamp, and the most recent 20% (at least one interaction) are held out as the test set. This per-user temporal split avoids leakage from random splits, where a model might train on interactions that happen after the ones it is supposed to predict. Training uses all 1,289,576 remaining interactions from every user, including one-shot users, who cannot be evaluated but still contribute to item statistics. Evaluation is limited to the 23,952 users who have at least five interactions, at least two positive training interactions, and at least one positive held-out interaction. A held-out item is considered relevant if its rating is at least 4.0. Ranking is done over the full catalogue, excluding only the user's training items. We intentionally avoid sampled negatives because they are known to distort ranking metrics [40]. Accuracy is reported at cut-off 10, with NDCG@10 as the main metric, alongside MAP@10, hit ratio, precision, recall, and F1. For beyond-accuracy measures, we compute intra-list diversity (one minus the mean pairwise similarity of recommended items in a content-embedding space), category diversity, novelty (mean self-information of recommended items), and catalogue coverage. Every experiment is repeated five times with independent 5,000-user samples and reseeded models, and results are reported as mean $\pm$ standard deviation. Headline comparisons are also tested using per-user paired Wilcoxon signed-rank tests, pooled over runs ($n = 25,000$) and Holm-corrected.

Hyperparameter Selection

Hyperparameters were tuned on a validation split taken from the training data by applying the same per-user temporal rule again, so the test set played no role in model selection. The grids and the values selected were: ItemKNN neighbourhood size $k \in \{50, 100, 200, 500\}$, selecting 200; PureSVD factors $f \in \{16, 32, 64, 128, 256\}$, selecting 256; and BPR factors $\in \{32, 64, 128\}$, learning rate $\in \{0.01, 0.05\}$, iterations $\in \{100, 300\}$, selecting 128, 0.05, and 300. Random seeds 0–4 govern user sampling and model initialisation. As a reference point we also reproduce the collaborative component of an earlier prototype of our engine, which assigned every candidate a fixed score of 0.8 instead of computing item–item similarity. It appears in the tables as "CF proxy (0.8)", and the 50/30/20 hybrid built on it as "Hybrid-P 50/30/20".

IMPLEMENTATION

System Architecture

The system runs on a Python/Flask backend with SQLAlchemy and MySQL, paired with a React single-page frontend. We centralised the recommendation logic into a single engine class that supports content-based, collaborative, and hybrid approaches. This design keeps the API layer intact even if we swap out internal components. To improve on standard per-request processing, the system avoids on-the-fly similarity computation by using precomputed item neighbourhoods retrieved from an indexed database table for collaborative filtering, while content filtering relies on TF-IDF cosine similarities computed against an in-memory index that is built on the first request and reused thereafter.

Offline Artefacts and Database Design

To handle the bulk of the computational workload offline, a single build script generates two main artefacts in approximately 25 seconds on a standard laptop. The first is an item-similarity table containing approximately 1.56 million rows of item, neighbour, score, and rank values, which replaces complex online similarity calculations with a fast indexed join. The second is a materialised statistics table containing precomputed review counts, average ratings, and trend scores for each product. Precalculating these statistics substantially improves performance. Originally, computing trends meant re-aggregating the entire 1.35-million-row ratings table on every API call, taking a median time of 8.8 seconds. With the materialised table, that same query finishes in just 26 ms. The gain comes from moving the aggregation off the request path, a standard precomputation trade-off for read-heavy e-commerce workloads [32]: cheaper reads in exchange for a rebuild step and a bounded period of staleness.

Algorithms 1 and 2 outline the offline build process and the online scoring pipeline, respectively.

Algorithm 1 – Offline artefact build

Input: ratings R , positive threshold $\rho=4.0$, neighbourhood size $k=200$

- 1 $P \leftarrow \{(u,i) : r_{ui} \geq \rho\}$
- 2 normalise columns of P to unit L2 norm
- 3 for each item i : $S_i \leftarrow \text{TopK of } (P^T P)_i$, excluding i
- 4 write S to item_similarity(ProductId, NeighborId, Score, Rank)
- 5 stats $\leftarrow$ per-item (review_count, avg_rating, review_count $\times$ avg_rating)
- 6 write stats to product_stats

Algorithm 2 – Online hybrid scoring

Input: reference product p , list size n , weights (w_C, w_{CF}, w_T)

- 1 $C \leftarrow$ top 3n items by TF-IDF cosine to p , $\text{sim} \geq \theta_c$, price within $\pm 50\%$
- 2 $CF \leftarrow$ top 3n neighbours of p from item_similarity, $\text{avg_rating} \geq p$
- 3 $T \leftarrow$ top 3n same-category items from product_stats by trend_score
- 4 for each component $X \in \{C, CF, T\}$: $\tilde{s}_X \leftarrow$ min-max normalise X 's scores
- 5 $\text{score}(i) \leftarrow w_C \cdot \tilde{s}_C(i) + w_{CF} \cdot \tilde{s}_{CF}(i) + w_T \cdot \tilde{s}_T(i)$, $i \in C \cup CF \cup T$
- 6 if all components empty: return content-based fallback
- 7 return top- n items by score

API

The REST API exposes endpoints for recommendation, product detail, category, and analytics, with input validation and structured error responses. Endpoints are version-prefixed, so the API can evolve without breaking existing clients.

RESULTS

Baseline Comparison

Tables 2 and 3 compare the performance of the individual methods and the hybrid model on accuracy and beyond-accuracy metrics. Three findings stand out, concerning the strength of the behavioural signal, the weakness of the attribute signal, and the concentration of the popularity baseline:

Table 2. Ranking Accuracy of Individual Methods and Hybrids (mean $\pm$ std over 5 runs; $k = 10$)

System	Precision	Recall	F1	HR	NDCG	MAP
Popularity	0.0022 $\pm$ 0.0002	0.0143 $\pm$ 0.0013	0.0038 $\pm$ 0.0003	0.0220 $\pm$ 0.0018	0.0068 $\pm$ 0.0006	0.0038 $\pm$ 0.0005
Content	0.0005 $\pm$ 0.0001	0.0021 $\pm$ 0.0004	0.0008 $\pm$ 0.0001	0.0044 $\pm$ 0.0007	0.0012 $\pm$ 0.0002	0.0006 $\pm$ 0.0002
CF proxy (0.8)	0.0011 $\pm$ 0.0002	0.0059 $\pm$ 0.0013	0.0018 $\pm$ 0.0003	0.0103 $\pm$ 0.0020	0.0029 $\pm$ 0.0005	0.0015 $\pm$ 0.0002
ItemKNN	0.0107 $\pm$ 0.0003	0.0604 $\pm$ 0.0015	0.0174 $\pm$ 0.0005	0.0910 $\pm$ 0.0019	0.0418 $\pm$ 0.0013	0.0307 $\pm$ 0.0013
PureSVD	0.0063 $\pm$ 0.0004	0.0318 $\pm$ 0.0019	0.0099 $\pm$ 0.0005	0.0530 $\pm$ 0.0019	0.0216 $\pm$ 0.0013	0.0149 $\pm$ 0.0011
BPR	0.0069 $\pm$ 0.0004	0.0374 $\pm$ 0.0014	0.0111 $\pm$ 0.0005	0.0592 $\pm$ 0.0026	0.0238 $\pm$ 0.0012	0.0162 $\pm$ 0.0009
Hybrid-P 50/30/20	0.0019 $\pm$ 0.0002	0.0113 $\pm$ 0.0014	0.0032 $\pm$ 0.0003	0.0184 $\pm$ 0.0014	0.0063 $\pm$ 0.0007	0.0039 $\pm$ 0.0006
Hybrid 50/30/20	0.0031 $\pm$ 0.0001	0.0145 $\pm$ 0.0010	0.0049 $\pm$ 0.0002	0.0264 $\pm$ 0.0011	0.0111 $\pm$ 0.0005	0.0079 $\pm$ 0.0004
Hybrid 10/70/20	0.0108 $\pm$ 0.0003	0.0616 $\pm$ 0.0020	0.0176 $\pm$ 0.0005	0.0922 $\pm$ 0.0025	0.0425 $\pm$ 0.0013	0.0311 $\pm$ 0.0012

Table 3. Beyond-Accuracy Metrics (mean $\pm$ std over 5 runs)

System	ILD	Category diversity	Novelty (bits)	Coverage
Trend (pop.)	0.903 $\pm$ 0.000	0.400 $\pm$ 0.000	9.34 $\pm$ 0.00	0.0006 $\pm$ 0.000
Content	0.317 $\pm$ 0.003	0.181 $\pm$ 0.001	15.13 $\pm$ 0.00	0.646 $\pm$ 0.002
CF proxy (0.8)	0.883 $\pm$ 0.001	0.562 $\pm$ 0.002	14.08 $\pm$ 0.00	0.391 $\pm$ 0.001
ItemKNN	0.886 $\pm$ 0.002	0.569 $\pm$ 0.002	15.07 $\pm$ 0.01	0.560 $\pm$ 0.003
PureSVD	0.911 $\pm$ 0.001	0.568 $\pm$ 0.002	11.73 $\pm$ 0.01	0.047 $\pm$ 0.001
BPR	0.911 $\pm$ 0.003	0.606 $\pm$ 0.005	15.12 $\pm$ 0.02	0.601 $\pm$ 0.003
Hybrid 10/70/20	0.860 $\pm$ 0.002	0.523 $\pm$ 0.002	14.57 $\pm$ 0.02	0.545 $\pm$ 0.002

First, the item-neighbourhood model outperforms the rest by a wide margin. ItemKNN reaches an NDCG@10 of 0.0418 ± 0.0013 , which is roughly double the tuned latent-factor baselines (PureSVD at 0.0216, BPR at 0.0238) and about 35 times higher than the content component. Replacing the earlier fixed collaborative heuristic with genuine ItemKNN improved performance by fourteen times, from 0.0029 to 0.0418 (paired Wilcoxon $p < 10^{-300}$). Training cost separates the models as sharply as accuracy: ItemKNN fits in 0.3 s against 12.6 s for BPR and 23.6 s for PureSVD, with the content model at 0.1 s and the popularity and fixed-proxy baselines effectively free. H3 is therefore supported on both counts, and we return to the mechanism in the Discussion.

Second, the content-based component was the weakest signal for accuracy, at 0.0012. In fact, popularity alone was 5.7 times more accurate, which leads us to reject H2. However, the content component did achieve the highest catalogue coverage of any system tested (0.646), spreading recommendations across a wider part of the catalogue than any behavioural component when used alone.

Because that representation depends partly on an unreliable URL field, we repeated the comparison using the product-type field alone, which is complete for every item. The conclusion does not depend on the questionable metadata. Using product type alone the content model reaches an NDCG@10 of 0.0005 and popularity is 14.4 times more accurate (paired Wilcoxon, $p = 9.8 \times 10^{-76}$); using the URL slug alone it reaches 0.0004 and popularity is 17.6 times more accurate. The same re-test puts the published type-plus-slug configuration at 5.8, so on a consistent comparison it is the most favourable of the three for

content-based filtering, and H2 is rejected under all three configurations. The slug appears to help only by separating otherwise identical items rather than by describing them.

Finally, the popularity baseline highlights the limits of a naive approach. With catalogue coverage of only 0.06%, it recommends the same 14 products to almost every visitor.

Ablation Study

Table 4 evaluates all seven component combinations, and Figure 2 shows the same comparison in ranked form.

Table 4. Ablation over Component Combinations (mean $\pm$ std; C = content, CF = ItemKNN, T = trend)

Combination	NDCG@10	F1@10	HR@10
C only	0.0012 $\pm$ 0.0002	0.0008 $\pm$ 0.0001	0.0044 $\pm$ 0.0007
CF only	0.0418 $\pm$ 0.0013	0.0174 $\pm$ 0.0005	0.0910 $\pm$ 0.0019
T only	0.0068 $\pm$ 0.0006	0.0038 $\pm$ 0.0003	0.0220 $\pm$ 0.0018
C+CF (62/38)	0.0109 $\pm$ 0.0009	0.0047 $\pm$ 0.0003	0.0255 $\pm$ 0.0016
C+T (71/29)	0.0026 $\pm$ 0.0004	0.0013 $\pm$ 0.0001	0.0079 $\pm$ 0.0004
CF+T (60/40)	0.0380 $\pm$ 0.0011	0.0154 $\pm$ 0.0005	0.0836 $\pm$ 0.0026
C+CF+T 50/30/20	0.0111 $\pm$ 0.0005	0.0049 $\pm$ 0.0002	0.0264 $\pm$ 0.0011
C+CF+T 10/70/20	0.0425 $\pm$ 0.0013	0.0176 $\pm$ 0.0005	0.0922 $\pm$ 0.0025

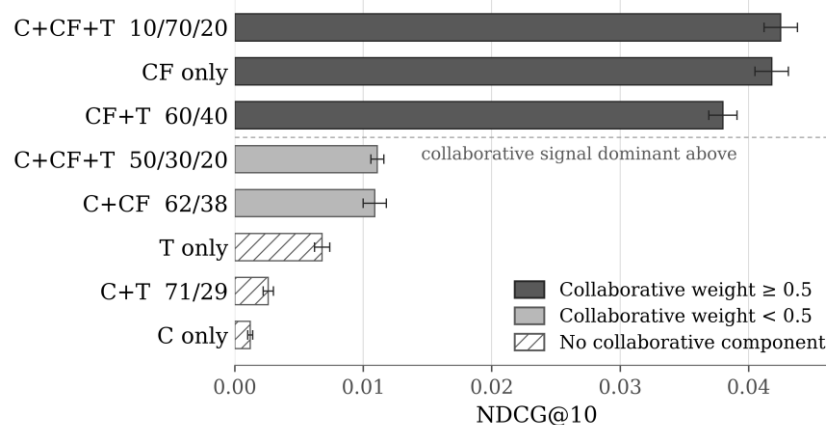


Figure 2. NDCG@10 by Component Combination (ablation, Table 4). Error Bars Show One Standard Deviation Over Five Runs. Fills Group the Combinations by the Weight Given to the Collaborative Component: The Key Difference Between Configurations is not Whether the Component is Present, but Whether it Dominates.

Every combination that includes the collaborative component achieves an NDCG@10 of at least 0.0109, whereas combinations without it do not exceed 0.0068. Two findings stand out. First, the CF+T model with a fixed 60/40 weighting (0.0380) performs worse than CF alone, indicating that adding the trend component can reduce performance when its weight is too high. The sensitivity analysis shows that the trend signal is only beneficial at low weights. Second, the full hybrid with the original 50/30/20 weighting (0.0111) performs substantially worse than ItemKNN, whereas the same three components weighted at

10/70/20 achieve an NDCG@10 of 0.0425 and outperform ItemKNN. These results show that the effectiveness of the hybrid depends on the choice of weights rather than the combination of components alone. At the optimal weighting, the hybrid achieves a small but statistically significant improvement over the best individual model ($\Delta \text{NDCG@10} = +0.0006$, $p = 0.012$). This supports H1 at this operating point. The practical size of the advantage is reported below, and the robustness analysis shows that it depends strongly on how much interaction data is available.

Weight Sensitivity

Figure 3 shows NDCG@10 for all 66 weight combinations. The results do not reveal a single sharp optimum. Instead, several configurations with a collaborative weight between 0.6 and 1.0 and a trend weight of up to 0.3 achieve similar performance. Performance decreases steadily as the content weight increases beyond 0.2. At the original weighting of 50/30/20, NDCG@10 is only 26% of the best result. This suggests that the hybrid is not highly sensitive to small changes in the weights, provided that the collaborative component remains dominant. The two best configurations show almost no difference in performance ($\Delta \text{NDCG@10} = 0.00002$), so the choice cannot be based on accuracy alone. We retain 10/70/20 rather than 0/80/20 as the reference configuration because the two are statistically indistinguishable here, so the content weight has no measurable cost. We no longer justify it by the cold-start fallback: the churn analysis reported below shows that, at this weighting, the content component provides no exposure for products without interaction history.

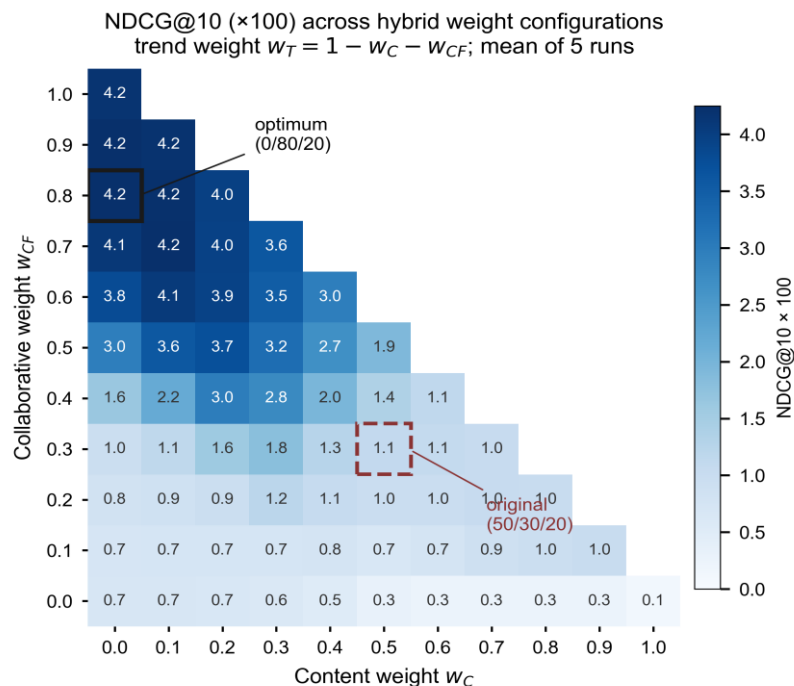


Figure 3. NDCG@10 Across the Hybrid Weight Simplex (mean of 5 runs). The Optimum Region is Collaborative-Dominant; The Originally Proposed 50/30/20 Configuration is Marked.

Statistical Significance

Table 5 reports paired per-user Wilcoxon signed-rank tests across the pooled runs. Every comparison remains significant after Holm correction, and all but one at $p < 0.001$. The exception is the hybrid's improvement over ItemKNN alone, which is significant at the 5% level ($p = 0.012$). Its rank-biserial correlation is 0.082, and its confidence interval reaches almost to zero, so significance and magnitude should be considered together throughout this table rather than separately. The largest differences are observed when comparing ItemKNN with the fixed collaborative proxy, and when comparing the hybrid using ItemKNN with the hybrid using the fixed proxy at the same weights. These results show that replacing the proxy with true item-based collaborative filtering has the greatest effect on recommendation performance.

Table 5. Paired per-user Wilcoxon Signed-Rank Tests on NDCG@10 (pooled over 5 runs, $n = 25,000$; Holm-corrected).

Comparison	Mean Δ	95% CI of Δ	r	p (Holm)
Hybrid 10/70/20 vs Trend	+0.0356	{+0.0336, +0.0377}	0.756	5.9×10^{-257}
Hybrid 10/70/20 vs Content	+0.0413	{+0.0394, +0.0432}	0.984	$< 10^{-300}$
Hybrid 10/70/20 vs CF proxy	+0.0396	{+0.0376, +0.0415}	0.915	$< 10^{-300}$
Hybrid 10/70/20 vs ItemKNN	+0.0006	{+0.0002, +0.0011}	0.082	1.2×10^{-2}
Hybrid 10/70/20 vs PureSVD	+0.0209	{+0.0189, +0.0229}	0.439	3.4×10^{-92}
Hybrid 10/70/20 vs BPR	+0.0187	{+0.0170, +0.0204}	0.501	1.2×10^{-109}
ItemKNN vs CF proxy	+0.0389	{+0.0370, +0.0409}	0.912	$< 10^{-300}$
Hybrid 50/30/20 (real CF) vs Hybrid-P 50/30/20	+0.0047	{+0.0040, +0.0055}	0.489	3.0×10^{-31}

Note: The interval is a 95% bootstrap confidence interval for the mean difference (10,000 resamples), and r is the matched-pairs rank-biserial correlation, so the size of each difference can be read alongside its p -value. The first and fourth rows make the contrast clear: both are significant, but one changes the ranking for most users while the other does so for very few.

Practical Significance

A p -value computed over 25,000 paired observations shows that a difference is real, but not necessarily that it is large enough to matter in a store. We therefore also report standardised effect sizes and an operational interpretation of each main comparison. The advantage of the hybrid over ItemKNN is real but very small. Cohen's d_z is 0.017 and the matched-pairs rank-biserial correlation is 0.082, with a 95% bootstrap confidence interval for the mean difference of {+0.0002, +0.0011}. The two systems show identical items to 95.1% of users; among the 4.9% whose lists differ at all, the hybrid ranks better for 631 users and worse for 605. In terms of hit ratio, the hybrid brings a relevant product into the top ten for 1.2 additional users in every thousand, which is one additional served user for every 862 recommendation lists.

The comparisons treated as secondary in the paper are much larger on the same scale. Replacing the fixed collaborative proxy with item-based collaborative filtering gives a rank-biserial correlation of 0.912 and serves one additional user for every 12 lists. Re-weighting the hybrid from 50/30/20 to 10/70/20 gives 0.825 and one additional user for every 15 lists. In practical terms, the choice of collaborative component and the choice of

weights each have an effect between fifty and seventy times larger than adding content and trend signals to an already working collaborative model. H1 is therefore supported statistically, but the difference is small, and we do not claim that the hybrid layer alone justifies the additional implementation effort.

Robustness to Sparsity and Catalogue Churn

The results above describe a single operating point. To see how much they depend on this setting, we repeated the evaluation while reducing the training data in three ways and separately varied the two arbitrary constants in the protocol. In every condition, the test set and evaluated user samples are kept fixed, so all conditions are scored against the same held-out purchases, and a user whose profile becomes empty after thinning remains in the denominator. Thinning is applied to the training records before any derived quantity is calculated, so item popularity and the equation (4) trend gate reflect only the interactions that remain. These runs use three seeds rather than five; at full density, they reproduce Table 2 within one standard deviation. Table 6 reports all three thinning conditions.

Removing a random fraction of all training interactions reduces the performance of the behavioural models but not the popularity baseline. Retaining 75%, 50%, 25% and 10% of training interactions, ItemKNN falls from 0.0410 to 0.0291, 0.0151, 0.0060 and 0.0021, while the popularity baseline stays at 0.0066, 0.0065, 0.0066 and 0.0067. This is because random thinning preserves the relative order of popular products. Below roughly a quarter of the observed density, a best-seller list is therefore more accurate than item-based collaborative filtering. The hybrid stays close to ItemKNN at high density but separates from it as data becomes scarce: in these three-seed runs its advantage over ItemKNN alone grows from 1.7% at full density to 4.6% at half, 52% at a quarter and 262% at a tenth. This happens because the trend component continues to contribute after the collaborative signal has weakened.

The two sides of the interaction matrix behave differently, which explains the ordering of the models. Capping the interactions retained per product reduces the mean number of positive interactions per product from 41.6 to 25.1, 17.4, 7.7 and 3.9. Across this range, the advantage of ItemKNN over PureSVD falls steadily from 1.88 times to 1.43, 1.28, 1.20 and finally 1.00. Thus, at about four positive interactions per product, the neighbourhood model has no advantage over the latent-factor model. Capping the interactions retained per user has the opposite effect: at caps of five, three and two interactions per user, the same ratio is 2.32, 2.05 and 1.89, at or above its value on the full data. This is because the interactions removed by a per-user cap come from the long-profile users who provide most of the benefit to the latent-factor models. The advantage of item-based collaborative filtering in this setting is therefore mainly a property of item-side density, while user-side sparsity does not reduce it.

One boundary condition should be stated, because the two margins are not fully independent. Item-item similarity is itself computed from user profiles, so a user contributing a single interaction does not contribute to co-occurrence. At a cap of one interaction per user, co-occurrence disappears and ItemKNN collapses to 0.0003, below

both latent-factor baselines and the popularity baseline. The item-side signal that the neighbourhood model depends on is generated by users with more than one interaction, even though the model does not represent those users individually.

Table 6. Robustness of NDCG@10 to Training-Data Sparsity (mean of 3 runs). Random Thinning Removes Interactions at Random; the Per-user and Per-item Caps Thin One Margin of The Interaction Matrix at a Time.

Condition	Kept %	Pos./item	Popularity	ItemKNN	PureSVD	BPR	Hybrid
Full data	100	41.6	0.0067	0.0410	0.0218	0.0235	0.0417
Random 75%	75	31.2	0.0066	0.0291	0.0184	0.0163	0.0299
Random 50%	50	20.9	0.0065	0.0151	0.0116	0.0090	0.0158
Random 25%	25	10.6	0.0066	0.0060	0.0054	0.0043	0.0091
Random 10%	10	4.7	0.0067	0.0021	0.0016	0.0016	0.0076
≤ 5 per user	95	39.7	0.0066	0.0397	0.0171	0.0217	0.0401
≤ 3 per user	91	37.8	0.0065	0.0286	0.0140	0.0179	0.0285
≤ 2 per user	84	35.0	0.0063	0.0198	0.0105	0.0116	0.0204
≤ 1 per user	69	28.4	0.0057	0.0003	0.0011	0.0005	0.0051
≤ 50 per item	60	25.1	0.0009	0.0236	0.0165	0.0161	0.0257
≤ 25 per item	42	17.4	0.0003	0.0151	0.0118	0.0115	0.0158
≤ 10 per item	18	7.7	0.0004	0.0047	0.0039	0.0035	0.0048
≤ 5 per item	9	3.9	0.0003	0.0015	0.0015	0.0013	0.0016

The second analysis tests the claim that the content component makes the hybrid more robust to catalogue turnover. Because only 0.02% of products here are genuinely cold, we created this condition by removing all training interactions from a random 10%, 25% and 50% of products while leaving their held-out purchases in the test set. Overall accuracy declines as expected, with the hybrid falling from 0.0417 to 0.0384, 0.0314 and 0.0215 (Table 7). The more informative measure is what happens to the cold products themselves. Recall at ten for held-out purchases that fall on cold products is 0.0000 for every system at 10% churn and rises only to 0.0031 for the content-only model at 50% churn. The content component does place cold products in its lists in proportion to their share of the catalogue, filling 9.9%, 25.2% and 52.7% of its recommendations with them. However, it almost never selects the particular cold product that a user later went on to buy, which follows from the metadata degeneracy reported in the pre-processing section. At the recommended 10/70/20 weighting, the hybrid places no cold products in its lists at all because the collaborative component scores them at zero and holds 70% of the weight. Surfacing new products and ranking them accurately are therefore separate problems, and this content representation solves neither at any weighting that preserves overall accuracy.

The third analysis varies the relevance threshold and the eligibility floor (Table 8). Changing the relevance threshold from a rating of at least 4.0 to at least 3.0 or 5.0 changes the absolute NDCG@10 values but leaves the ordering of the systems almost unchanged, with Kendall rank correlations of 0.87 and 0.73 against the published ordering. The hybrid remains the most accurate system under both thresholds. Raising the eligibility floor has a stronger effect. Among users with at least ten interactions, ItemKNN reaches 0.0384

compared with 0.0370 for the hybrid, and among users with at least twenty interactions, it reaches 0.0455 compared with 0.0403. Thus, for users with longer histories, the hybrid's advantage does not simply become smaller but actually reverses. PureSVD also closes most of its gap on this population, reaching 0.0406 compared with ItemKNN's 0.0455. This is the same pattern seen in the thinning experiments, but from the other direction: the benefit of combining signals decreases as the amount of behavioural evidence per user increases.

Table 7. Catalogue-churn Stress Test. A Random Share of Products Has all Training Interactions Removed While Its Held-Out Purchases Are Retained. Cold-item Recall is the Share of Those Purchases Placed in the Top Ten.

Metric	Churn	Content	ItemKNN	Hybrid 10/70/20	Hybrid 50/30/20
NDCG@10	0%	0.0011	0.0410	0.0417	0.0107
	10%	0.0012	0.0376	0.0384	0.0100
	25%	0.0010	0.0304	0.0314	0.0082
	50%	0.0010	0.0209	0.0215	0.0044
Recall@10, cold items	10%	0.0000	0.0000	0.0000	0.0000
	25%	0.0013	0.0000	0.0000	0.0000
	50%	0.0031	0.0001	0.0000	0.0000
Cold items in list (%)	10%	9.93	0.02	0.00	0.04
	25%	25.24	0.41	0.00	0.40
	50%	52.68	7.19	0.00	3.04

Table 8. Sensitivity of NDCG@10 to the Relevance Threshold and the Evaluation Floor. τ is Kendall's Rank Correlation of the System Ordering Against the Published Condition.

Condition	Eligible users	ItemKNN	PureSVD	BPR	Hybrid	τ
published (≥ 4.0 , ≥ 5)	23,952	0.0410	0.0218	0.0233	0.0417	1.000
relevance ≥ 3.0	26,210	0.0441	0.0239	0.0234	0.0447	0.867
relevance ≥ 5.0	18,449	0.0387	0.0232	0.0229	0.0388	0.733
eval floor ≥ 10	6,027	0.0384	0.0271	0.0198	0.0370	0.733
eval floor ≥ 20	1,367	0.0455	0.0406	0.0310	0.0403	0.467

Together, these analyses qualify H1. The advantage of the hybrid over the best single model is statistically real but practically negligible at the operating point in Table 2. It grows substantially as data becomes scarcer but reverses for users with longer histories. We therefore accept H1 only in the data-poor setting targeted by this study, and state it in this conditional form rather than as a general result.

Live Performance

Table 9 summarises the performance of the deployed recommendation system on the production setup (running MySQL 9.7). Performance was measured using 100 sequential requests for each recommendation method after an initial warm-up request, followed by a burst test of 50 requests across four threads. Each method was tested in a separate process so that memory usage could be measured accurately. The hybrid recommendation engine achieved a median response time of 34 ms and a 95th percentile latency of 40 ms. During

the burst test, it processed 107 requests per second. This is approximately 13 times faster than an earlier build of our engine, which required 445 ms per request, while also supporting full item-based collaborative filtering. Updating the recommendation models is also efficient. Rebuilding both offline artefacts takes about 25 seconds, allowing the models to be refreshed with a scheduled job, such as an overnight update. One limitation should be noted. The rebuilt database uses fixed placeholder product prices because the original supplier prices were not available. As a result, the price-band filter functions correctly, but it does not fully reflect real-world pricing.

Table 9. Live Performance of the Deployed Engine

Method	Median (ms)	p95 (ms)	Throughput (req/s)	Memory RSS (MB)
Content-based	1	1	1,121	157
Collab. (ItemKNN)	1	1	2,919	137
Trend	26	28	135	137
Hybrid 10/70/20	34	40	107	156

DISCUSSION

The hypotheses are supported in the conditional form stated above, but the more useful content of the results is what they reveal about the conditions under which each paradigm works. This section therefore moves from the specific to the general. It begins with the mechanisms behind the individual results, turns to a comparison with the architectures that define current practice, and then states a set of propositions: what we believe transfers beyond this dataset, under which conditions, and what would show that it does not. The remaining subsections consider the popularity baseline, the dropshipping setting, and the limits of the study in light of these propositions.

Why Behavioural Similarity Dominates and Content Fails Here

Two different mechanisms produce the ordering of the components here, and separating them is what P1 and P5 formalise. The first is where the evidence sits. The interaction matrix is extremely sparse overall, but unevenly so: the median user has only one rating, while each product carries an average of 57. A model indexed by items therefore estimates each of its quantities from tens of observations, while a model indexed by users estimates them from one or two. That is why the collaborative component reaches an NDCG@10 around 35 times that of the content component, and it is also why item-to-item collaborative filtering was adopted at scale by Amazon [5, 6]. The second mechanism is not sparsity at all, but resolution. The dataset offers a product type and a URL slug, and the audit reported in the Preprocessing section shows that the slug does not reliably identify the product it is attached to, leaving 877 distinct content representations for 23,838 products. No similarity function can separate items that its representation maps to the same point, so the content component recommends near-duplicates rather than the products users actually choose next, which are often from other categories. The distinction matters because the two failures have different remedies: the first is relieved by more interaction data, while the second requires better metadata. It also accounts for the

weighting. Authors in [13] describes weighted hybrids as a way of combining methods, but a weight is not a free design choice; it reflects the relative strength of what is being combined, so the collaborative-dominant optimum found here is a property of this dataset rather than a value to carry into the next one.

Why the Hybrid's Gain is Small but Real

A hybrid can improve on its strongest component only where that component has left items in the wrong order, and at full density ItemKNN has left few. This is P4 observed at a single operating point. The trend signal does useful work where many candidates have similar collaborative scores, promoting items with strong review performance, but it reaches only part of the catalogue, since 31.5% of items do not meet the eligibility threshold of at least ten reviews and an average rating above 4.0. The small content weight likewise allows similar products to remain in the list without reordering much of it. What remains is an improvement of 1.5% in relative NDCG@10, statistically significant across 25,000 paired observations, so H1 is supported at this density, although its practical margin is narrow and, as the robustness analysis shows, conditional on the amount of data available. The same reasoning accounts for a result we reported earlier and no longer stand behind. An earlier analysis of our own showed the hybrid raising F1 from 0.80 to 0.93, an improvement produced by a circular definition of relevance rather than by the model; under the revised protocol it disappeared, leaving the much smaller but reliable gain reported here. This is the gap between offline evaluation and real-world performance that in [41] describe, arising within a single study rather than between studies.

One further comparison puts the margin in perspective and bears on how much weight a statistically reliable difference should carry. Moving the evaluation floor from users with at least five interactions to users with at least ten, a defensible protocol choice rather than a change of model, moves the hybrid from 0.0007 above ItemKNN to 0.0014 below it within the same set of runs. The effect of that protocol decision is therefore larger than, and opposite in sign to, the effect of the modelling decision the protocol is being used to evaluate. Authors in [42] document how widely such choices vary across published recommender evaluations, which is one reason small margins reported under a single protocol are difficult to compare across papers. We therefore state H1 as conditional on both a stated protocol and a stated density, and we treat the operational quantity, one additional served user for every 862 lists, rather than the p-value, as the number a deployment decision should turn on.

Why Simple Beats Deep Under Extreme Sparsity

The latent-factor baselines lose here for the reason P1 states, not because they are more expressive than the problem needs. Matrix factorisation and BPR must estimate a vector of 128 or 256 parameters for every user. Most evaluation users, however, supply only one to four training interactions from which to estimate it. ItemKNN estimates no user representation at all and draws on item-level interactions, which are denser in this dataset. The consequence is that the tuned latent-factor models reach around half the accuracy of the neighbourhood model while requiring substantially more training time, 13–24 seconds

against 0.3 seconds per fit. These results are consistent with the reproducibility findings of [15, 16]. They extend those findings to a setting with much higher sparsity and stricter resource constraints. They do not imply that deep or sequential models are ineffective in general. The following subsection considers each family in turn. Once more session-level data becomes available, evaluating sequential methods in a live setting would be a useful direction for future work. Our claim is the narrower one that, under the sparsity and budget constraints considered here, neighbourhood methods offer a better combination of accuracy and computational cost than the latent-factor models we compared them with. H3 is therefore supported, and in fact exceeded. The neighbourhood model does not merely match the latent-factor baselines at lower cost. It outperforms them on every accuracy metric while training in 0.3 seconds against 13–24 seconds.

Comparison with Modern Recommendation Architectures

The models evaluated here are deliberately conventional, so it is fair to ask how current recommender architectures would behave in the same setting. We cannot answer it by running them, which is outside the budget assumed in this study. We can, however, measure how far this catalogue sits from the data on which those architectures are reported, and reason from a property of our own results. The two systems that estimate a parameter vector for every user, PureSVD with 256 factors and BPR with 128, reach NDCG@10 of 0.0216 and 0.0238. The system that estimates none, ItemKNN, reaches 0.0418. The penalty for relying on user-side parameters is therefore close to a factor of two in this dataset, and it comes from the evaluation population rather than from any detail of training: the median evaluated user has four training positives, and 30% have three or fewer. A model whose representational power sits on the user side faces this limitation regardless of how it is optimised.

The distance can be measured directly because the filters those papers apply to their data are stated in them and can be applied to ours. Neural collaborative filtering retains users with at least twenty interactions [7]. SASRec discards users and items with fewer than five actions [11], and BERT4Rec keeps users with at least five [12]. Applied to our catalogue, the first retains 0.16% of users and 3.46% of interactions. The second retains 2.4% of users and 13.88% of interactions while also removing 54% of the catalogue. The third retains 3.17% of users and 18.07% of interactions (Table 10).

What these filters remove is not a tail: 77.1% of our users hold exactly one interaction, and users with fewer than five hold 81.9% of all interactions. Two of these papers evaluate on Amazon Beauty, the same product category used here. After filtering, their versions carry 7.6 and 8.8 actions per user, against 1.53 in the unfiltered catalogue. The graph benchmarks are further away still. The sparsest of LightGCN's three datasets is about ten times denser than ours [9].

These architectures are therefore not simply untested at this operating point. The preprocessing that conventionally precedes them defines the operating point away from this regime, and with it the users a small store most needs to serve.

Table 10. Catalogue Interaction Density and Benchmark Comparison Under Published Preprocessing Filters

Dataset	Users	Items	Interactions	Per user	Density (%)	this study
This study (as evaluated)	883,753	23,838	1,348,246	1.53	0.0064	1.0
This study, NCF filter	1,375	11,892	46,703	34.0	0.2856	44.6
This study, BERT4Rec filter	28,040	22,445	243,601	8.69	0.0387	6.0
This study, SASRec filter	21,208	10,907	187,194	8.83	0.0809	12.6
MovieLens-1M (NCF)	6,040	3,706	1,000,209	165.6	4.4684	698.2
Pinterest (NCF)	55,187	9,916	1,500,809	27.2	0.2743	42.9
Gowalla (LightGCN)	29,858	40,981	1,027,370	34.4	0.0840	13.1
Yelp2018 (LightGCN)	31,668	38,048	1,561,406	49.3	0.1296	20.2
Amazon-Book (LightGCN)	52,643	91,599	2,984,108	56.7	0.0619	9.7
Amazon Beauty (SASRec)	52,024	57,289	0.4M	7.6	0.0134	2.1
Amazon Beauty (BERT4Rec)	40,226	54,542	0.35M	8.8	0.0160	2.5

Note: Benchmark counts are as reported in the [7, 9, 11, 12]. SASRec and BERT4Rec report actions to one decimal place in millions, so their interaction counts and densities are approximate; their per-user averages are as reported. The final column gives density relative to this study's evaluated dataset.

Neural collaborative filtering [7] replaces the inner product of matrix factorisation with a learned interaction function but still uses a per-user embedding which it relies on the same user-side signal that limits BPR and PureSVD. LightGCN [9] propagates embeddings over the user-item graph, and its advantage comes from combining information across multi-hop neighbourhoods. That advantage depends on how much graph there is to traverse. With a median of four training interactions per evaluated user and no side information to add edges, the user-side neighbourhood available for propagation is small. The smoothing that gives LightGCN its accuracy on denser benchmarks therefore has little information to work with here. We would expect it to behave more like a well-regularised item-similarity model than like the graph model reported on standard benchmarks. The density figures above make the same point from the data side: the neighbourhood that propagation would traverse here is roughly an order of magnitude thinner than in the benchmarks where the method was shown to work well.

The sequential and transformer families are excluded by the data rather than simply by its sparsity. SASRec [11] and BERT4Rec [12] model the order of actions within a user session, and BERT4Rec also trains by masking items within a sequence. Both require sequences of meaningful length. Our data consists of ratings with timestamps but has no session boundaries, and the median user contributes a single rating, so for most users there is no sequence to model at all. This is not a matter of tuning or hardware: the input these architectures require is simply not present in the dataset. Session-level logging is part of the planned live deployment, and once it exists, a sequential model can be tested in precisely the setting where it would be expected to help. The filtered Beauty datasets used in both papers illustrate this concretely. Reaching their 7.6 and 8.8 actions per user requires

discarding roughly 97% of our users, those who fall below the five-action floor. That is the population this study is about.

Two qualifications apply. First, while the distance between the datasets is measured, how these models would rank on ours remains an expectation rather than a measurement. The reproducibility literature we rely on [15, 16] shows that expectations about neural recommender performance are more often wrong in the optimistic direction than the pessimistic one. Second, the argument is specific to this operating point. A store with longer user histories, richer first-party metadata, or session logs would move along exactly the axis these architectures require, and the reasoning above would no longer hold. Our claim concerns the regime we evaluate, not the methods in general.

Propositions, Boundary Conditions and Design Principles

We now state what we take the measurements to show beyond this dataset. Four constructs are used. Evidence locality is the margin of the interaction matrix on which observations accumulate. It is measured as mean positive interactions per user and per item. Parameter locality is the margin on which a model commits its estimates. PureSVD and BPR fit free parameters on both sides, while ItemKNN fits none at all. The quantities it estimates, the item-item similarities, are indexed by items alone. Metadata resolution is the number of distinct representations a feature set induces over a catalogue. Marginal combination utility is the gain of a weighted hybrid over its strongest single component. Each proposition below is stated with the mechanism we believe produces it, the evidence in this study, the conditions under which we expect it to hold, and what would falsify it, see Table 11.

P1 - Parameter-evidence alignment. Under sparsity, ranking accuracy is governed by the alignment between a model's parameter locality and the data's evidence locality, rather than by its representational capacity. The mechanism is estimation variance. Parameters placed on the sparse margin are fitted from few observations, so their error can dominate whatever the optimiser does. In this study, the item-indexed model reaches NDCG@10 of 0.0418. The two user-parameterised models reach 0.0216 and 0.0238. This is at a median of four training positives per evaluated user, compared with 41.6 per product. The proposition is restricted to settings where the two margins differ materially. It says nothing about catalogues that are dense on both sides. It would be falsified if a user-parameterised model matched an item-indexed one at comparable density asymmetry and with comparable tuning effort.

P2 - Asymmetry rather than aggregate sparsity. The ordering of paradigms is predicted by the ratio between item-side and user-side density, rather than by the sparsity of the matrix as a whole. Thinning the item margin drives the ItemKNN-to-PureSVD ratio from 1.88 down to 1.00. In contrast, capping interactions per user drives the same ratio up to 2.32. Both operations remove interactions and both increase aggregate sparsity. A single sparsity figure is therefore not a sufficient description of a dataset for the purpose of choosing a model. The prediction is testable on any two catalogues. Equal aggregate

sparsity with opposite asymmetry should produce opposite orderings. An ordering that instead tracked total density would falsify the proposition.

P3 - Ordered degradation thresholds. Each paradigm has a density below which it forfeits its advantage. These thresholds are ordered rather than simultaneous. Here, the neighbourhood advantage disappears at about four positive interactions per product. Below roughly a quarter of the observed density, a non-personalised best-seller list is more accurate than item-based collaborative filtering. At one interaction per user, co-occurrence vanishes altogether. The neighbourhood model then collapses to 0.0003, below every other system tested. The boundary condition is that the two margins are coupled through co-occurrence. Item-side evidence is produced by users with more than one interaction, even though the model does not represent those users individually. The practical consequence is that a store below its threshold should expect data, rather than a change of model, to be what helps.

P4 - Declining marginal utility of combination. The utility of combining signals is a decreasing function of the strength of the strongest component. The advantage of the hybrid over the collaborative model alone is 262% at a tenth of the observed density, 52% at a quarter, and 1.7% at full density in the three-seed thinning runs. This compares with 1.5% in the five-seed headline evaluation. The advantage is negative for users with ten or more interactions, where ItemKNN reaches 0.0384 against 0.0370 for the hybrid. The mechanism is that the weaker components mostly re-rank items that the dominant component has already ordered correctly. Their contribution therefore shrinks as that ordering improves. It follows that mixing weights are a property of a dataset at a particular point in its growth, rather than a transferable constant. This provides a more specific explanation for the non-transfer of published weights than dataset-specific characteristics alone. The prediction is that a store's hybrid advantage declines as it accumulates interactions. An advantage that stayed constant or grew would falsify the proposition.

P5 - The metadata resolution ceiling. The attainable accuracy of a content component is bounded by the number of distinct representations its feature set induces over the catalogue. This holds independently of the similarity function used and of the weight it receives. Two products mapped to the same point cannot be ranked against each other by any function of that representation. Here, 23,838 products reduce to 877 distinct vectors. The content component is the weakest signal under every configuration tested, including the product-type-only representation, which is free of the metadata defect. The proposition applies wherever item features are coarse relative to the catalogue. It predicts that improvement on the content side comes from increasing the number of distinct representations, rather than from changing the algorithm that consumes them.

P6 - Exposure and ranking are distinct capabilities. Placing items with no interaction history into a list and ranking them correctly are separate problems. A component can solve the first while contributing nothing to the second. The churn test shows the content component filling 9.9%, 25.2%, and 52.7% of its list with cold products. Yet cold-item recall at ten never

exceeds 0.0031. At the recommended weighting, the hybrid surfaces none at all. The boundary condition is P5. This separation is what a degenerate representation produces. A representation that resolved individual products could, in principle, supply both capabilities. The proposition matters because cold-start robustness is routinely claimed for hybrid designs on the strength of exposure alone. Exposure is the weaker of the two properties, and by itself it does not necessarily serve a user.

Table 11. Six Propositions, Supporting Measures, and Conditions for Their Applicability

	Proposition	Evidence in this study	Boundary condition
P1	Accuracy follows parameter-evidence alignment	0.0418 item-indexed vs 0.0216 / 0.0238 user-parameterised; 4 positives per user vs 41.6 per item	Margins must differ materially; silent where both are dense
P2	Asymmetry, not aggregate sparsity, orders the paradigms	Item-side thinning 1.88 $\rightarrow$ 1.00; user-side caps 1.88 $\rightarrow$ 2.32	Requires the two margins to be thinned independently
P3	Paradigms have ordered degradation thresholds	$\approx$ 4 positives per item; best-sellers win below $\approx$ 25% density; collapse to 0.0003 at 1 per user	Margins coupled through co-occurrence
P4	Combination utility falls as the best component improves	+262% at 10% density, +52% at 25%, +1.7% at full, negative at $\geq$ 10 interactions per user	Components unequal in strength; weights re-tuned per dataset
P5	Metadata resolution caps any content component	877 distinct vectors for 23,838 products; content weakest under every feature set	Item features coarse relative to catalogue size
P6	Exposure and ranking are separate capabilities	Cold items fill 9.9-52.7% of content lists; cold recall@10 $\leq$ 0.0031; 0 at 10/70/20	Holds under P5; a resolving representation could supply both

Note: The evidence column cites results reported in the Results section and the metadata audit in Pre-processing. No proposition rests on a measurement made outside this study.

These propositions sharpen three existing lines of work rather than standing apart from them. The first is the item-neighbourhood tradition. Deshpande and Karypis [43] framed item-based top-N recommendation as an estimation problem over item co-occurrence. The linear models that followed, SLIM [44] and its closed-form relative [45], improve on this by regularising item-item weights rather than introducing user parameters. P1 offers a data-side account of why this family is strong under sparsity. It places its parameters on the margin where the observations are, so added capacity is spent where it can be estimated. The second line is the finding of [46] that simple methods are competitive on top-N tasks. This was reinforced for neural recommenders by [15] and Rendle et al. [16]. That work is usually read as a claim about evaluation rigour and baseline tuning. P2 and P3 suggest a complementary reading in which part of the effect comes from the data. How far a dataset sits above a paradigm's threshold, and on which margin its density sits, determine how much room a more expressive model has to demonstrate an advantage. The third is the literature on popularity bias [47], which treats the dominance of popular items as a distortion to be corrected. P3 identifies a regime in which it is not a distortion but the accuracy-maximising policy. Below roughly a quarter of our observed density, a

non-personalised best-seller list outranks item-based collaborative filtering. What is a defect in a dense catalogue can therefore be the correct decision in a very sparse one. This is a boundary condition on the popularity-bias literature rather than an exception to it.

Two things follow for practice, and one for the status of these claims. First, the question a small retailer should ask is not how much interaction data the store holds in total, but how that data is distributed. A few thousand products, each carrying tens of interactions, can support a neighbourhood model well. The same volume spread across a much larger catalogue does not (P1-P3). Second, effort spent on metadata and effort spent on algorithms are not interchangeable. Below a certain resolution, no content method can contribute much (P5, P6). Finally, all six propositions are derived from one dataset in one product category. They are therefore offered as expectations to be tested, rather than as established results. P1 to P4 follow from how models allocate parameters and should transfer to settings with an unequal interaction distribution. P5 and P6 are statements about information content and should transfer wherever item features are coarse. We would expect all six to matter less in the settings where modern architectures are usually evaluated. Those settings are typically chosen precisely because interaction data is dense on both sides. Table 10 quantifies how far that is from the case here.

Popularity, Diversity, and the Best-Sellers Status Quo

Against the best-seller list a small store would otherwise run, the case for the hybrid rests on coverage and accuracy, and on neither of the two properties usually claimed for hybrid designs. Coverage is the clear gain. The popularity baseline reaches 0.06% of the catalogue, concentrating recommendations on approximately fourteen products regardless of the user. In contrast, the hybrid reaches 54% and places an average of 5.2 product categories in a ten-item list, compared with 4.0 for popularity. Diversity is not a gain. Measured against ItemKNN alone rather than against the baseline, the hybrid is slightly less diverse on every beyond-accuracy metric. Its intra-list diversity is 0.860 versus 0.886 for ItemKNN, and its coverage is 0.545 versus 0.560. What the hybrid layer contributes, therefore, is the small accuracy improvement rather than greater variety. Robustness to items with no interaction history is not a gain either. This is consistent with P6. The churn simulation reported in the Results section shows that, at this weighting, the hybrid places no such products in its lists at all. The content-only model, which does surface these products, is the least diverse system tested, with an intra-list diversity of 0.317. This is consistent with the over-specialisation problem reported in previous research work [25]. Whether the coverage gain over the best-seller baseline changes user behaviour, such as clicks or purchases, will be evaluated through the planned A/B test.

Dropshipping-Specific Conditions

The conditions that motivate this design fall into three groups, and the study can speak to them unequally. The first group is addressed directly by the system. The lack of first-party quality signals is handled by rating-quality filtering, which is applied across all

recommendation components. The 25-second rebuild also limits how long the model relies on outdated information.

The second group covers conditions the system is designed for but that cannot be demonstrated here. Supplier metadata inconsistency is absent because the product information arrives in a single, standardised format. In a real deployment, this would require attribute normalisation of the kind described in [33]. We therefore identify it as a deployment requirement rather than as a component evaluated in this study. Catalogue turnover is the more serious case because the content component was included partly to address it. In principle, that component can score a product with no interaction history. However, this dataset offers almost no opportunity to observe it doing so. Only 4 of the 23,838 items (0.02%) have no positive interactions, accounting for just 29 of 45,279 held-out purchases. The churn simulation creates this condition artificially. At the recommended weighting, however, the hybrid places no such products in its lists. Whether the fallback has value in a catalogue with frequent new arrivals therefore remains untested. This is another way in which the public dataset differs from dropshipping conditions (gap G2). The third group consists of conditions the dataset cannot represent at all. These include supplier changes, inventory fluctuations, fulfilment delays, duplicate product listings across suppliers, and customer abandonment behaviour. To our knowledge, no publicly available dataset includes these dynamics. Capturing them through live system logs will be part of the future data collection phase. Their absence represents the main limitation of this study in terms of how closely the evaluation reflects real-world dropshipping conditions.

We measured how far the dataset departs from these conditions rather than only describing the difference. Dividing the fourteen and a half years of interactions into eight windows with equal interaction volume, an average window introduces 7.2% new products and retires 2.4%, while the newly introduced products account for 6.1% of that window's interactions. In the final window, which spans 88 days, only 0.6% of active products are new, and they account for 3.2% of interactions and 3.6% of positive ratings. The median product stays in the catalogue for 1,186 days, with an interquartile range of 712 to 1,926 days, and only 0.7% of products have a lifespan shorter than 90 days. Demand is only moderately concentrated, with a Gini coefficient of 0.497 for product popularity, and the most popular 1% of products receive 12.7% of interactions. This is a slow-moving catalogue. A dropshipping catalogue of the kind described in the Introduction turns over considerably faster, so the dataset understates exactly the condition that motivates the content component. The near-absence of cold items is therefore a property of this catalogue rather than of the intended deployment setting.

Limitations

This study has several limitations.

1. All results are based on offline experiments. The effect of the recommendation system on real user behaviour, such as click-through rate, engagement, or conversion, has not yet been measured.

2. The evaluation uses one dataset from one product category, so no claim in this paper is established for a second domain. Two kinds of generalisation should be distinguished. The measured quantities, weights, thresholds, and effect sizes are properties of this catalogue and should not be transferred. P4 predicts that the weights, in particular, will not hold even for this store as it accumulates more data. The propositions are intended to transfer, but only as expectations under the stated conditions. None of them is established for catalogues that are dense on both margins. In such settings, they predict only that the effects they describe become small.
3. A replication on a second category with a different density asymmetry would be the cheapest decisive test of P1 to P3. It is also the test we would run first.
4. The rebuilt database uses placeholder prices, which limits the realism of the price-based filtering component.
5. The content-based model has only a product type and a URL slug to work with, and the slug does not identify the product it is attached to. Additional attributes, such as ingredient information, may improve content-based recommendations. Therefore, the rejection of H2 applies only to the metadata available in this study.
6. Ratings were converted into binary preferences using a threshold of ≥ 4 , which removes some information about the strength of user preferences.
7. Sequential or contextual recommendation methods were not evaluated because the dataset does not contain session-level information.
8. The dataset contains almost no cold items (4 of 23,838 have no positive interaction), so the fallback cannot be observed under the natural operating conditions. The churn simulation reported in the Results section creates this condition instead and finds that the content component surfaces cold products but does not rank them accurately. At the recommended weighting, the hybrid does not surface any of them because the collaborative component assigns them a score of zero and carries most of the weight. The cold-start benefit of the content weight is therefore not supported by evidence in this study, and we treat it as an open question rather than a design advantage.
9. The advantage of the hybrid over ItemKNN alone is not consistent across the evaluation population. For users with at least ten training interactions, the ordering reverses in favour of ItemKNN. The hybrid should therefore be understood as better suited to stores where users have very short histories, rather than as generally preferable.

Future Work

Several directions follow from this study. First, a live A/B evaluation could measure whether the recommendation system improves user behaviour, such as click-through rate and add-to-cart rate, compared with a best-seller baseline. The current implementation is already prepared for this type of evaluation.

Second, future studies could examine richer content representations using additional product information, such as ingredients and review text, to determine whether the content-based component improves when more detailed metadata is available. Third, once sufficient session data becomes available, lightweight sequential recommendation methods could be evaluated.

Further replication on a second product domain would help determine whether the weight-selection approach generalises beyond the current dataset. Finally, LLM-assisted normalisation of multi-supplier metadata feeds [31] could be explored as a preprocessing step to improve content quality. In this role, LLMs would support metadata preparation rather than replace the ranking model itself.

CONCLUSION

This study examined whether a small dropshipping store can deploy a recommendation system that outperforms a simple best-seller list while remaining within realistic technical and computational constraints. The results indicate that this is feasible. In the examined dataset, where the median user has only one rating, item-based collaborative filtering provided most of the useful recommendation signal, whereas content and trend components contributed less to ranking accuracy. The content component was nevertheless essential for exposing products with no interaction history. A churn simulation showed that these products were not ranked reliably and, under the recommended weighting, were not surfaced at all. Cold-start robustness therefore remains an open problem rather than an established property of the proposed design.

Replacing the initial collaborative heuristic with item-based collaborative filtering increased NDCG@10 by approximately fourteen-fold. The final hybrid model, with weights of 10/70/20, outperformed the individual components and tuned baselines, although the absolute improvement was operationally modest at the observed interaction density and diminished for users with longer interaction histories.

The complete system generated recommendations with a median latency of 34 ms, required approximately 25 seconds for an offline rebuild, and achieved 54% catalogue coverage, compared with 0.06% for the best-seller baseline. These results indicate that a simple neighborhood-based model can be an effective choice under severe sparsity and limited computational resources. In this setting, it achieved higher accuracy than latent-factor models while requiring substantially less training time. More broadly, the findings suggest that recommendation performance depends strongly on where sufficient evidence exists within the interaction matrix rather than solely on model sophistication. The advantage of the neighborhood model over latent-factor approaches increased with the number of interactions per product and disappeared at approximately four interactions.

These observations are formulated as six propositions with explicit boundary conditions rather than as universally generalizable findings, because they are derived from a single catalogue within one product category. The propositions, rather than the specific

model weights, represent the primary candidates for transfer to other settings. The conditions under which these propositions may fail are also explicitly identified.

Finally, the findings are based exclusively on offline evaluation. Future work should therefore examine whether the observed improvements translate into meaningful changes in user behavior, engagement, and conversion through controlled live A/B testing.

NOMENCLATURE

API	Application Programming Interface
ASIN	Amazon Standard Identification Number
BPR	Bayesian Personalised Ranking
CF	Collaborative Filtering
F1	Harmonic mean of precision and recall
GPU	Graphics Processing Unit
HR@k	Hit Ratio at cut-off k
IDF	Inverse Document Frequency
ILD	Intra-List Diversity
ItemKNN	Item-based k-Nearest-Neighbour collaborative filtering
MAP@k	Mean Average Precision at cut-off k
NDCG@k	Normalised Discounted Cumulative Gain at cut-off k
p95	95th percentile
REST	Representational State Transfer
RSS	Resident Set Size (process memory)
SVD	Singular Value Decomposition
TF	Term Frequency
TF-IDF	Term Frequency – Inverse Document Frequency
u, i, j	Index of a user (u) and of items (i, j)
R	User-item rating matrix, $r_{ui} \in \{1, \dots, 5\}$
ρ	Positive-feedback threshold, $\rho = 4.0$
k	Cut-off for the evaluation metrics, $k = 10$; also the neighbourhood size in Algorithm 1, $k = 200$
p	Wilcoxon p-value; also the reference product in Algorithm 2
P	Binary matrix of positive interactions, $P = \{(u, i) : r_{ui} \geq \rho\}$
P^T	Transpose of P
A, B	Generic vectors compared by cosine similarity in Eq. (2)
C, CF, T	Candidate sets produced by the content, collaborative and trend components
X	A component in $\{C, CF, T\}$

t, d, D	Term, document (product text), and document corpus
$df(t)$	Number of documents containing term t
$TF(t,d)$	Relative frequency of term t in document d
$IDF(t,D)$	$\ln(N / df(t)); N = D $, $df(t)$ = documents containing t
n	Requested recommendation list length; each component contributes a pool of $3n$ candidates
$TopK$	Operator returning the k highest-scoring entries
$sim(i,j)$	Cosine similarity between products i and j
θ_c	Content cosine-similarity threshold, $\theta_c = 0.2$
$s_c(u,i)$	Content score: max over $j \in I_{u+}$ of $sim_{TFIDF}(i,j)$
$s_{cf}(u,i)$	Collaborative score: sum of $sim_{cos}(i,j)$ over the user's training positives lying in the top- k neighbourhood of item i in Eq. (3)
$s_T(i)$	Trend score: $review_count(i) \times avg_rating(i)$; eligibility ≥ 10 reviews and average rating > 4.0
S, S_i	Item-item cosine similarity table; S_i is the top- k neighbour row for item i
$N_k(i)$	Set of the k items most similar to item i under S ; the summation in Eq. (3) is restricted to this set ($k = 200$)
I_{u+}	Items user u rated $\geq \rho$ in the training period
$\tilde{s}_X$	Per-user min-max normalisation of a component score over candidate items
w_C, w_{CF}, w_T	Hybrid mixing weights (recommended 0.1/0.7/0.2), $w_C + w_{CF} + w_T = 1$
$score(u,i)$	$score(u,i) = w_C \tilde{s}_c(u,i) + w_{CF} \tilde{s}_{cf}(u,i) + w_T \tilde{s}_T(i)$
f	Number of latent factors (PureSVD $f = 256$; BPR $f = 128$)
Δ	Difference in a metric between two systems, e.g. $\Delta NDCG@10$
τ	Kendall's rank correlation between two orderings of the compared systems

AUTHOR CONTRIBUTIONS

Conceptualisation, M.S., and Z.S.K.; Investigation, Z.S.K., and E.H.; Methodology, M.S., Z.S.K., and E.H.; Data Curation, Z.S.K.; Software, M.S., and Z.S.K.; Validation, M.S., and Z.S.K.; Formal Analysis, M.S., and Z.S.K.; Writing – Original Draft, M.S., and Z.S.K.; Writing – Review & Editing, M.S., Z.S.K., and E.H.; Supervision, M.S.

DATA AVAILABILITY STATEMENT

The dataset is publicly available on Kaggle [39] (accessed 8 July 2026). All code (the offline evaluation pipeline, the artefact build scripts and the application) is available at <https://github.com/ZShala/dropshipping-hybrid-recommender/>. Experiments used

Python 3.10 with scikit-learn, SciPy and implicit 0.7; random seeds 0–4; hyperparameters as reported in the Hyperparameter Selection section.

ACKNOWLEDGMENT

The authors thank the academic staff of the Faculty of Computer Science and Engineering in the University for Business and Technology, for their guidance and support throughout this research.

CONFLICT OF INTERESTS

The authors declare that there is no conflict of interest associated with this publication.

REFERENCES

1. Arora, N., et al. The Value of Getting Personalization Right—or Wrong—is Multiplying. McKinsey & Company, November 12, 2021. Available online <https://www.mckinsey.com/capabilities/growth-marketing-and-sales/our-insights/the-value-of-getting-personalization-right-or-wrong-is-multiplying> (accessed on 8 May 2026).
2. Retail Customer Experience. Personalization Influencing Most Online Shoppers. August 22, 2022. Available online <https://www.retailcustomerexperience.com/news/personalization-influencing-most-online-shoppers/> (accessed on 8 May 2026).
3. Guo, M., Yan, N., Cui, X., Wu, S. H., Ahsan, U., West, R., Al Jadda, K. Deep Learning-Based Online Alternative Product Recommendations at Scale. *Proc. 3rd Workshop on e-Commerce and NLP (ECNLP 3)*, Seattle, WA, USA. **2020**, pp. 19–23.
4. Lee, B. How AI Product Recommendation Engines Are Transforming E-Commerce Revenue. Doba Blog, June 8, 2025. Available online <https://www.doba.com/blog/marketing-and-sales-growth/conversion-strategies/how-ai-product-recommendation-engines-are-transforming-e-commerce-revenue-37861> (accessed on 8 April 2026).
5. Sarwar, B., Karypis, G., Konstan, J., Riedl, J. Item-Based Collaborative Filtering Recommendation Algorithms. *Proc. 10th Int. Conf. World Wide Web (WWW '01)* **2001**, pp. 285–295.
6. Linden, G., Smith, B., York, J. Amazon.com Recommendations: Item-to-Item Collaborative Filtering. *IEEE Internet Comput.* **2003**, 7(1), 76–80.
7. He, X., Liao, L., Zhang, H., Nie, L., Hu, X., Chua, T.-S. Neural Collaborative Filtering. *Proc. 26th Int. Conf. World Wide Web (WWW '17)* **2017**, pp. 173–182.
8. Zhang, S., Yao, L., Sun, A., Tay, Y. Deep Learning Based Recommender System: A Survey and New Perspectives. *ACM Comput. Surv.* **2019**, 52(1), 5.
9. He, X., Deng, K., Wang, X., Li, Y., Zhang, Y., Wang, M. LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation. *Proc. 43rd Int. ACM SIGIR Conf. Research and Development in Information Retrieval (SIGIR '20)* **2020**, pp. 639–648.
10. Gao, C., et al. A Survey of Graph Neural Networks for Recommender Systems. *ACM Trans. Recomm. Syst.* **2023**, 1(1), 1–51.
11. Kang, W.-C., McAuley, J. Self-Attentive Sequential Recommendation. *Proc. IEEE Int. Conf. Data Mining (ICDM)* **2018**, pp. 197–206.

12. Sun, F., et al. BERT4Rec: Sequential Recommendation with Bidirectional Encoder Representations from Transformer. *Proc. 28th ACM Int. Conf. Information and Knowledge Management (CIKM '19)* **2019**, pp. 1441–1450.
13. Burke, R. Hybrid Recommender Systems: Survey and Experiments. *User Model. User-Adapt. Interact.* **2002**, 12(4), 331–370.
14. Bodduluri, K.C., et al. Exploring the Landscape of Hybrid Recommendation Systems in E-Commerce. *IEEE Access* **2024**, 12, 28273–28296.
15. Ferrari Dacrema, M., Cremonesi, P., Jannach, D. Are We Really Making Much Progress? A Worrying Analysis of Recent Neural Recommendation Approaches. *Proc. 13th ACM Conf. Recommender Systems (RecSys '19)* **2019**, pp. 101–109.
16. Rendle, S., Krichene, W., Zhang, L., Anderson, J. Neural Collaborative Filtering vs. Matrix Factorization Revisited. *Proc. 14th ACM Conf. Recommender Systems (RecSys '20)* **2020**, pp. 240–248.
17. Arunasalam, A., et al. The Dark Side of E-Commerce: Dropshipping Abuse as a Business Model. *Proc. Network and Distributed System Security Symposium (NDSS)* San Diego, CA, USA, **2024**, pp. 1-18.
18. Goldberg, D., Nichols, D., Oki, B. M., Terry, D. Using Collaborative Filtering to Weave an Information Tapestry. *Commun. ACM* **1992**, 35(12), 61–70.
19. Adomavicius, G., Tuzhilin, A. Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. *IEEE Trans. Knowl. Data Eng.* **2005**, 17(6), 734–749.
20. Su, X., Khoshgoftaar, T. M. A Survey of Collaborative Filtering Techniques. *Adv. Artif. Intell.* **2009**, 2009, 421425.
21. Koren, Y., Bell, R., Volinsky, C. Matrix Factorization Techniques for Recommender Systems. *Computer* **2009**, 42(8), 30–37.
22. Rendle, S., Freudenthaler, C., Gantner, Z., Schmidt-Thieme, L. BPR: Bayesian Personalized Ranking from Implicit Feedback. *Proc. 25th Conf. Uncertainty in Artificial Intelligence (UAI)* **2009**, pp. 452–461.
23. Pazzani, M., Billsus, D. Learning and Revising User Profiles: The Identification of Predictive Features. *Mach. Learn.* **1997**, 27 (3), 313–331. <https://doi.org/10.1023/A:1007369909943>.
24. Chhipa, S., Berwal, V., Hirapure, T., Banerjee, S. Recipe Recommendation System Using TF-IDF. *ITM Web Conf.* **2022**, 44, 02006.
25. Jannach, D., Zanker, M., Felfernig, A., Friedrich, G. *Recommender Systems: An Introduction*; Cambridge University Press: Cambridge, **2010**.
26. Schein, A.I., Popescul, A., Ungar, L.H., Pennock, D.M. Methods and Metrics for Cold-Start Recommendations. *Proc. 25th Annual Int. ACM SIGIR Conf. Research and Development in Information Retrieval (SIGIR '02)* **2002**, pp. 253–260.
27. Wu, L., He, X., Wang, X., Zhang, K., Wang, M. A Survey on Accuracy-Oriented Neural Recommendation. *IEEE Trans. Knowl. Data Eng.* **2023**, 35(5), 4425–4445.
28. de Souza Pereira Moreira, G., et al. Transformers4Rec: Bridging the Gap between NLP and Sequential/Session-Based Recommendation. *Proc. 15th ACM Conf. Recommender Systems (RecSys '21)* **2021**, pp. 143–153.

29. Boka, T.F., Niu, Z., Neupane, R.B. A Survey of Sequential Recommendation Systems. *Inf. Syst.* **2024**, 125, 102427.
30. Yu, J., et al. Self-Supervised Learning for Recommender Systems: A Survey. *IEEE Trans. Knowl. Data Eng.* **2024**, 36(1), 335–355.
31. Wu, L., et al. A Survey on Large Language Models for Recommendation. *World Wide Web* **2024**, 27(5), 60.
32. Gaddam, S.R. Database Performance Optimization: Strategies That Scale. *Int. J. Comput. Eng.* **2025**, 7(11), 1–12.
33. Mishra, R.S., Mehta, K., Rasiwasia, N. Scalable Approach for Normalizing E-Commerce Text Attributes (SANTA). *Proc. 4th Workshop on e-Commerce and NLP (ECNLP 4)* **2021**, pp. 101–110.
34. Chen, H., Chiang, R. H. L., Storey, V. C. Business Intelligence and Analytics: From Big Data to Big Impact. *MIS Q.* **2012**, 36(4), 1165–1188.
35. Davenport, T.H., Harris, J.G. *Competing on Analytics: The New Science of Winning*; Harvard Business School Press: Boston, MA, **2007**.
36. Gartner. Predicts 2024: AI Powers Personalization, Revenue Growth and Efficiency in Consumer Goods. Gartner, 2024. Available online <https://www.gartner.com/en/documents/5285163> (accessed on 8 April 2026).
37. Berisha, F., Bytyçi, E. Addressing Cold Start in Recommender Systems with Neural Networks. *Int. J. Comput. Appl.* **2023**, 45(7–8), 485–496.
38. Deldjoo, Y., et al. A Review of Modern Fashion Recommender Systems. *ACM Comput. Surv.* **2023**, 56(4), Article 87.
39. Satra, P. Amazon Beauty Products [Dataset]. Kaggle, 2023. Available online <https://www.kaggle.com/datasets/satrapankti/amazon-beauty-product-recommendation> (accessed on 8 April 2026).
40. Krichene, W., Rendle, S. On Sampled Metrics for Item Recommendation. *Proc. 26th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining (KDD '20)* **2020**, pp. 1748–1757.
41. Herlocker, J. L., Konstan, J. A., Terveen, L. G., Riedl, J. T. Evaluating Collaborative Filtering Recommender Systems. *ACM Trans. Inf. Syst.* **2004**, 22(1), 5–53.
42. Sun, Z., Yu, D., Fang, H., Yang, J., Qu, X., Zhang, J., Geng, C. Are We Evaluating Rigorously? Benchmarking Recommendation for Reproducible Evaluation and Fair Comparison. *Proc. 14th ACM Conf. Recommender Systems (RecSys '20)* **2020**, pp. 23–32.
43. Deshpande, M., Karypis, G. Item-Based Top-N Recommendation Algorithms. *ACM Trans. Inf. Syst.* **2004**, 22(1), 143–177.
44. Ning, X., Karypis, G. SLIM: Sparse Linear Methods for Top-N Recommender Systems. *Proc. IEEE Int. Conf. Data Mining (ICDM)* **2011**, pp. 497–506.
45. Steck, H. Embarrassingly Shallow Autoencoders for Sparse Data. *Proc. World Wide Web Conf. (WWW '19)* **2019**, pp. 3251–3257.
46. Cremonesi, P., Koren, Y., Turrin, R. Performance of Recommender Algorithms on Top-N Recommendation Tasks. *Proc. 4th ACM Conf. Recommender Systems (RecSys '10)* **2010**, pp. 39–46.
47. Steck, H. Item Popularity and Recommendation Accuracy. *Proc. 5th ACM Conf. Recommender Systems (RecSys '11)* **2011**, pp. 125–132.