

Online Appendix: Impact of Data Privacy Regulations on Recommender Systems Performance

Appendix A: Examples of Data Privacy Practices from Online Platforms

Figure A11 BestBuy Provides Consumer Portal for Manually Deleting Past Data

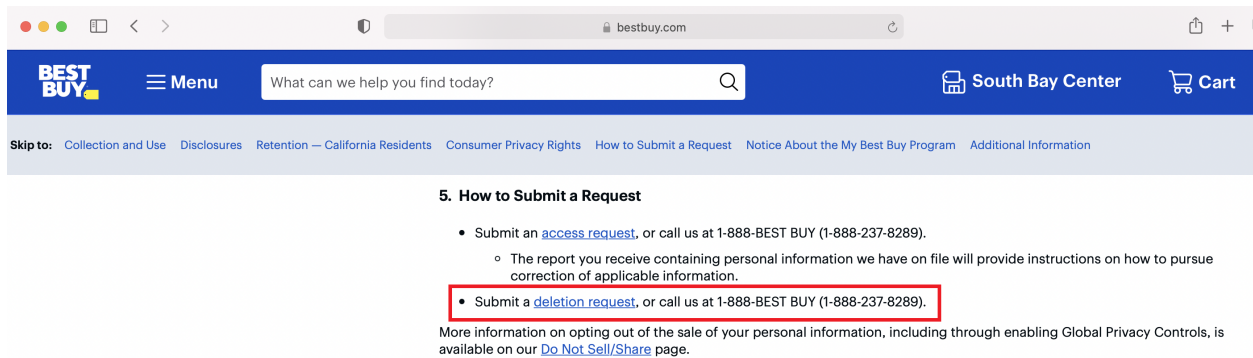


Figure A12 Youtube Provides Consumer Portal for Auto-Deleting Past Data

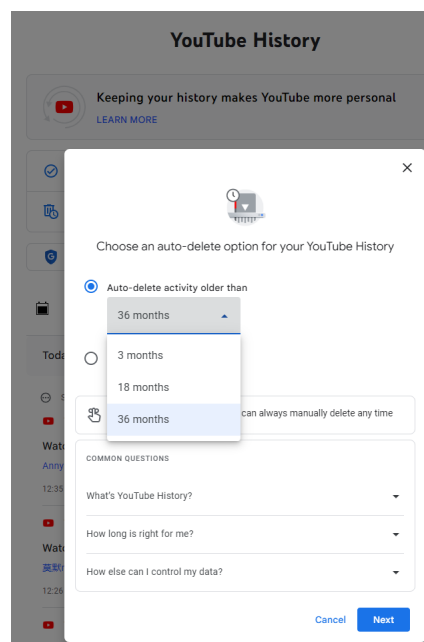
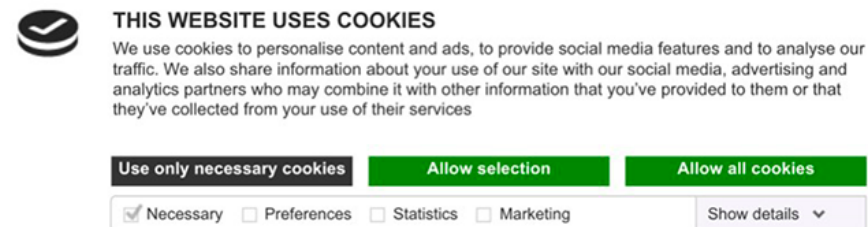


Figure A13 Cookie Consent Banner



Appendix B: Main Simulation Components and Notation

Table B4 Key Notation for the Simulation Framework

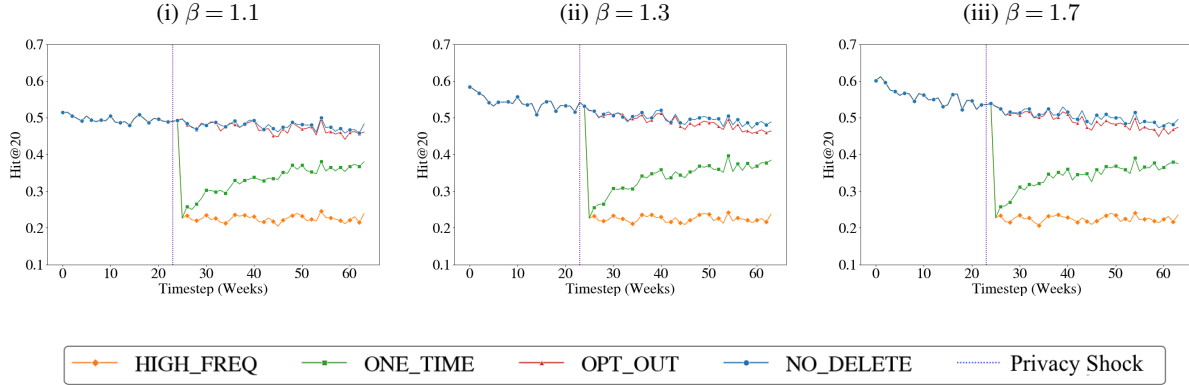
Aspect	Notations and Explanations
A. Item population	
Item set	I : elements of the set are individual items indexed by i ($i \in I$).
Item content	e_i : A latent vector of item i 's ($i \in I$) content features
B. User population	
User set	U : elements of the set are individual users indexed by u ($u \in U$).
User preferences	e_u or $\hat{e}_u$: latent vector that represents user u 's ($u \in U$) stable or dynamic preference profile.
Degree of user preference change	α : the degree of change in user preference profile, with 0 indicating original (unchanged) profile and 1 indicating a complete change into a different profile.
User needs	$PrefList_u$: User u 's list of all items ranked by the inherent user preferences.
User reliance on recommendations	w : the probability of user's consumption from $RecList$ during a given visit; correspondingly, $(1 - w)$ indicates user's consumption probability from $PrefList_u$.
Recommendation list exploration	β : controls how far down the recommendation list users are likely to explore before choosing an item.
User consumption frequency	$\Delta_{u,t}$: representation of length of time between user u 's consumption at t and the next consumption. Also, next scheduled next consumption for user u is referred to as $UserNextCons_u$ value.
User consumption data	$ConsData_u^t$: all historical consumption data of user u up to (and including at) time t .
User privacy request	$DataRequest_u$: the enactment of a certain data privacy practice by the user u ; for example, delete all personal data under ONE_TIME.
C. Recommender engine	
Recommendation generation	$GenerateRecs$: recommendation generator(s) used to train the RS and produce a list of recommendations for each user for the next time period.
Performance measurement	$Evaluate$: performance metrics, for measuring different aspects of recommender systems performance, for example, accuracy, diversity, etc.
Training Data	$TrainData_u^t$ represents user-item consumption history that is made available to RS, which is often a subset of all consumption data $ConsData_u^t$ due to privacy practices.
System Recommendations	$RecList_u^t$ represents the list of recommended items ranked by the system's predicted preferences of user u at time t
D. Scenario Change	
Privacy Shock	$T_{DataReq}$ is the time when users can start requesting data privacy actions.
Preference Transition	$T_{PrefChange}$ is the time when users' preferences change.

Appendix C: Recommendation List Exploration

As discussed in Section 4.1, parameter β controls the extent to which each user explores the recommended item list before making a selection for item consumption. We vary the value of β with 1.1, 1.3 (the default choice), and 1.7, and the results under stable user preferences are presented in Figure C14. These three values of β approximately represent scenarios where the user consumes from top-20, top-10, and top-5 recommendations, respectively; for example, the top-1 item has 10.7%, 23.1%, and 41.2% chance of being consumed with these three values of β , respectively. Notably, the value of β does not affect the key qualitative patterns of main results, while the overall RS performance increases for larger β values. Since larger β s concentrate the probability mass more towards the top-ranked items, the users' consumption

behavior becomes more predictable and thus the system performance is improved. The consistency of main results with different β values verifies the robustness of the simulation framework and of the key insights.

Figure C14 Recommendation List Exploration, Static User Preferences, Affected Users, Hit@20, BPR



Note that across all figures, 50% of users are affected and all users' reliance on recommendations is 20%.

Appendix D: Main Experiment Results with Additional Accuracy Metrics: MRR@20 and NDCG@20

Mean Reciprocal Rank of top- K recommended items (MRR@K) evaluates the ranking quality by considering the position of the first relevant item in the recommendation list. It is defined as:

$$\text{MRR@K} = \frac{1}{|U|} \sum_{u \in U} \frac{1}{\text{rank}_u} \quad (7)$$

where rank_u is the position of the first relevant item in user u 's recommendation list RecList_u^t within the top- K items. If no relevant items are found within top- K , rank_u is set to infinity, resulting in a reciprocal rank of zero.

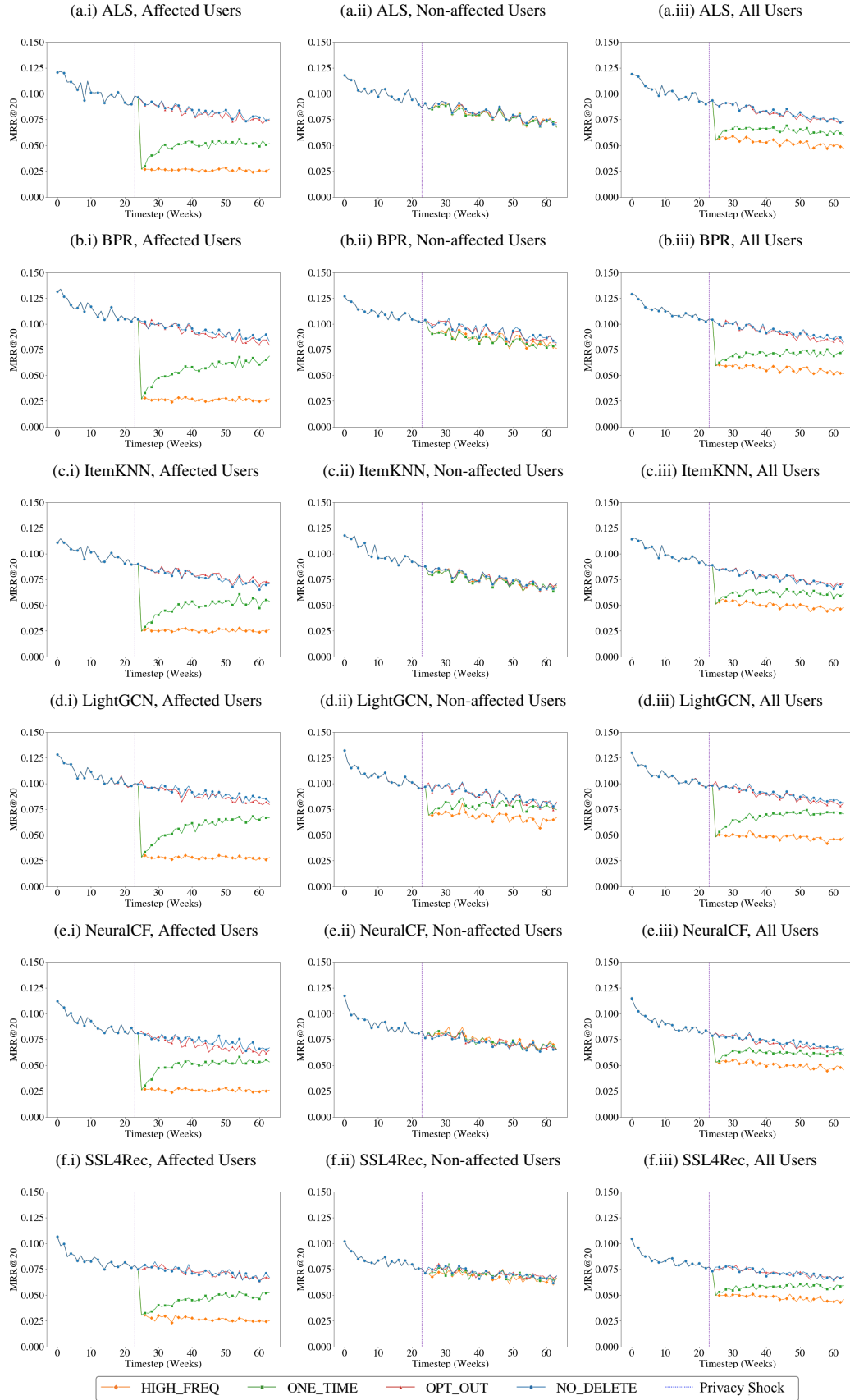
Normalized Discounted Cumulative Gain of top- K recommended items (NDCG@K) measures ranking quality by considering both the position and relevance of items in the recommendation list:

$$\text{NDCG@K} = \frac{1}{|U|} \sum_{u \in U} \frac{\sum_{r=1}^K \frac{\text{rel}_{u,r}}{\log_2(r+1)}}{\sum_{r=1}^K \frac{1}{\log_2(r+1)}} \quad (8)$$

where $\text{rel}_{u,r} = 1$ if the item at rank r in RecList_u^t is relevant for user u , and 0 otherwise. The denominator represents the ideal score when all relevant items are ranked at the top.

These metrics provide insights into the accuracy of RS in ranking relevant items at the top positions. We provide main experiment results in terms of MRR@20 and NDCG@20 below. They are consistent with the results (in terms of Hit@20) presented in the main paper.

Figure D15 Impact of Data Privacy Practices with Stable User Preferences, MRR@20



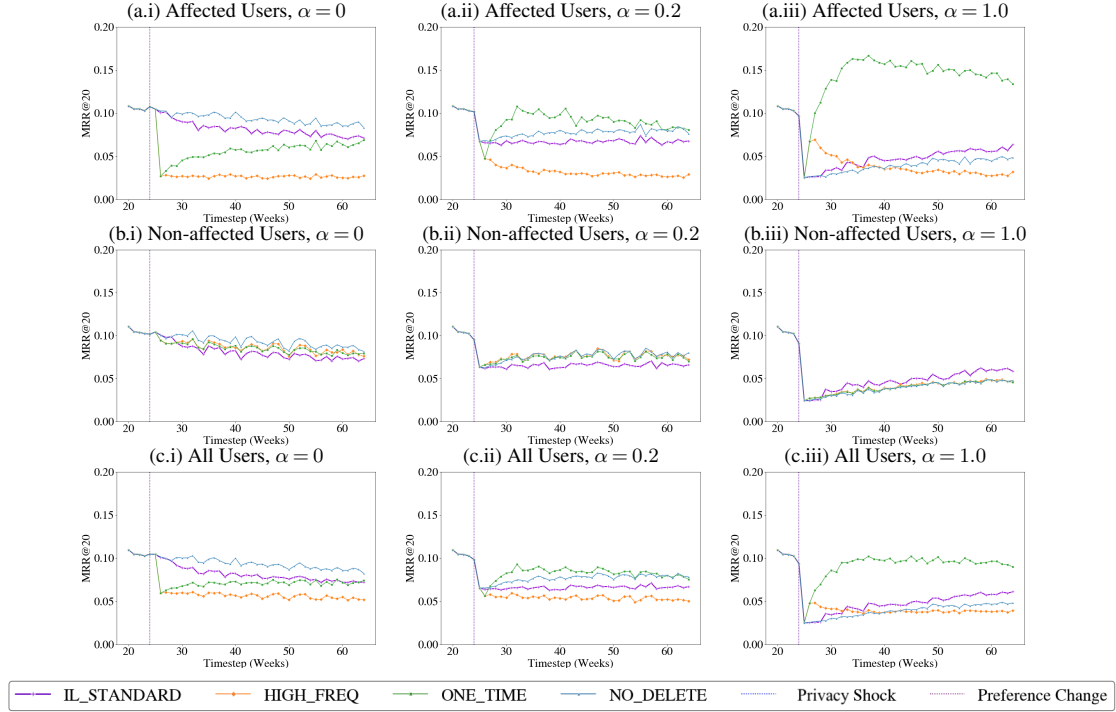
Note that across all figures, 50% of users are affected, and all users' reliance on recommendations is 20%.

Figure D16 Impact of Data Privacy Practices with Dynamic User Preferences, MRR@20



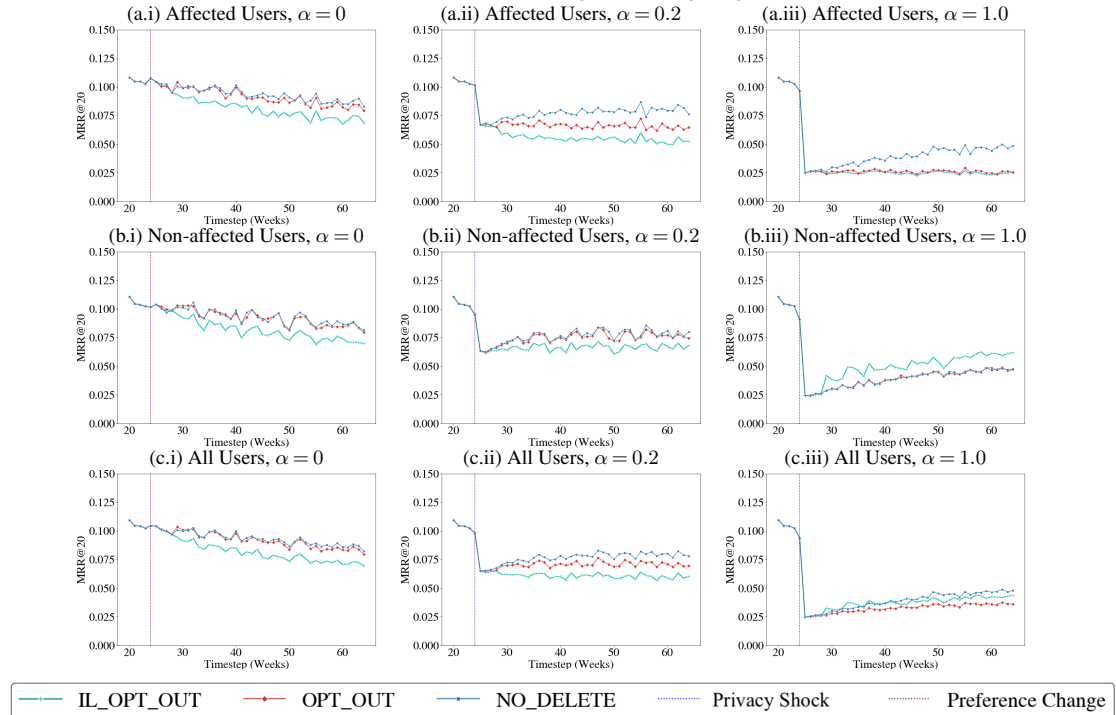
Note that across all figures, 50% of users are affected and all users' reliance on recommendations is 20%.

Figure D17 Incremental and Non-incremental Learning under Deleting Practices, MRR@20



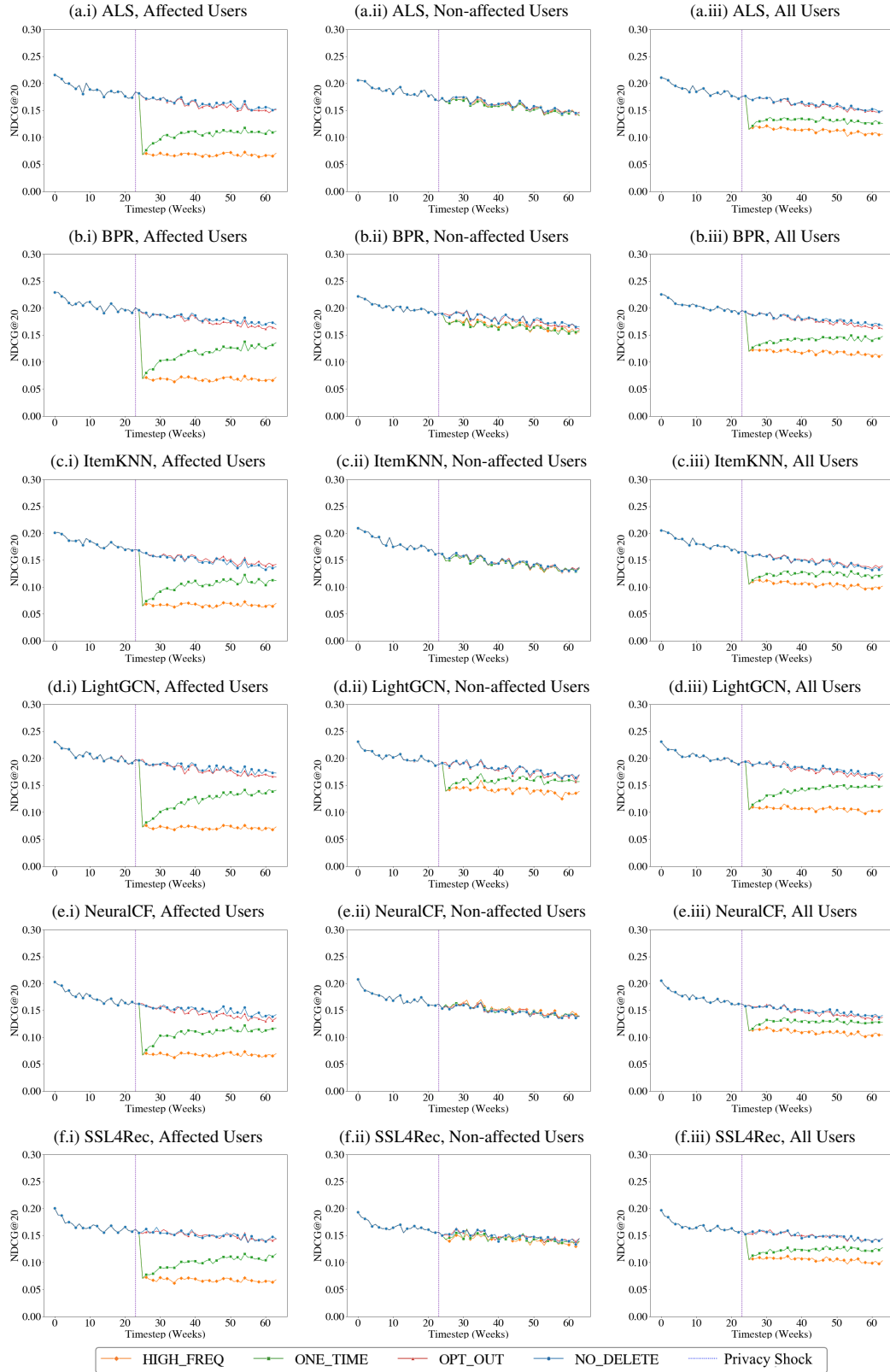
The results are on affected users and the simulation is under a setting where 50% of users are affected, algorithm is BPR, and the user preference changes together with $T_{DataReq}$.

Figure D18 Incremental and Non-incremental Learning under Opting Out Practice, MRR@20



The results are on affected users and the simulation is under a setting where 50% of users are affected, the algorithm is BPR, and the user preference changes together with $T_{DataReq}$.

Figure D19 Impact of Data Privacy Practices with Stable User Preferences, NDCG@20



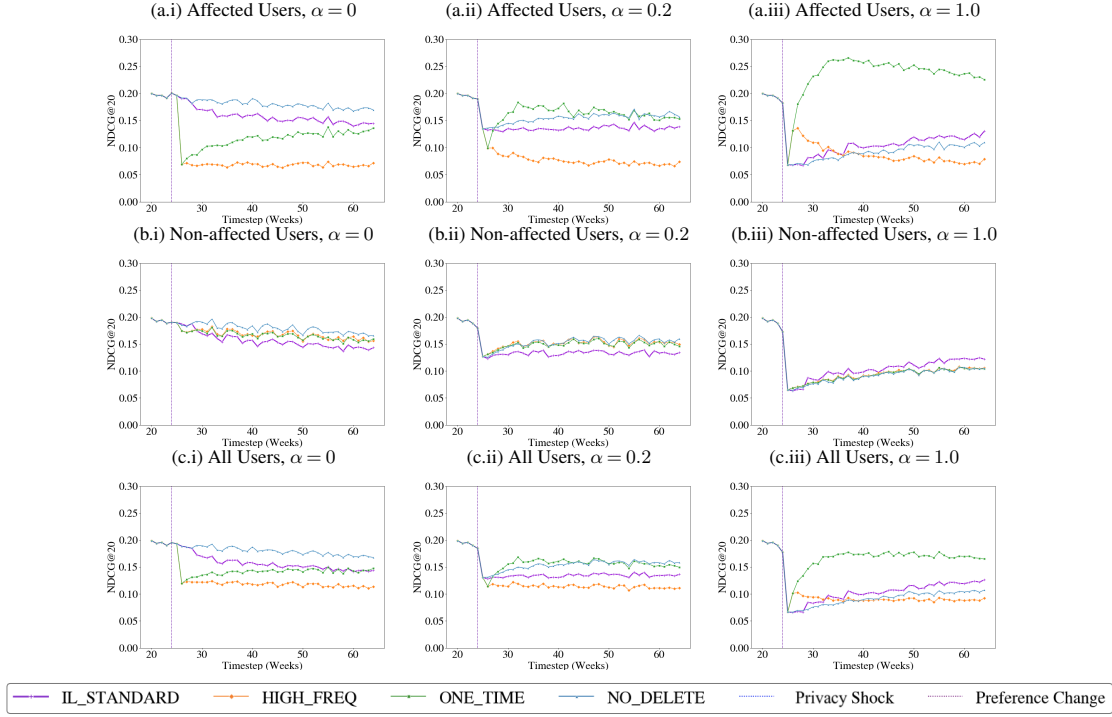
Note that across all figures, 50% of users are affected and all users' reliance on recommendations is 20%.

Figure D20 Impact of Data Privacy Practices with Dynamic User Preferences, NDCG@20



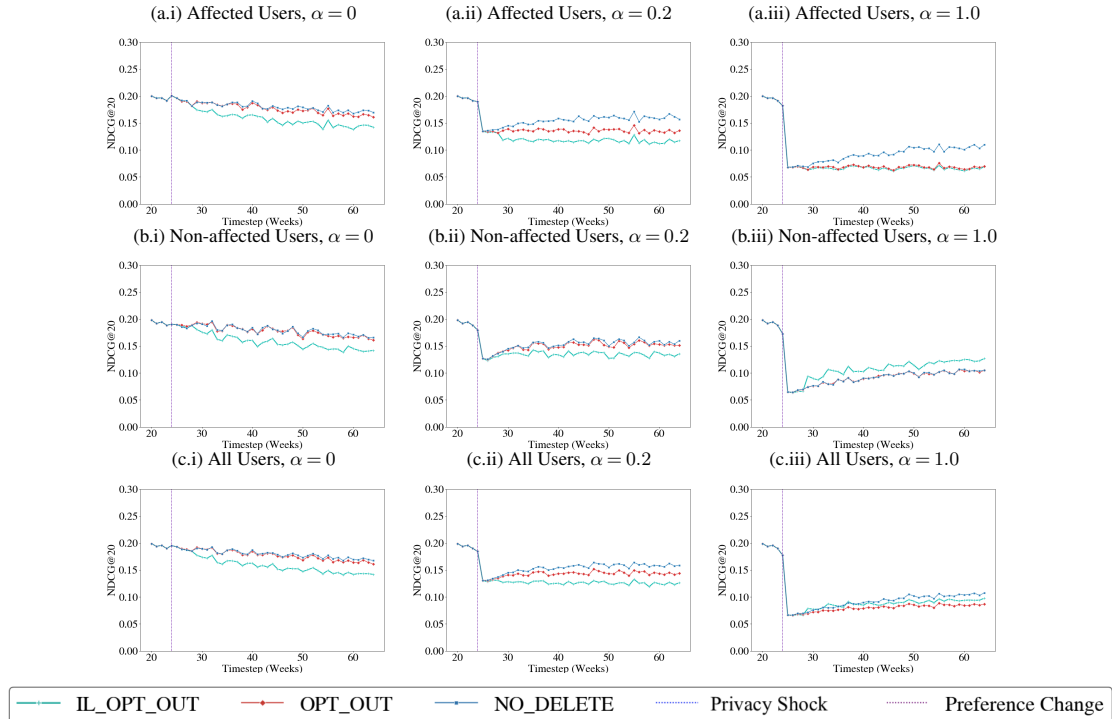
Note that across all figures, 50% of users are affected and all users' reliance on recommendations is 20%.

Figure D21 Incremental and Non-incremental Learning under Deleting Practices, NDCG@20



The results are on affected users and the simulation is under a setting where 50% of users are affected, algorithm is BPR, and the user preference changes together with $T_{DataReq}$.

Figure D22 Incremental and Non-incremental Learning under Opting Out Practice, NDCG@20



The results are on affected users and the simulation is under a setting where 50% of users are affected, the algorithm is BPR, and the user preference changes together with $T_{DataReq}$.

Appendix E: Main Experiment Results with Additional Seed Datasets

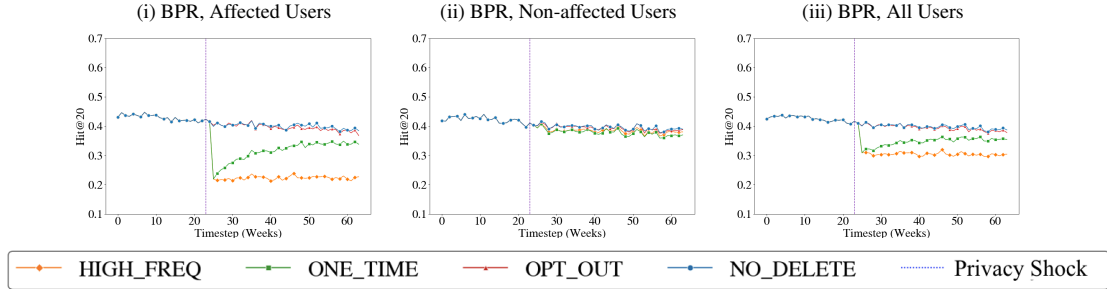
In this section, we present main experiment results with additional seed datasets *Amazon Book* and *Retailrocket*. The summary statistics of all datasets (i.e., the main dataset used in the paper and two additional seed datasets) are shown in Table E5.

Table E5 Summary Statistics of Seed Datasets

Dataset	Domain (Interactions)	Sparsity Level	User/Product Ratio
Multi-Channel Retailer	Retail (Purchases)	1.438×10^{-4}	18,938/40,716 (0.465)
Retailrocket	Retail (Views)	8.079×10^{-6}	1,404,179/234,838 (5.98)
Amazon Book	Books (Purchases)	4.005×10^{-5}	603,668/367,982 (1.64)

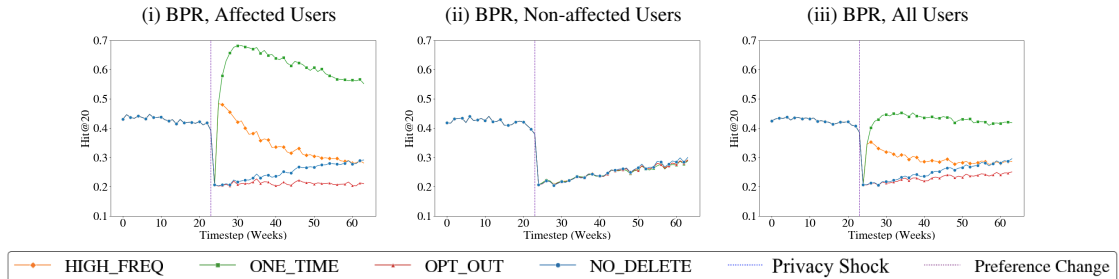
E.1. Main Experiment Results with *Amazon Book*

Figure E23 Impact of Data Privacy Practices with Stable User Preferences, Amazon Book, Hit@20



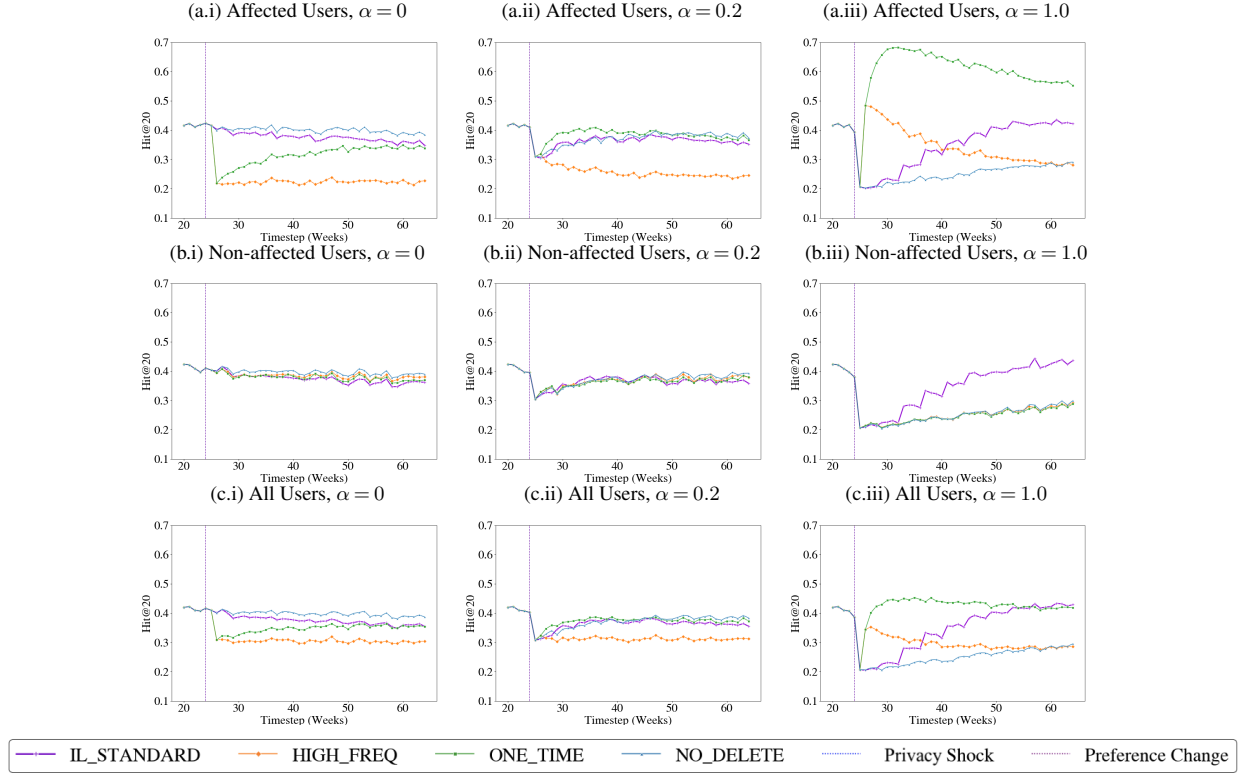
Note that across all figures, 50% of users are affected and all users' reliance on recommendations is 20%.

Figure E24 Impact of Data Privacy Practices with Dynamic User Preferences, Amazon Book, Hit@20



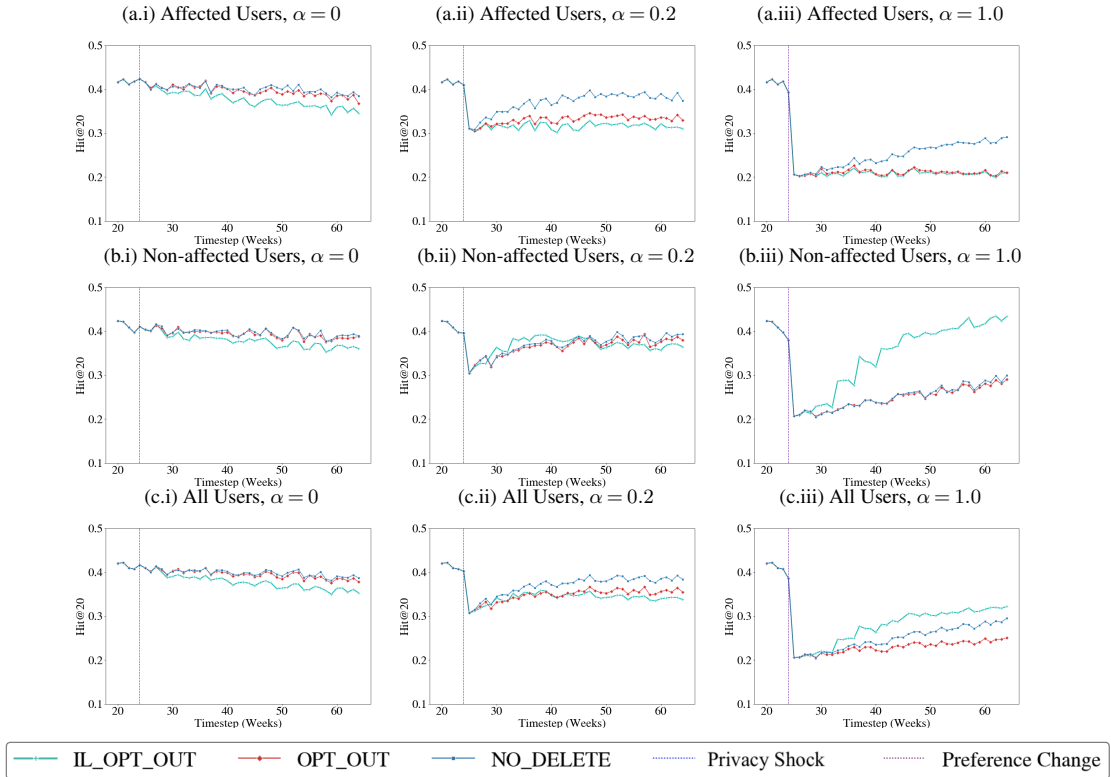
Note that across all figures, 50% of users are affected and all users' reliance on recommendations is 20%.

Figure E25 Incremental and Non-incremental Learning under Deleting Practices, Amazon Book, Hit@20



The results are on affected users and the simulation is under a setting where 50% of users are affected, algorithm is BPR, and the user preference changes together with $T_{DataReq}$.

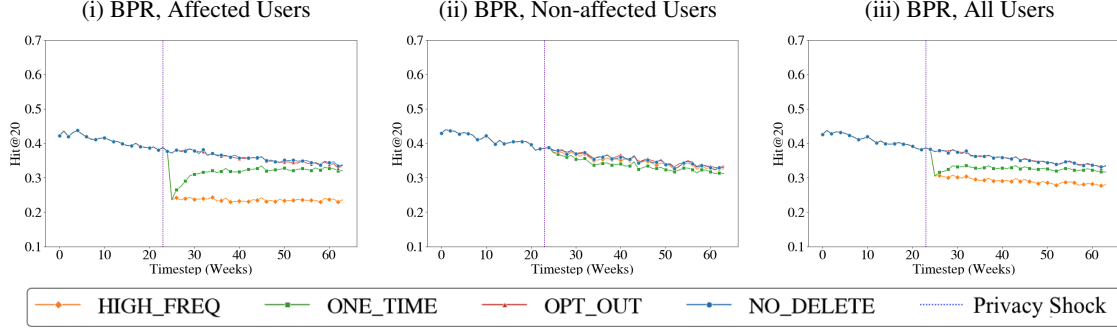
Figure E26 Incremental and Non-incremental Learning under Opting Out Practice, Amazon Book, Hit@20



The results are on affected users and the simulation is under a setting where 50% of users are affected, the algorithm is BPR, and the user preference changes together with $T_{DataReq}$.

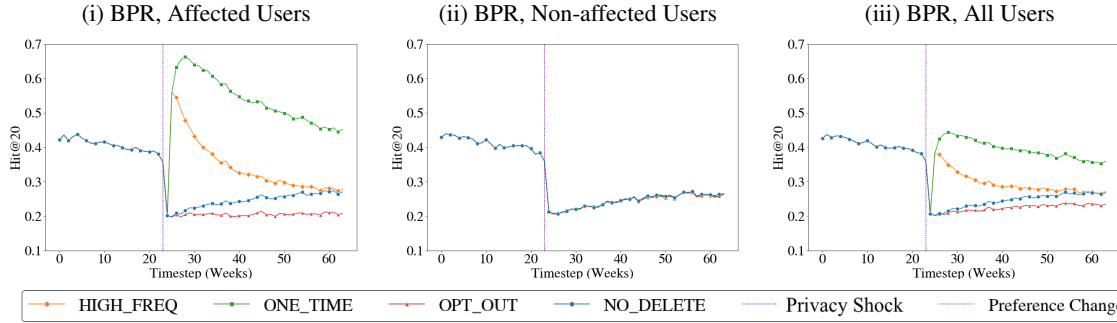
E.2. Main Experiment Results with *Retailrocket*

Figure E27 Impact of Data Privacy Practices with Stable User Preferences, *Retailrocket*, Hit@20



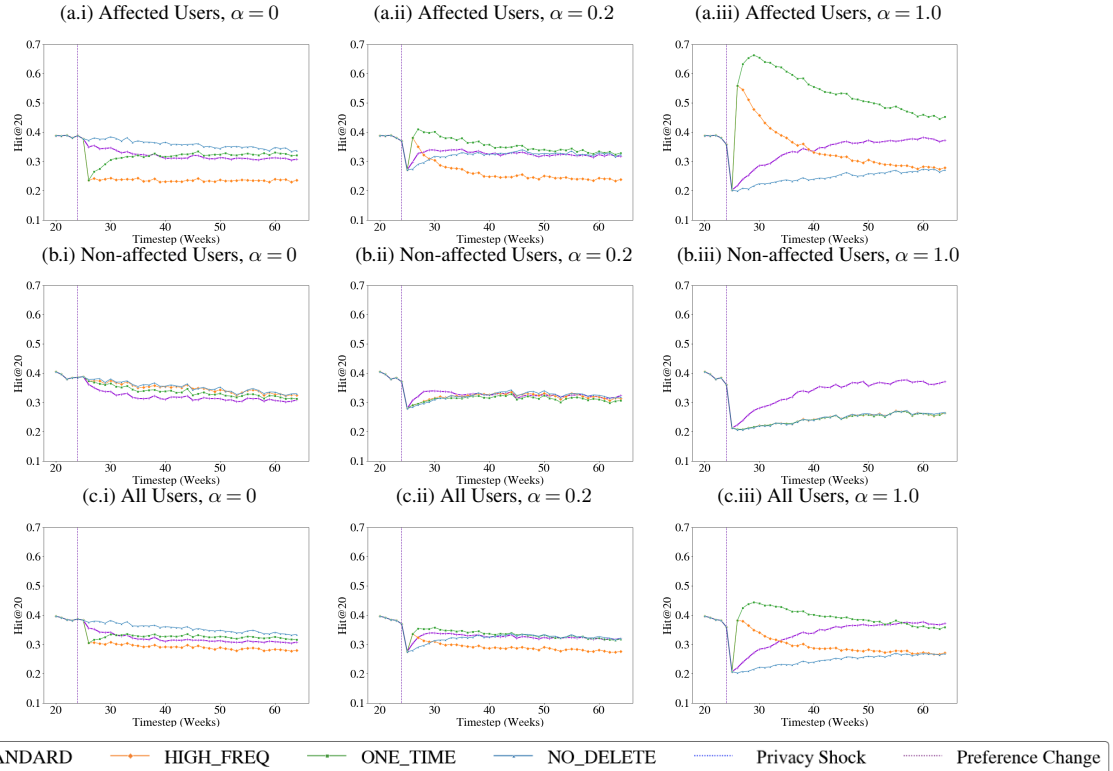
Note that across all figures, 50% of users are affected and all users' reliance on recommendations is 20%.

Figure E28 Impact of Data Privacy Practices with Dynamic User Preferences, *Retailrocket*, Hit@20



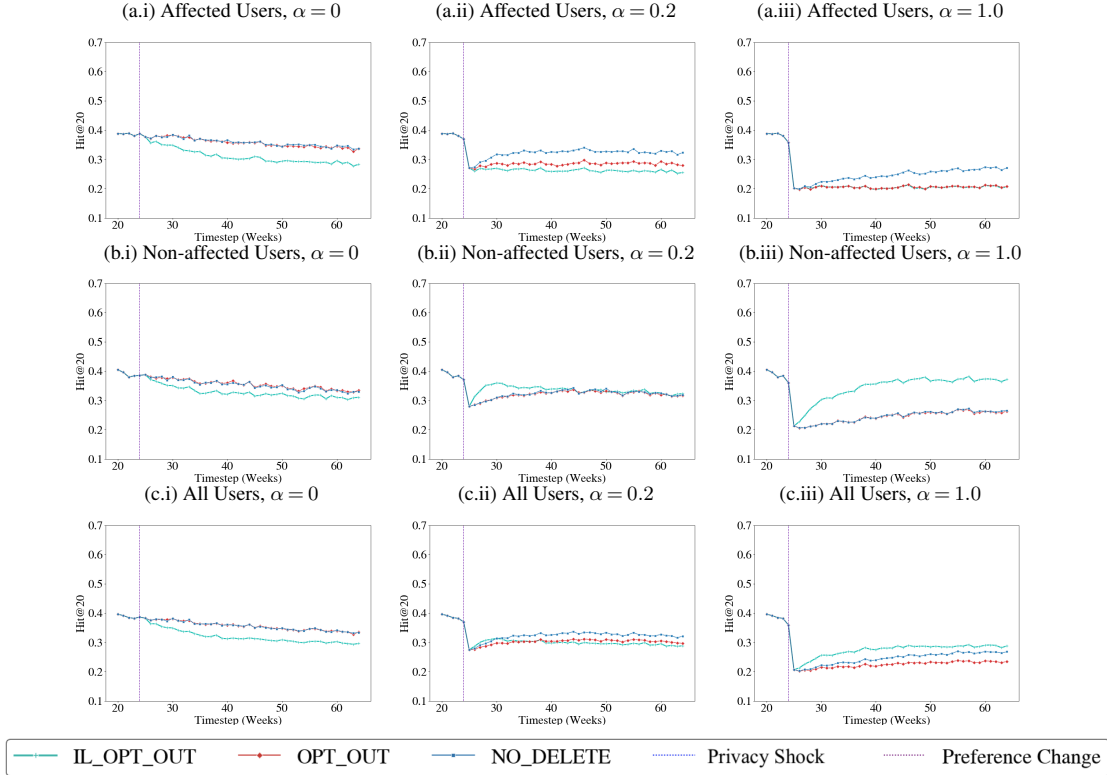
Note that across all figures, 50% of users are affected and all users' reliance on recommendations is 20%.

Figure E29 Incremental and Non-incremental Learning under Deleting Practices, *Retailrocket*, Hit@20



The results are on affected users and the simulation is under a setting where 50% of users are affected, algorithm is BPR, and the user preference changes together with $T_{DataReq}$.

Figure E30 Incremental and Non-incremental Learning under Opting Out Practice, Retailrocket, Hit@20



The results are on affected users and the simulation is under a setting where 50% of users are affected, the algorithm is BPR, and the user preference changes together with $T_{DataReq}$.

Appendix F: Benchmarking/Validation Results for Recommender System Algorithms

To ensure validity and correctness of our simulation framework, we benchmarked our implementations of the recommender system (RS) algorithms, specifically Neural Collaborative Filtering (NeuralCF), LightGCN, and SSL4Rec. We adopted the official implementations (i.e., without any modifications) of Bayesian Personalized Ranking (BPR), Alternating Least Squares (ALS), and ItemKNN models directly from the LensKit recommender package;¹ and, thus, did not include them in this benchmarking exercise.

We conducted the benchmarking and compared our results with the results reported in the respective original papers (using publicly available datasets as well as the metrics mentioned in the original papers). Tables F6, F7, and F8 summarize these comparisons. Our implementations align closely with the architectures and procedures described in the original works. Although our benchmarking results do not match the reported values precisely (due to some inherent random elements in the model training and evaluation processes and minor implementation variations), the performance numbers are highly comparable.

¹ LensKit recommender package: <http://lenskit.org/>

Table F6 Benchmarking Results for NeuralCF

	Hit@10	NDCG@10
Reported	0.701	0.425
Our Implementation	0.687	0.411

Benchmarking is done on the MovieLens dataset. For benchmarking purposes, our implementation of NeuralCF has 32 hidden factors and 3 hidden layers (# of weights: 128, 64, 32) without pre-training. We follow the same finetuning process as described in Section 4.1 of the original NeuralCF paper. Reported results are from Table 2 of the original NeuralCF paper. Results of our implementation represent an average across 5 runs (using different random seeds).

Table F7 Benchmarking Results for LightGCN

	Recall@20	NDCG@20
Reported	0.0411	0.0315
Our Implementation	0.0408	0.0313

Benchmarking is done on the Amazon Book dataset. For benchmarking purposes, our implementation of LightGCN has 2 hidden layers (# of weights: 64) with a learning rate of 0.001 and default mini-batch size of 1024. We follow the same finetuning process as described in Section 4.1.2 of the original LightGCN paper. Reported results are from Table 3 of the original LightGCN paper. Results of our implementation represent an average across 5 runs (using different random seeds).

Table F8 Benchmarking Results for SSL4Rec

	MAP@10	Recall@20
Reported	0.0171	0.0573
Our Implementation	0.0162	0.0568

Benchmarking is done on the Wikipedia dataset. For benchmarking purposes, our implementation of SSL4Rec aligns with the baseline architecture described in Section 4.2 of the original SSL4Rec paper, and we follow the same hyper-parameter search process as described in that same section. Reported results are from Table 2 of the original SSL4Rec paper. Results of our implementation represent an average across 5 runs (using different random seeds).

Appendix G: Hyper-Parameter Tuning for Recommender System Algorithms

The six recommendation algorithms examined in this study cover a wide range of families of recommender system models. ALS and BPR are two canonical representations of the matrix factorization models. ItemKNN is a notable example of the traditional neighborhood-based collaborative filtering methods. NeuralCF, LightGCN, and SSL4Rec represent the family of recommendation algorithms based on modern neural network architectures.

The hyper-parameters of these models are tuned on historical data *ConsData*⁰. For the ALS model, we did a grid search over three features: the number of latent factors $\{8, 16, 32\}$, the number of iterations $\{10, 20, 50\}$, the regularization factor $\{0.1, 0.2, 0.4, 0.8\}$, and the weight on positive instances $\{10, 20, 40\}$.

For the BPR model, we did a grid-search over: the number of latent factors $\{8, 16, 32\}$, the number of iterations $\{10, 20, 50\}$, the regularization factor $\{0.1, 0.2, 0.4, 0.8\}$, and the learning rate $\{0.01, 0.05, 0.1\}$. For the ItemKNN model, we did a grid-search over: the maximum number of neighbors $\{10, 20, 30, 50, 80\}$, the minimum number of neighbors $\{1, 5, 10\}$, the minimum similarity to be considered a neighbor $\{1 \times 10^{-6}, 1 \times 10^{-4}, 1 \times 10^{-2}, 1 \times 10^{-1}\}$, and the aggregation scheme using either the simple sum or the weighted sum. For the LightGCN model, we searched over: the number of latent features $\{8, 16, 32\}$, the learning rate $\{0.001, 0.005, 0.01, 0.05\}$, and the dropout rate $\{0.0, 0.2, 0.4, 0.6\}$. For the NeuralCF model, we tested a wide neural architecture and a deep neural architecture for the Multi-layer Perceptron (MLP) part of the model. Specifically, for a wide neural architecture, we have one layer of MLP with 64 latent features. For a deeper neural architecture, we have 3 layers of MLPs with 32, 16, and 8 latent features respectively. We also tune the dropout for the MLP layers using a set of values $\{0.2, 0.4, 0.6\}$. For the Generalized Matrix Factorization (GMF) layer, we searched over a number of latent features $\{8, 16, 32\}$ and the learning rate $\{0.0001, 0.0005, 0.001, 0.005\}$. We trained the best configuration of LightGCN and NeuralCF to convergence with an Adam optimizer (Zhang 2018). For the SSL4Rec model, we conducted a grid search over four key hyper-parameters. Specifically, we tuned the dimensionality of the latent embeddings over the set $\{64, 128, 256\}$, as this parameter governs the expressiveness of the user and item representations passed to the neural towers. We varied the dropout rate applied to item embeddings over the set $\{0.0, 0.1, 0.2, 0.4\}$ to assess its effect on the contrastive learning regularization. For the learning rate, we evaluated values in the set $\{0.0005, 0.001, 0.005, 0.01\}$, as this controls the step size in the parameter updates via the Adam optimizer. Finally, we explored different weights on the contrastive learning loss $\alpha \in \{0.05, 0.1, 0.2, 0.5\}$, which governs the trade-off between the recommendation loss and the self-supervised auxiliary loss. The temperature parameter τ used in both the recommendation loss and the contrastive loss was tuned over $\{0.05, 0.1, 0.2\}$. We trained the final model configuration to convergence using the Adam optimizer.

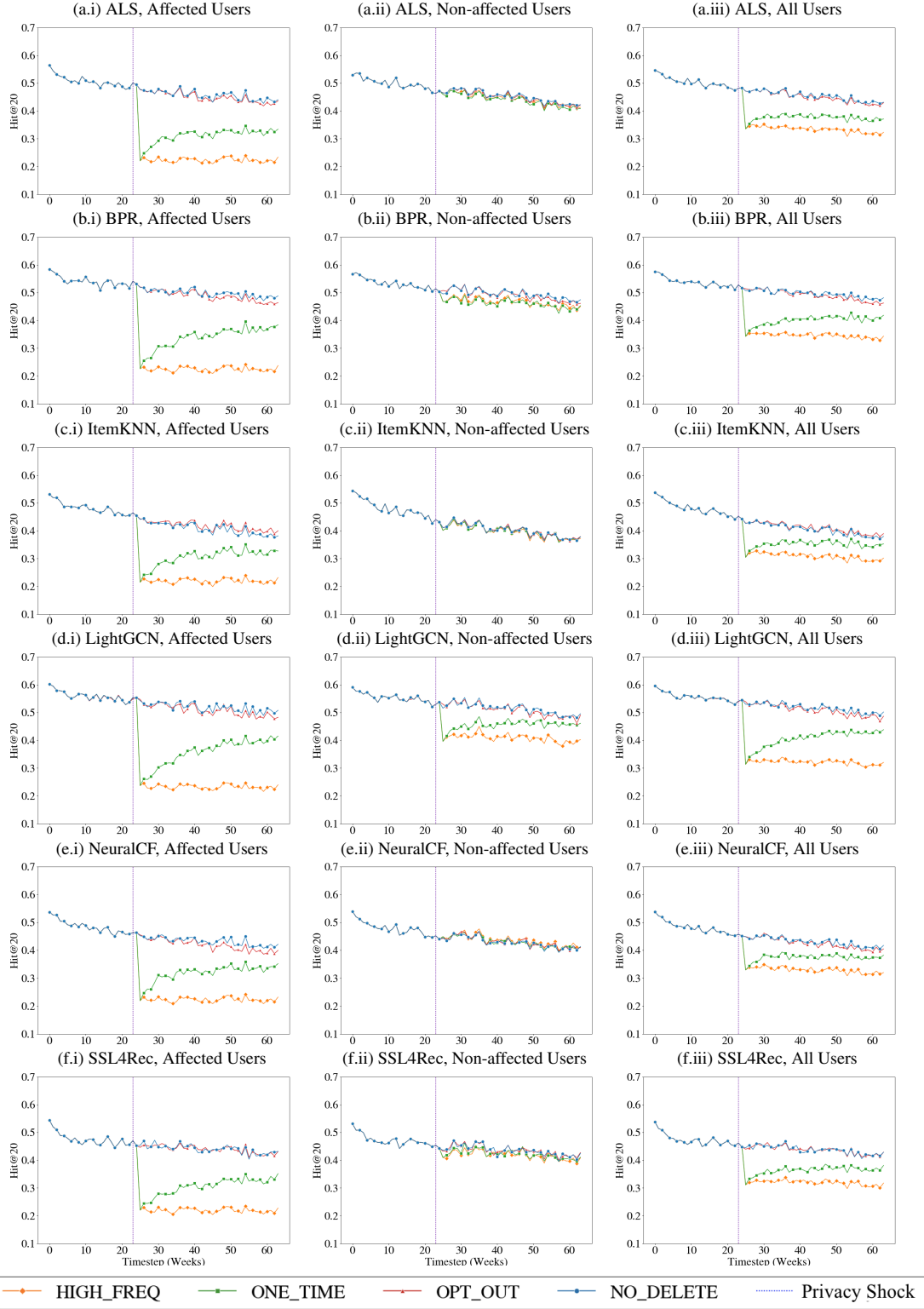
As a result of the fine-tuning process, we select the configuration of the model that yields best *Hit@20*, and fix the configuration throughout the simulation period. For ALS, the best configuration is: the number of latent factors, 16; the number of iterations, 20; the regularization factor, 0.1; the weight on positive instances,

40. For BPR, the best configuration is: the number of latent factors, 32; the number of iterations, 100; the regularization factor, 0.01. For ItemKNN, the best configuration is: the maximum number of neighbors, 50; the minimum number of neighbors, 1; the minimum similarity to be considered a neighbor, 1×10^{-6} ; and the weighted sum as the aggregation scheme. For LightGCN, the best configuration is: the number of latent factors, 16; the number of latent factors of the GMF layer, 0.0; the dropout rate, 0.0. For NeuralCF, the best configuration is: the number of latent factors of the GMF layer, 16; the learning rate, 0.001; the dropout rate of the MLP layers, 0.4; with the deep MLP architecture. For SSL4Rec, the best configuration is: the embedding size, 128; the dropout rate, 0.1; the learning rate, 0.001; $\alpha = 0.1$; and $\tau = 0.1$.

In the incremental learning setup, we fine-tune the incrementally updated BPR with the same set of hyper-parameters specified above except for the learning rate. The set of learning rates for fine-tuning the incremental BPR is $\{0.0025, 0.005, 0.01, 0.05, 0.1\}$. We explore smaller learning rates because the size of the training data for each incremental update is smaller, and a large learning rate with a small batch of data may result in a noisy gradient descent.

Appendix H: Main Results of Experiment 1: Additional RS Algorithms

Figure H31 Impact of Data Privacy Practices with Stable User Preferences



Note that across all figures, 50% of users are affected and all users' reliance on recommendations is 20%.

Appendix I: Computational Complexity Analysis under Different Privacy Practices

We discuss the computational complexity associated with recommender systems (RS) algorithms under different privacy practices. Given that inference typically incurs much smaller computational cost than training, our discussion focuses on the training cost of RS algorithms.

To facilitate a general theoretical discussion, we abstract away the complex model architecture of neural methods and discuss three neural methods (i.e., LightGCN, NeuralCF, SSL4Rec) as a family together. We define N , M , and R as the number of users, items, and user-item interactions, respectively. Let K denote the dimensionality of the user/item latent feature space.

- **BPR:** The training complexity is $O(KR)$ (He et al. 2016), where K arises from the gradient updates on weights in the user/item latent representation and R from iterating over user-item interactions.
- **ALS:** The training complexity is $O((M + N)K^2 + RK + K^3)$ (Takács et al. 2011).
- **ItemKNN:** The training complexity is $O(M^2N)$ (Deshpande and Karypis 2004).
- **Neural Methods:** Neural methods employ the gradient descent paradigm for updating model weights during the training process. It iterates over user-item interactions to calculate the loss, and for each iteration it back-propagates to update the weights defined in the model specification. The computational complexity of this process can be generally modeled as $O(WR)$, where W is the total number of model weights. In practice, the value of W for different neural learning algorithms is a result of specific model architecture / design choices.

The data privacy practices analyzed in this study primarily affect the number of users (N) and user-item interactions (R) in the training data. They may also indirectly affect the number of items (M), if sparsely consumed items disappear from the training data due to all associated users enacting deletion. However, in this discussion, we focus on the first-order effects of privacy practices on N and R , assuming M and K remain constant.

Assume a fraction γ of users enact data deletion via the ONE_TIME or HIGH_FREQ privacy practices. Then, the post-deletion user and interaction counts become $(1 - \gamma)N$ and $(1 - \gamma)R$, respectively. As a result, model training complexity is as follows:

- BPR: $O((1 - \gamma)KR)$
- ALS: $O((M + (1 - \gamma)N)K^2 + (1 - \gamma)KR + K^3)$
- ItemKNN: $O((1 - \gamma)M^2N)$
- Neural Methods: $O((1 - \gamma)WR)$

This shows that, under deletion-based data privacy practices, increasing the fraction of affected users leads to lower model training complexity across all algorithms. The decrease of the computational complexity is essentially linear in γ across most learning algorithms, and slightly sublinear for ALS.

In contrast, under the OPT_OUT privacy practice, affected users do not delete existing data but stop adding new data in the future. In other words, as time goes by and the system accumulates more user data under NO_DELETE (but not under OPT_OUT for affected users), the computational cost of model training under NO_DELETE would be higher than under OPT_OUT. Consider a future time period, where the number of user-item interactions (which was R at the time of the privacy shock) increases by R' under NO_DELETE, i.e., a future time period with $R + R'$ interactions in total. If a fraction γ of users use OPT_OUT, then the number of total interactions that would be available to RS at that future time period would instead be $R + (1 - \gamma)R'$. Comparing the reduction of interaction data from $R + R'$ to $R + (1 - \gamma)R'$, for most learning algorithms, the model training computational complexity under OPT_OUT would reduce sublinearly in γ (as compared to the computational complexity under NO_DELETE). One exception would be ItemKNN, the asymptotic computational complexity of which would remain the same (as it depends on M and N , rather than R).

Notably, the use of data privacy practices decreases the computational cost of training RS models, due to the reduction of training data.

Appendix J: Impact of Item Distinctiveness on RS Performance in the Presence of Data Privacy Practices and User Preference Changes

To further unpack the nuanced performance patterns observed under deletion-based privacy practices, particularly in scenarios where the privacy shock (T_{DataReq}) and preference shift ($T_{\text{PrefChange}}$) are temporally separated, we extend our analysis here. As noted in the main paper, HIGH_FREQ can exhibit a pronounced

performance jump after a preference change—especially when $T_{\text{PrefChange}} > T_{\text{DataReq}}$. This appendix provides a deeper examination of this phenomenon by introducing and empirically validating the notion of *item distinctiveness* in users’ evolving preference structures. We show that early-stage post-change consumption is often dominated by highly distinctive items, which are easier for recommender systems to identify, even with limited recent data. This insight offers a conceptual lens to interpret the transient performance advantage of HIGH_FREQ over ONE_TIME in such settings.

We measure “distinctiveness” of item i_k at position k on user u ’s PrefList_u as:

$$D_{u,k} = \frac{p_u^{k-m} - p_u^{k+m}}{2m}$$

where $p_u^{k-m} = e_{i_{k-m}} \cdot e_u$ is user u ’s preference (i.e., relevance) score for item i_{k-m} at the $(k-m)^{\text{th}}$ position of PrefList_u . Intuitively, we measure the difference between the relevance scores of the items that are “near” the k^{th} item of PrefList_u – more specifically, between the item at m positions ahead of i_k and the item at m positions after i_k – to gauge how different (or distinguishable) the nearby items are from i_k in terms of relevance. A large difference indicates that i_k is in a neighborhood that has highly distinguishable items. Parameter m controls the size of the neighborhood in the calculation; we use $m = 2$ for this example.

Figure J32 shows the average item distinctiveness score at each rank in PrefList_u across all users. On average, items that are most relevant (top 5 or 10) are indeed quite different from each other (i.e., easier for the RS to distinguish and predict), but the rest of the items (i.e., beyond the top-10 most relevant items) are much more homogeneous in terms of their relevance. Then, at each time period, we measure the average distinctiveness score of consumed recommendations² with different algorithms under HIGH_FREQ, and plot how these scores evolve in the presence of a significant user preference change. Figure J33 shows that, across all algorithms, the average distinctiveness score of consumed recommendations increases immediately after $T_{\text{PrefChange}}$, because each user is able to consume highly distinctive items among the top ranks of the user’s new PrefList_u . This score is lower before $T_{\text{PrefChange}}$, because most highly distinctive items among the top ranks of the user’s old PrefList_u have already been consumed by that time.

²Consumed recommendations represent items that were consumed by users specifically due to the RS recommendations. We obtain the average distinctiveness score by averaging the distinctiveness across all consumed recommendations in a given period.

Figure J32 Average Item Distinctiveness (Across All Users) by Relevance Rank in $PrefList_u$

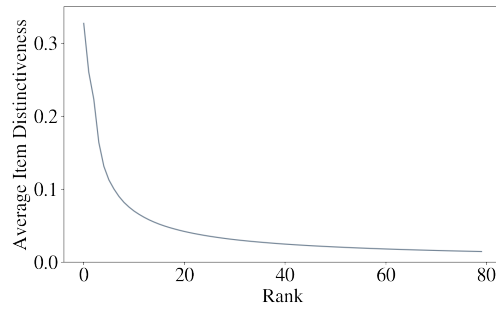
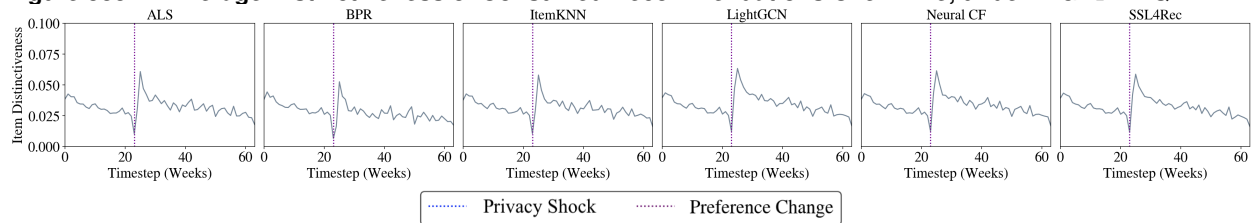


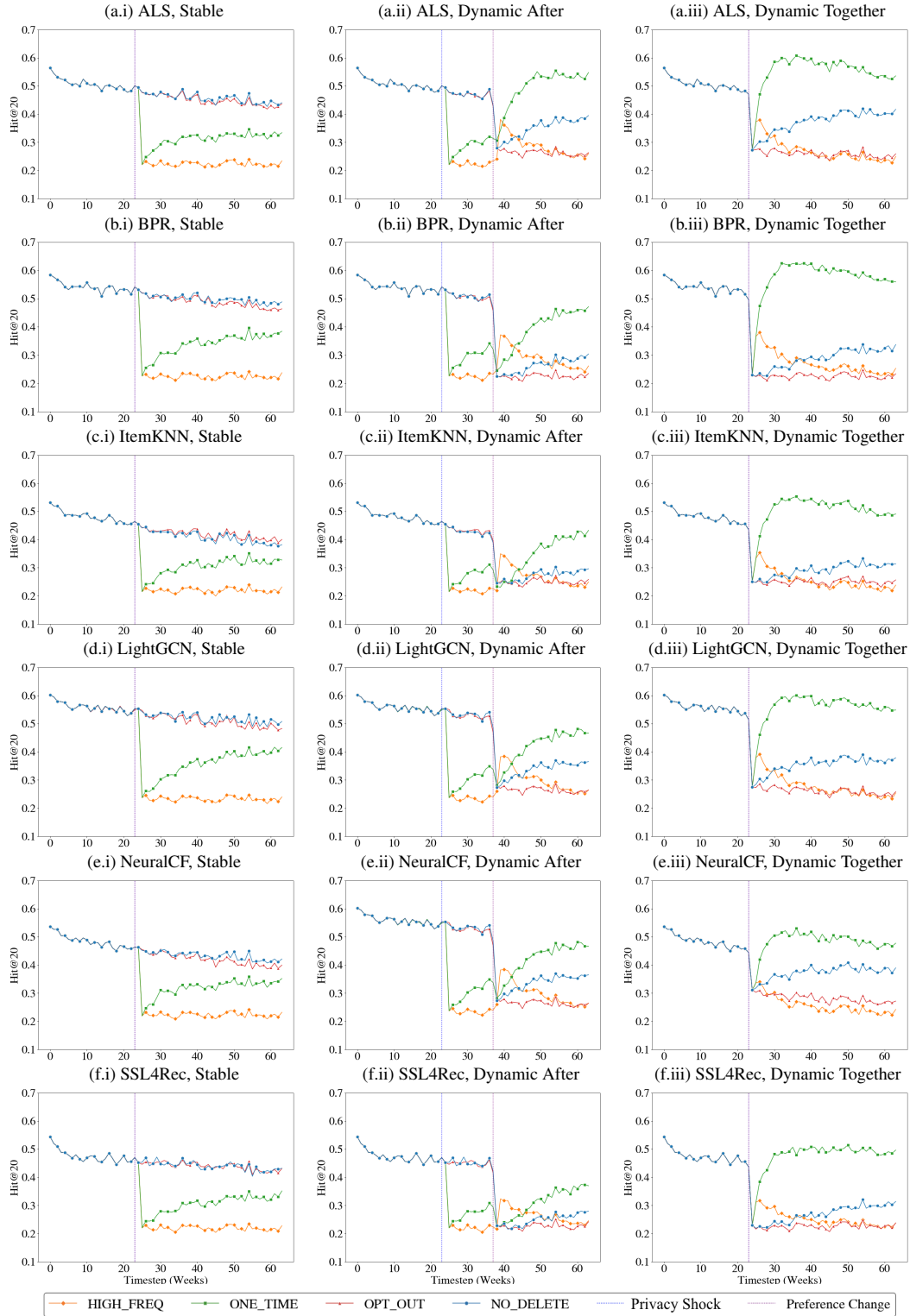
Figure J33 Average Distinctiveness of Consumed Recommendations Over Time, under HIGH_FREQ



For all graphs, the portion of the affected user population is 50%, and the degree of users' reliance on recommendations is 20%. The privacy shock happens together with the preference change. The results are for the affected user population.

Appendix K: Main Results of Experiment 2: Additional RS Algorithms

Figure K34 Impact of Data Privacy Practices with Dynamic User Preferences

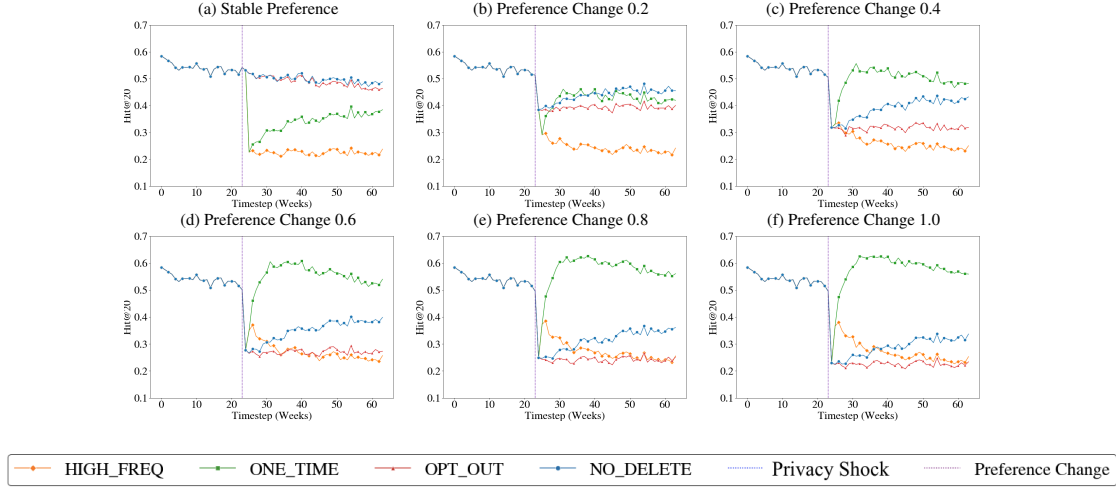


Note that across all figures, 50% of users are affected and all users' reliance on recommendations is 20%.

Appendix L: Varying the Degree of User Preference Change

This figure below shows the impact of varying the degree of user preference change (i.e., 0, 0.2, 0.4, 0.6, 0.8, 1.0) on RS performance under different privacy practices.

Figure L35 Impact of Varying Levels of User Preference Dynamic



The results are on affected users and the simulation is under a setting where 50% of users are affected, algorithm is BPR, and the user preference changes together with $T_{DataReq}$.

Appendix M: Statistical Test Results of Post-Privacy-Shock Performance Differences

Tables M9, M10, M11, and M12 present the results of RS performance after the privacy shock. In particular, they compare the accuracy performance of the default RS setting (i.e., under NO_DELETE) to the RS performance under different data privacy practices. The RS performance is measured at 20 weeks after the privacy shock (at $T = 44$ weeks), i.e., giving some time for each configuration to reach a more stable, steady-state performance level. Each reported value represents an average across 10 simulation runs conducted with different random seeds. Statistical significance, assessed using Student's t-tests, is indicated by asterisks in the tables. These results facilitate the comparison of RS performance under various privacy practices and corroborate the visual analyses provided in the main paper.

Table M9 Performance Differences after Privacy Shock, Stable User Preferences, Hit@20

	BPR	ALS	ItemKNN	NCF	LightGCN	SSL4Rec
NO_DELETE	0.4862	0.4497	0.4015	0.4242	0.5235	0.4306
OPT_OUT	-0.0097	-0.0099	+0.0022	+0.0042	-0.0206**	0.0000
HIGH_FREQ	-0.2723**	-0.2350**	-0.1886**	-0.2092**	-0.2993**	-0.2161**
ONE_TIME	-0.1422**	-0.1349**	-0.0950**	-0.1018**	-0.1499**	-0.1172**

* and ** denotes statistical significance at 5% and 1%, respectively. RS performance is measured at $T = 44$ weeks. The NO_DELETE row shows the actual RS performance; other rows show the difference between RS performance under data privacy practices and that of NO_DELETE.

Table M10 Performance Differences after Privacy Shock, Dynamic User Preferences, Hit@20

	BPR	ALS	ItemKNN	NCF	LightGCN	SSL4Rec
NO_DELETE	0.2829	0.3835	0.2938	0.3749	0.3668	0.2604
OPT_OUT	-0.0681**	-0.1352**	-0.0519**	-0.1050**	-0.1121**	-0.0444**
HIGH_FREQ	-0.0321**	-0.1330**	-0.0550**	-0.1360**	-0.1099**	-0.0182**
ONE_TIME	+0.3194**	+0.1954**	+0.2347**	+0.1160**	+0.2006**	+0.2284**

* and ** denotes statistical significance at 5% and 1%, respectively. RS performance is measured at $T' = 44$ weeks. The NO_DELETE row shows the actual RS performance; other rows show the difference between RS performance under data privacy practices and that of NO_DELETE.

Table M11 Performance Differences after Privacy Shock, IL_STANDARD, Hit@20

	$\alpha=0.0$	$\alpha=0.2$	$\alpha=1.0$
NO_DELETE	0.4862	0.4473	0.2829
HIGH_FREQ	-0.2723**	-0.2183**	-0.0321**
ONE_TIME	-0.1422**	-0.0231**	+0.3194**
IL_STANDARD	-0.0684**	-0.0657**	+0.0347**

* and ** denotes statistical significance at 5% and 1%, respectively. RS performance is measured at $T' = 44$ weeks. The NO_DELETE row shows the actual RS performance; other rows show the difference between RS performance under data privacy practices and that of NO_DELETE.

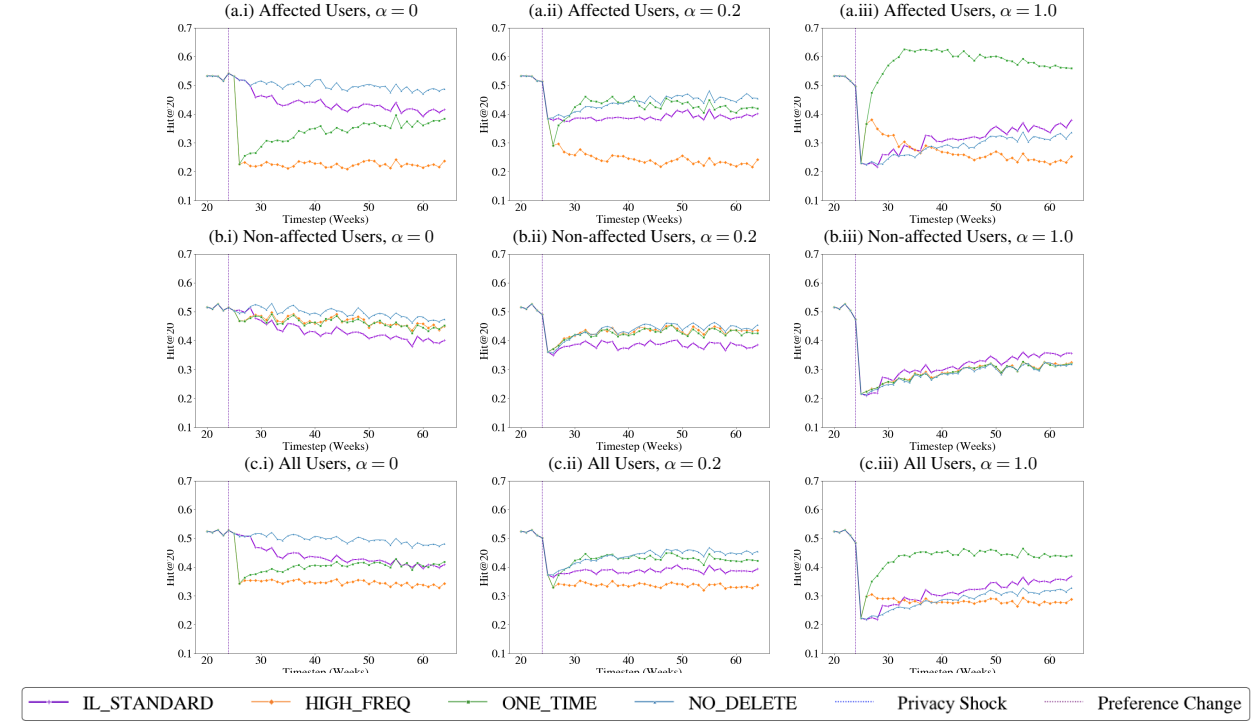
Table M12 Performance Differences after Privacy Shock, IL_OPT_OUT, Hit@20

	$\alpha=0.0$	$\alpha=0.2$	$\alpha=1.0$
NO_DELETE	0.4862	0.4473	0.2829
OPT_OUT	-0.0097	-0.0613**	-0.0681**
IL_OPT_OUT	-0.0594**	-0.0996**	-0.0714**

* and ** denotes statistical significance at 5% and 1%, respectively. RS performance is measured at $T' = 44$ weeks. The NO_DELETE row shows the actual RS performance; other rows show the difference between RS performance under data privacy practices and that of NO_DELETE.

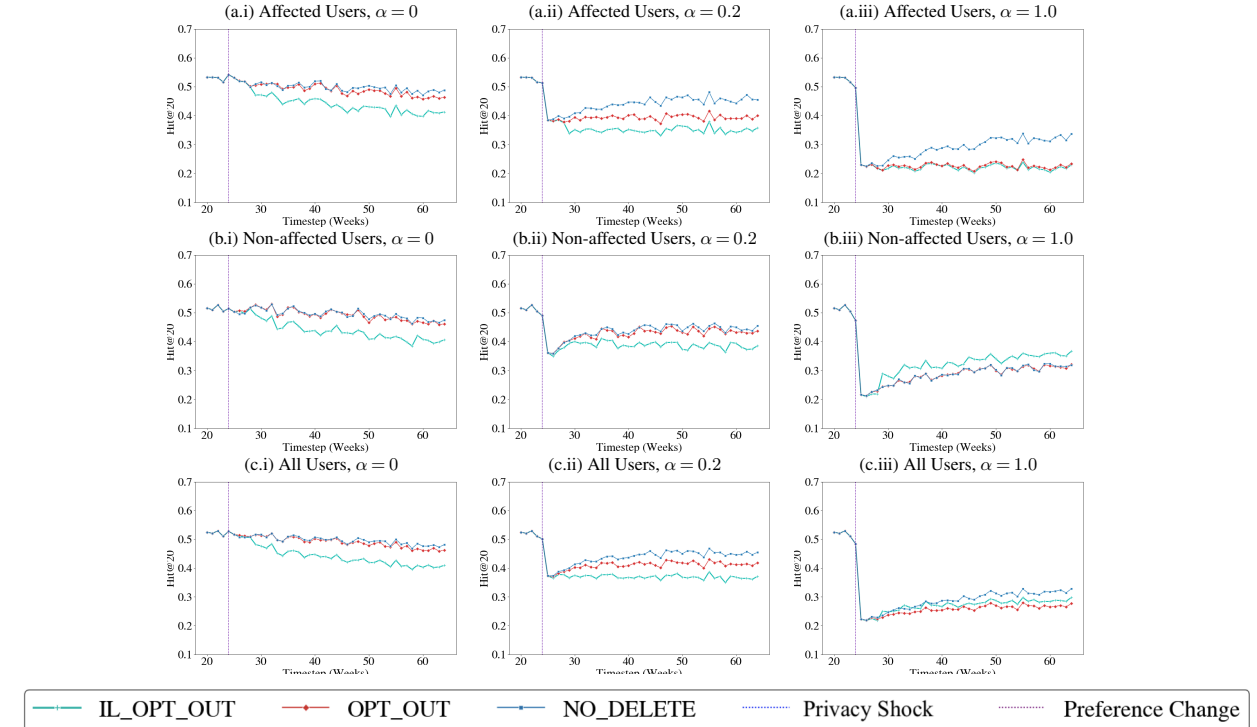
Appendix N: Impact of Data Privacy Practices with Incremental Learning

Figure N36 Incremental and Non-incremental Learning under Deletion-Based Practices



The results are on affected users and the simulation is under a setting where 50% of users are affected, algorithm is BPR, and the user preference changes together with $T_{DataReq}$.

Figure N37 Incremental and Non-incremental Learning under Opt-Out-Based Practices

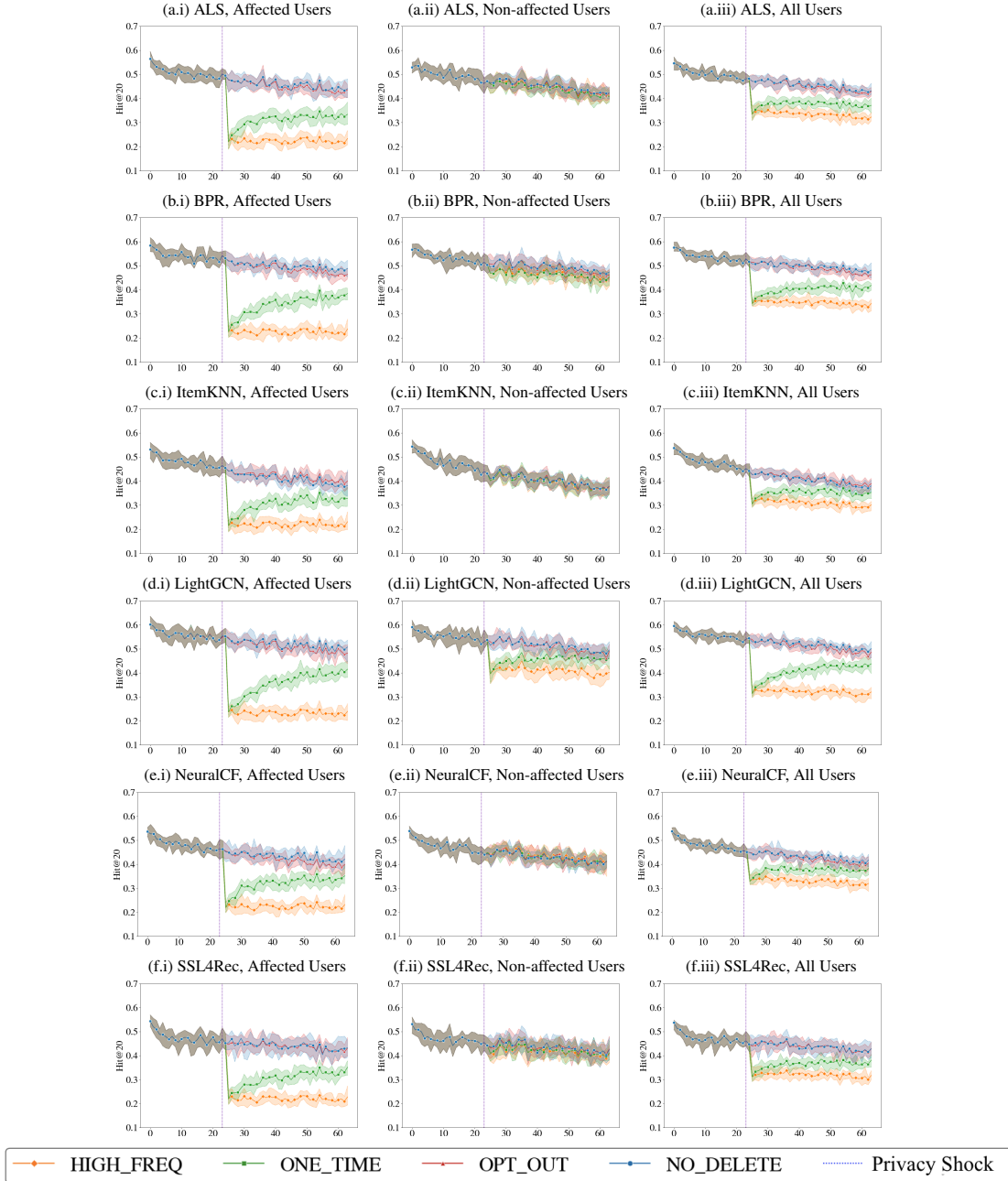


The results are on affected users and the simulation is under a setting where 50% of users are affected, algorithm is BPR, and the user preference changes together with $T_{DataReq}$.

Appendix O: Main Experiment Results with Confidence Bands

Figures O38, O39, O40, and O41 present an alternative visualization of the main results from Experiments 1, 2, and 3, incorporating confidence bands to illustrate the degree of variability across simulation runs.³

Figure O38 Impact of Data Privacy Practices with Stable User Preferences, Hit@20



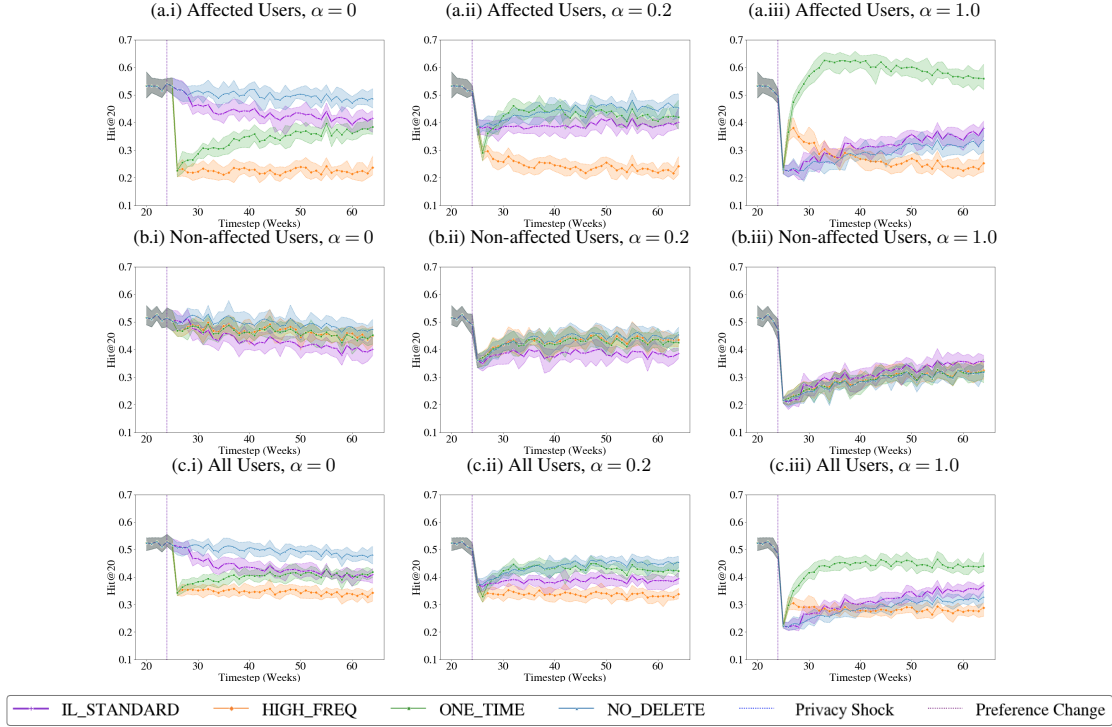
³The confidence bands are constructed from the maximum and minimum values of RS performance at each time step based on 10 independent simulation runs (with different random seeds).

Figure O39 Impact of Data Privacy Practices with Dynamic User Preferences, Hit@20



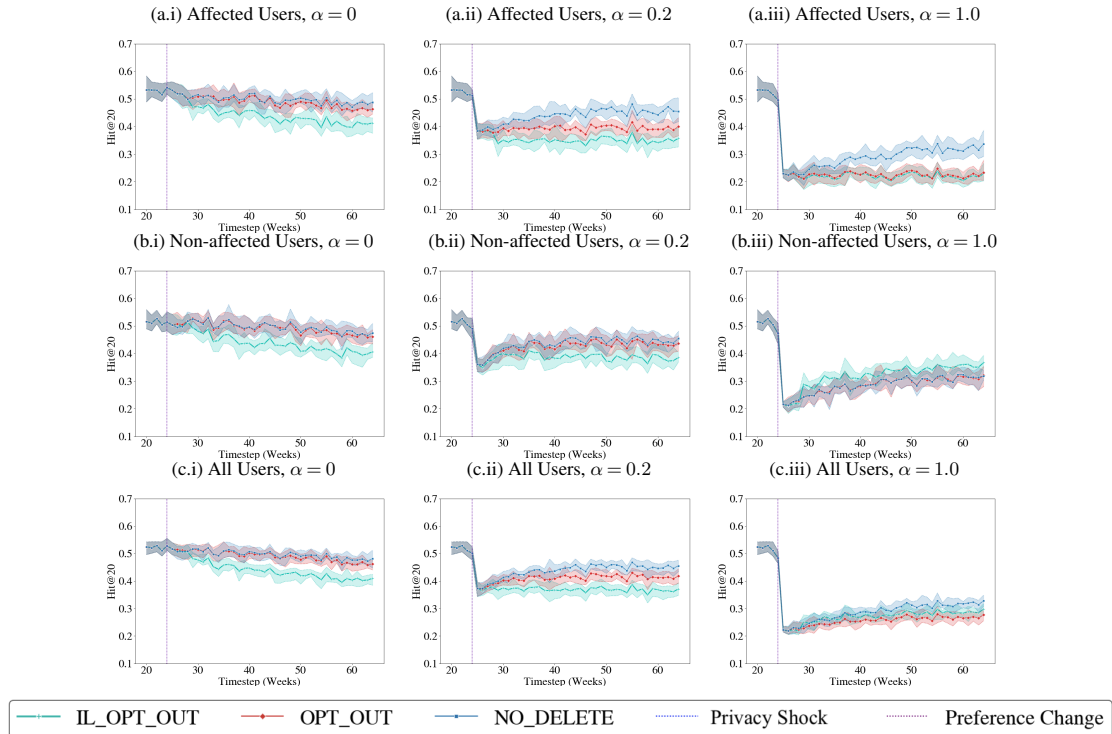
Note that across all figures, 50% of users are affected and all users' reliance on recommendations is 20%.

Figure O40 Incremental and Non-incremental Learning under Deletion-Based Practices, Hit@20



The results are on affected users and the simulation is under a setting where 50% of users are affected, algorithm is BPR, and the user preference changes together with $T_{DataReq}$.

Figure O41 Incremental and Non-incremental Learning under Opting Out Practice, Hit@20



The results are on affected users and the simulation is under a setting where 50% of users are affected, the algorithm is BPR, and the user preference changes together with $T_{DataReq}$.

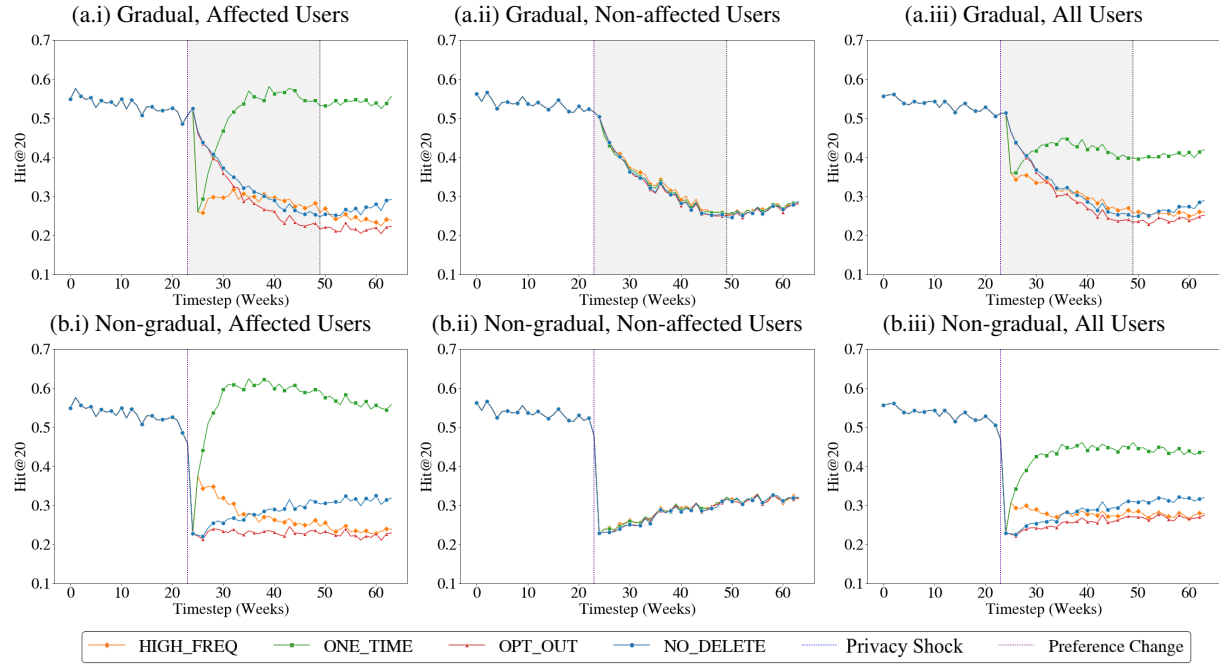
Appendix P: Impact of Privacy Practice on RS Performance with Gradual Evolution of Dynamic User Preference

In Experiment 2 of the main paper, we simulated user preference change by exogenously replacing each affected user's latent preference profile with that of another randomly selected user. While this setup enables a clean and controlled investigation of how data privacy practices interact with shifts in user preferences (e.g., due to changing user's needs), it implicitly assumes that user's preference change occurs instantaneously (i.e., somewhat abruptly). However, user preferences (or needs) in real-world settings may evolve in various ways, including gradually over time (rather than shift instantaneously). To address the robustness of our findings in the settings of gradual evolution of user preferences, we use a refined simulation setup in which user preferences change incrementally following the privacy and preference change shock.

To operationalize the process of gradual preference evolution, we implemented a *progressive* adjustment of the degree of user preference change (α). Starting with $\alpha = 0$, immediately after the preference change shock we incrementally increase α by 0.04 (i.e., by 4 percentage points) each week. This incremental change continues until $\alpha = 1$ by Week 49. Once the preference change is 100% complete, α remains fixed at $\alpha = 1$ for the remainder of the simulation period. As mentioned earlier, this approach allows us to simulate a smooth and continuous shift in user preferences.

In Figure P42, we compare the results under a gradual user preference evolution setup and the original (i.e., non-gradual) user preference setup. Comparing the top and bottom rows of In Figure P42, we observe that, at the end of the simulation period, RS performance under gradual preference change with each data privacy practice converges to a level highly similar as that under non-gradual user preference setup. The results are consistent in terms of the relative rankings of RS performance for affected users under different data privacy practices, at the end of the simulation: ONE_TIME > NO_DELETE > HIGH_FREQ > OPT_OUT. This indicates that the overall, cumulative impact of data privacy practices on RS performance is highly consistent under the gradual vs. non-gradual user preference change in the medium-to-long term horizon (with some expected, logical differences during the short-term period of the gradual preference change, as discussed below).

Figure P42 Impact of Privacy Practices on RS Performance (Hit@20) with Gradual Evolution of Dynamic User Preferences



The results are under a setting where 50% of users are affected, algorithm is BPR, and the user preference changes together with $T_{DataReq}$. In Figures (a-c), the shaded area is the period when the dynamic level gradually increases 4% weekly from 0% to 100%.

Under NO_DELETE and OPT_OUT practices, RS performance experiences a gradual decrease during the period of the gradual user preference change setting (instead of an abrupt decrease observed under the non-gradual preference change setting). This is expected, because the system retains the personal data collected before the preference change under these two privacy practices; the predictive value of this personal data gradually decreases as the user gradually shifts away from his/her old preferences. In terms of the HIGH_FREQ privacy practice, RS performance under the gradual user preference evolution does not surge after the preference change shock, although it does under the non-gradual preference change setting. As mentioned in the main paper (and discussed in Appendix J), the system performance surges because the top-ranked items in the new user preference list are more distinctive (and, hence, more accurately predictable). Now that the user preferences under the gradual evolution setting do not *immediately* switch fully to the new preferences, this distinctiveness (and, hence, the performance surge) is no longer exhibited. Finally, for ONE_TIME, we see fairly consistent patterns across the gradual vs. non-gradual preference change settings; RS performance first decreases due to the data deletion and then gradually increases as it keeps collecting more relevant (i.e., reflecting new user preferences) personal data.

Overall, as discussed above, the impact of data privacy practices on RS performance are consistent across the gradual vs. non-gradual preference change setups, which further highlights the generalizability of the key findings of the study.

Appendix Q: Impact of Data Privacy Practices on RS Performance: User Population with Heterogeneous Preference Changes

In this section, we perform an additional robustness check for our simulation setup. Specifically, our experiments in the main paper assume a *homogeneous* level of preference change in a population, i.e., all users exhibited the same degree of preference change. Here we implement a data-driven approach to derive *heterogeneous*, individual, user-specific levels of preference change that are based on the items that each user has consumed. In other words, this allows us to move from a more stylized setup, where the entire population has the same level of preference change, to a more nuanced setup, where each user exhibits their own specific level of preference change that is based on their consumption history.

To operationalize this approach, we propose a method to empirically estimate the “dynamism level” of each item based on whether (and how much) its latent representation evolves over time. To assess the item’s dynamism level, we first initialize the latent user preference profile e'_u and the latent item content profile e'_i using the first six months of our seeding dataset.⁴ Then, we initialize user preference profile e_u and item content profile e_i with the entire seeding dataset. We then define the dynamism level of item i as

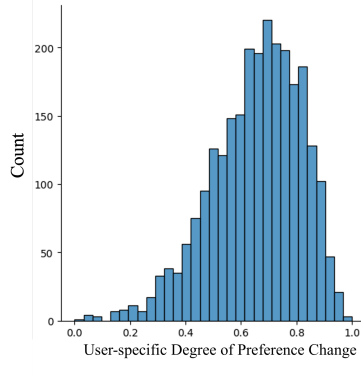
$$ItemDynamismLevel_i = 1 - CosSim(e'_i, e_i) \quad (9)$$

where $CosSim$ is a cosine similarity measure that is normalized using the Min-Max normalization approach. If the item’s latent features change significantly over time, then the similarity between e'_i and e_i is low and, hence, the dynamism level of item i is high.

We further define the consumption-driven, user-specific degree of preference change $\hat{\alpha}_u$ of user u as the average of $ItemDynamismLevel_i$ over all items i consumed by u before the preference change. Simply put, if user u consumed more “dynamic” items, then we model the user to have a higher degree of preference

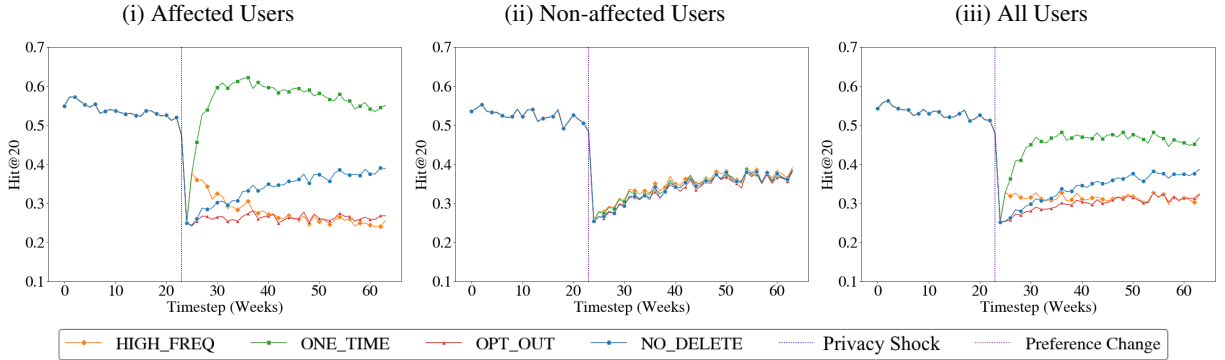
⁴We use Multi-Channel Retailer seeding dataset in this exploration. This entire dataset spans the period from 2018-06-01 to 2019-07-01. We use the data in the period between 2018-06-01 to 2018-12-01.

Figure Q43 Distribution of the consumption-driven, user-specific degree of preference change $\hat{\alpha}_u$



change. Figure Q43 shows the distribution of the consumption-driven, user-specific degree of preference change $\hat{\alpha}_u$ across the entire population. The mean of this distribution is 0.654 with a standard deviation of 0.161. This indicates that the user-specific degree of preference change $\hat{\alpha}_u$ is both quite high (on average) and quite diverse in the multi-channel retailer seeding dataset. This is consistent with the findings from the previous literature that user preferences can be strongly dynamic in the e-commerce context, e.g., can change significantly from visit to visit (Ludewig and Jannach 2018).

Figure Q44 RS Performance with Heterogeneous Degrees of User Preference Change

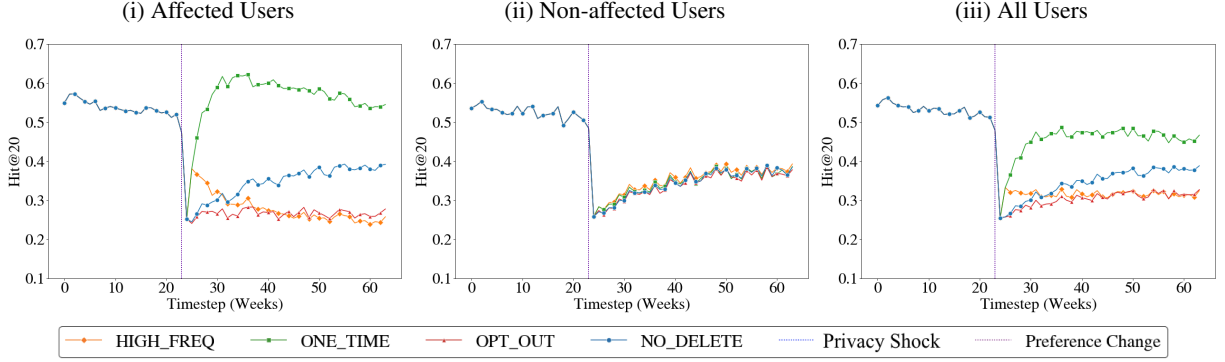


Note that across all figures, 50% of users are affected, algorithm is BPR, and the user preference changes together with $T_{DataReq}$.

In Figure Q44, we show the simulation results with the user population where each user u has an individual, consumption-driven degree of preference change $\hat{\alpha}_u$. The key RS performance patterns from Experiment 2 in the main paper, where each user had the same degree α of exogenous preference change, still hold. For visual comparison, in Figure Q45 we provide the results using the original dynamic preference setting with the homogeneous preference change degree $\alpha = 0.65$ (since the average value of $\hat{\alpha}_u = 0.654$), which shows nearly identical RS performance patterns to those in Figure Q44.

Importantly, the main findings regarding the impact of data privacy practices on RS performance remain consistent in this new setup of having heterogeneous, user-specific, consumption-driven levels of preference change, which further highlights the generalizability of the key findings of the study.

Figure Q45 RS Performance with Homogeneous Degree of Preference Change ($\alpha = 0.65$)



Note that across all figures, 50% of users are affected, algorithm is BPR, and user preferences change together with $T_{DataReq}$.

Appendix R: Category-Specific Dynamic User Preferences

In Appendix Q, we consider heterogeneous, user-specific dynamic preferences where the degree of preference change depends on the item that the user consumes. In this appendix, we specifically consider one important item characteristic, the item category, which can have an impact on the degree of preference change. We extend our simulation framework by considering category-specific dynamic user preferences.

We derive the item-category-specific degree of preference change by averaging the item dynamism levels (as defined in Equation 9 of Appendix Q) across all items within the same category. Since the item dynamism level reflects how strongly user preferences on that individual item change over time, the item-category-specific degree of preference change reflects the aggregate change on items from that category. We denote the category of the item j as c_j and denote the item-category-specific degree of preference change for category c as α_c .

To model the dynamic user preference change, we first assign user u with a random new user preference profile $PrefList_u$. We then obtain ground-truth rating (i.e., preference score) in the preference profile of the user u after the preference change ($\widehat{PrefList}_u$) as

$$\hat{p}_{uj} = (1 - \alpha_{c_j}) \times p_{uj} + \alpha_{c_j} \times p_{vj} \quad (10)$$

where p_{uj} is the preference score for item j in user u 's original preference profile ($PrefList_u$), and p_{vj} is the preference score for item j in user v 's preference profile ($PrefList_v$). This equation characterizes the intuition behind the item-category-dependent preference change: when an item belongs to a category with a stronger dynamism level (α_{c_j} is higher), the new preference score ($\hat{p}_{uj}$) deviates more strongly from the original preference score (p_{uj}). In essence, this model captures the idea that user preferences can be more dynamic for certain item categories than others.

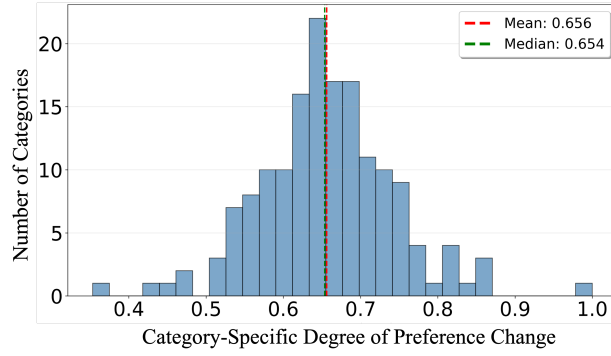
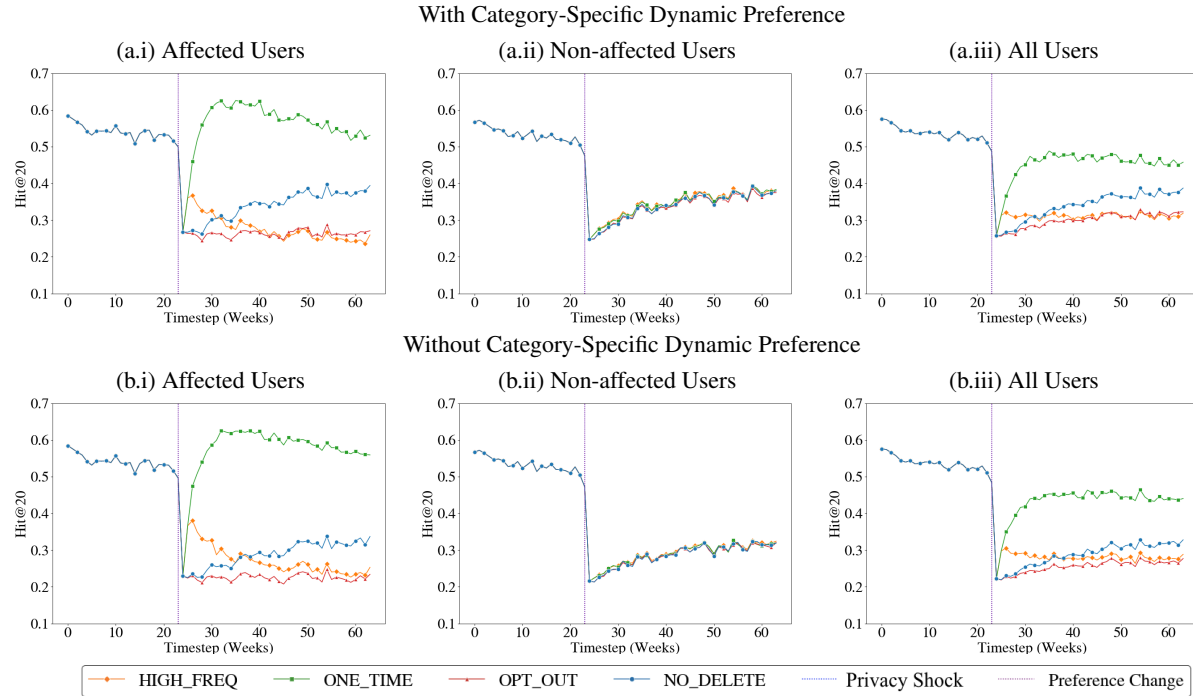


Figure R46 Histogram of Category-Specific Degree of Preference Change

We obtain the item category information for the Multi-Channel Retailer dataset. There are 159 categories in total across all items in the simulation. We visualize the distribution of the category-specific degree of preference change in Figure R46. The category-specific degree of preference change is 0.65 on average. We then simulate the main experiment setup for dynamic user preferences (i.e., in Experiment 2 of the main paper) with this category-specific dynamic preference modeling. Figure R47 presents a comparison on simulation results between the setup with category-specific dynamic preference modeling and the standard dynamic preference setup (as in Experiment 2 of the main paper). We observe that the general performance patterns are highly consistent across two setups, demonstrating strong robustness and generalizability of main findings in the paper.

Figure R47 Impact of Privacy Practices on RS Performance (Hit@20) with vs. without Category-Specific Dynamic Preference (Multi-channel Retailer Dataset)

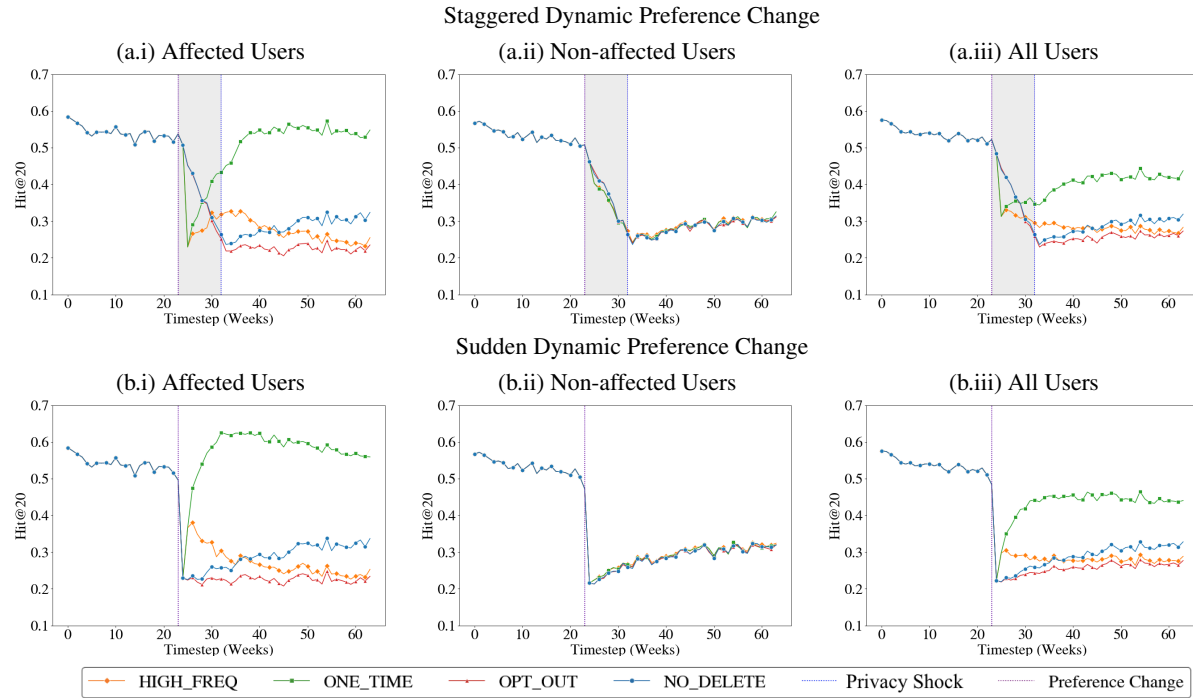


The results are under a setting where 50% of users are affected, algorithm is BPR, and the user preference changes together with $T_{DataReq}$. The simulation without category-specific dynamic preference sets a homogeneous degree of preference change at 100% for all users.

Appendix S: Staggered User Preference Change

We relax the assumption that the user preference change is a one-time, sudden shock by simulating a staggered user preference change. We randomize the users into ten groups, corresponding to the first ten weeks after the deletion shock. The user preferences for each group change during the assigned week. This setup maintains an experimentally controlled preference change while mimicking the non-synchronicity of the user preference change that can be more representative of the real-world settings. Figure S48 compares the simulation results for the staggered vs. sudden preference change settings. As would be expected, the performance degradation under NO_DELETE and OPT_OUT and the performance recovery under ONE_TIME become comparatively more gradual during the period of the staggered preference change. In terms of the system performance under HIGH_FREQ, the sudden surge after the privacy shock also becomes more gradual and smooth. After the staggered change period, the main patterns are highly consistent with those in the sudden preference change setup (used in the main paper).

Figure S48 Impact of Privacy Practices on RS Performance (Hit@20): Staggered vs. Sudden Dynamic Preference Change (Multi-channel Retailer Dataset)



The results are under a setting where 50% of users are affected, algorithm is BPR, and the user preference changes together with $T_{DataReq}$. The simulation sets a homogeneous degree of preference change at 100% for all users. The gray area highlights the period of the staggered dynamic preference change.

Appendix T: Calibrating User Reliance on Recommendations

We calibrate user reliance on recommendations based on real-world data. Specifically, we use the click-stream data of users in the Multi-Channel Retailer seeding dataset. The click-stream data records different types of user actions, such as clicks on the recommendation list and clicks on product pages. We estimate the user reliance on RS as the percentage of clicks on the recommendation list for each user. Figure T49 shows the distribution of the empirical user reliance on RS. There were 851 users who never clicked on the recommendation list. The average percentage of clicks on the recommendation list (across all users) is 0.145 with a standard deviation of 0.19.

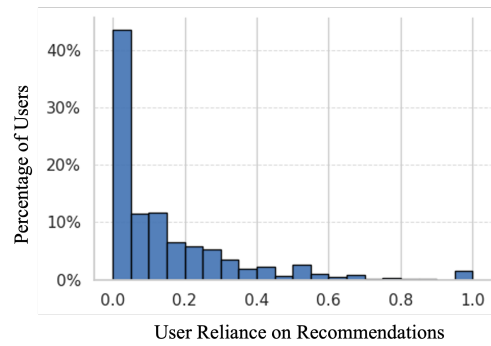
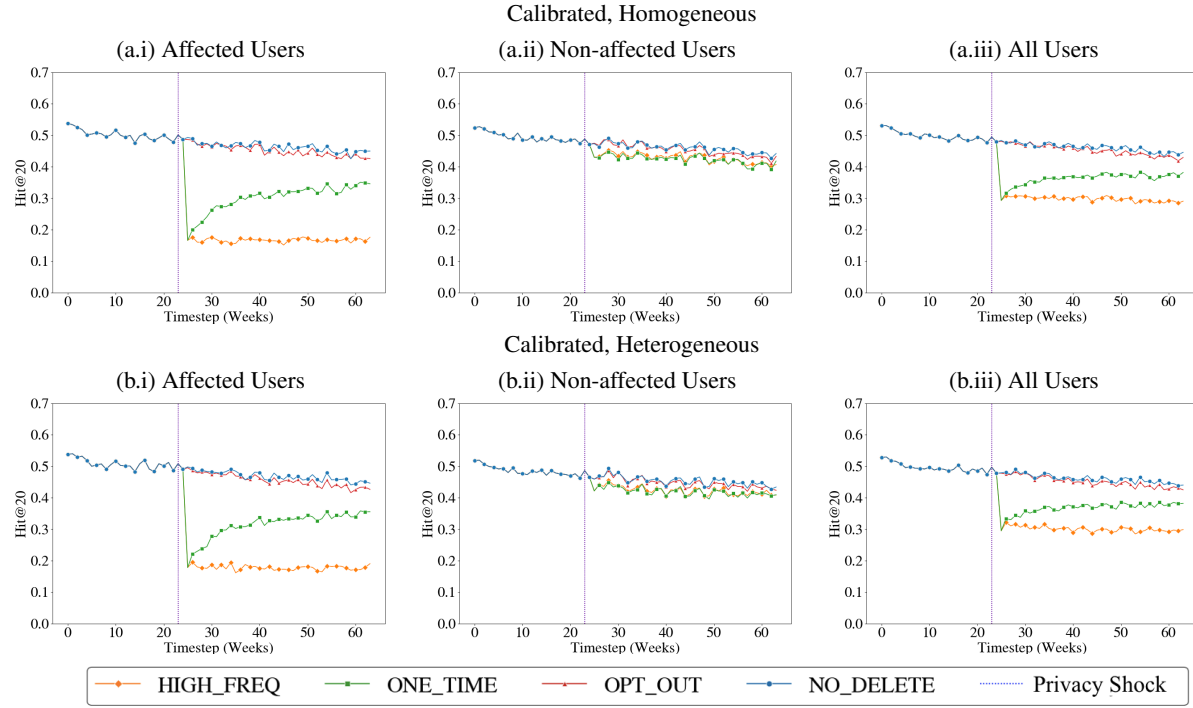


Figure T49 Empirical User Reliance on RS

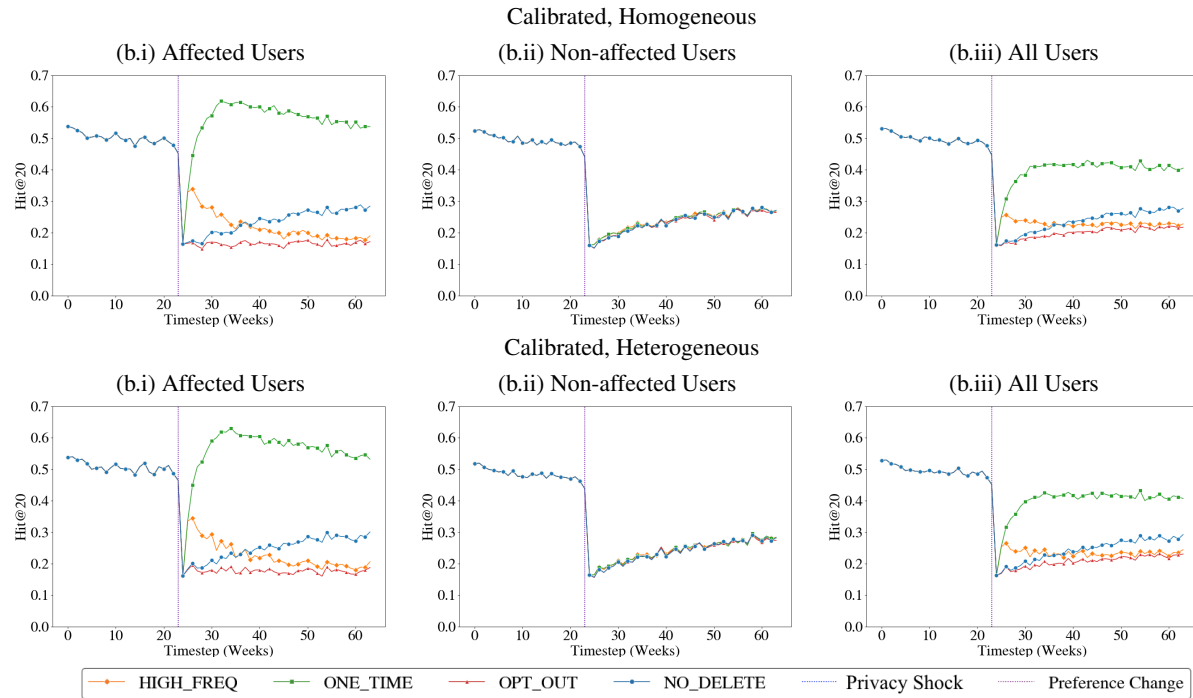
We run two versions of simulations calibrated to the empirical user reliance on recommendations. Specifically, in the first “Calibrated, Homogeneous” version, we calibrate the global, fixed user reliance to the average of the empirical user reliance (i.e., 14.5%); in the second “Calibrated, Heterogeneous” version, we calibrate each user’s reliance to its corresponding empirically estimated value (in a heterogeneous manner). Figures T50 and T51 show comparisons between simulation results of Calibrated Homogeneous vs. Heterogeneous reliance settings under stable and dynamic user preferences, respectively. The general RS performance patterns in both versions are highly consistent to those observed in the main paper, demonstrating strong robustness and generalizability of the main findings with realistic considerations.

Figure T50 Impact of Privacy Practices on RS Performance (Hit@20) with Calibrated Homogeneous vs. Heterogeneous User Reliance on Recommendations (Stable User Preferences, Multi-Channel Retailer Dataset)



The results are under a setting where 50% of users are affected, algorithm is BPR, and the user preference changes together with $T_{DataReq}$.

Figure T51 Impact of Privacy Practices on RS Performance (Hit@20) with Calibrated, Homogeneous vs. Heterogeneous User Reliance on Recommendations (Dynamic User Preferences, Multi-Channel Retailer Dataset)



The results are under a setting where 50% of users are affected, algorithm is BPR, and the user preference changes together with $T_{DataReq}$.

Appendix U: Dynamic User Reliance on Recommendations

We relax the assumption that user reliance on recommendations is static and homogeneous across users by introducing a dynamic and user-specific reliance model. We model the probability of a user relying on RS, r_t , using a time-dependent logistic function. Let $\tau^u(t)$ denote the most recent timestep before t at which user u consumed a recommended item, and let $p_{u,\tau^u(t)}^{rec}$ be the true preference score of that item. The user-specific dynamic reliance is defined as:

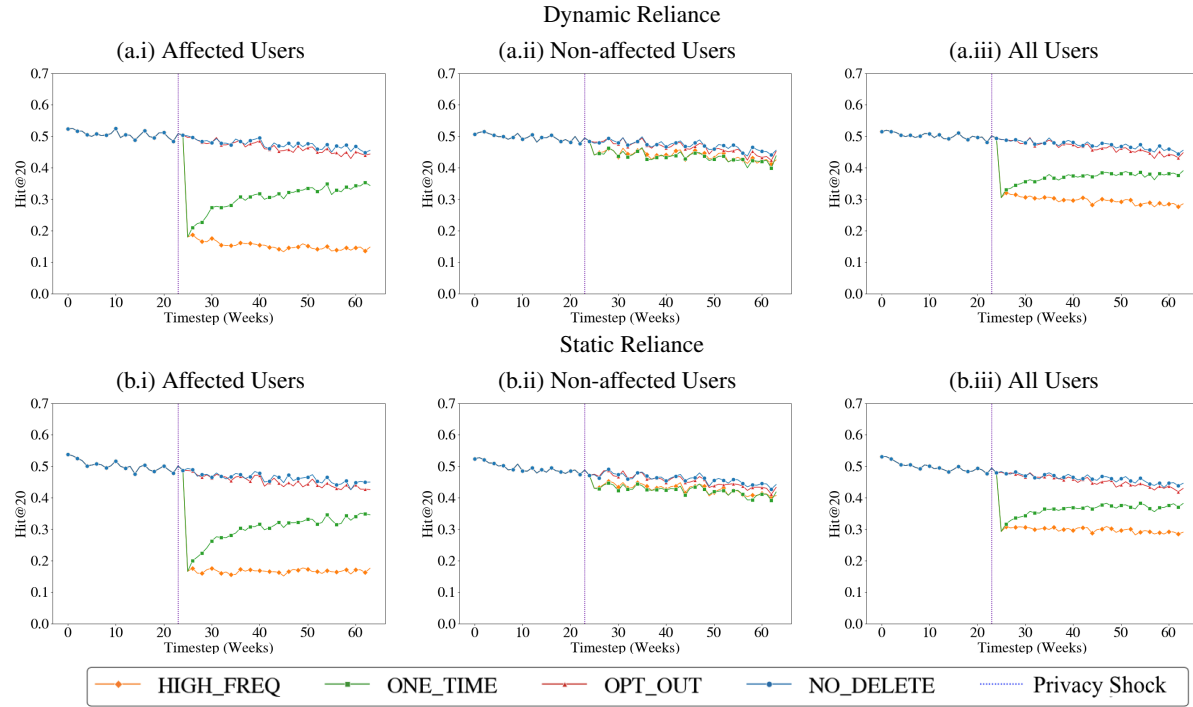
$$r_t^u = \sigma\left(\theta_0 + \theta_1 p_{u,\tau^u(t)}^{rec}\right) \quad (11)$$

where intercept θ_0 governs the base reliance probability. Reliance r_t^u is piecewise constant between consecutive recommendation-consumption events. In other words, the degree of user reliance on recommendations can only change when the user consumes from the recommended items list; this reliance remains unchanged if the user does not consume or consumes items not based on recommendations. In the case where the user consumes one of the recommended items, if the consumed item has low relevance for the user (i.e., the user’s ground-truth rating is low), the user reduces their reliance on RS. Thus, coefficient θ_1 can be interpreted as the user’s sensitivity to the system performance change, and coefficient θ_0 is the baseline reliance due to the user’s algorithmic appreciation or aversion (Logg et al. 2019, Dietvorst et al. 2015). This perspective that user reliance can be dynamically adjustable based on user’s experience of system performance is supported in previous studies on user trust in machine learning systems (Lohrenz et al. 2007, Yu et al. 2017, 2018).

We simulate the main experiment setups (i.e., the stable vs. dynamic user preference setups of Experiments 1 and 2 in the main paper) by replacing the static user reliance with dynamic user reliance model for the Multi-Channel Retailer seeding dataset. We set $\theta_0 = -2$ and $\theta_1 = 0.9$ to calibrate the user reliance (before the privacy shock) based on the one observable in the real world (14.5% based on our estimation for Multi-channel Retailer Dataset; please see Appendix T for details). We show simulation results under this setup of dynamic user reliance in Figures U52 and U53.

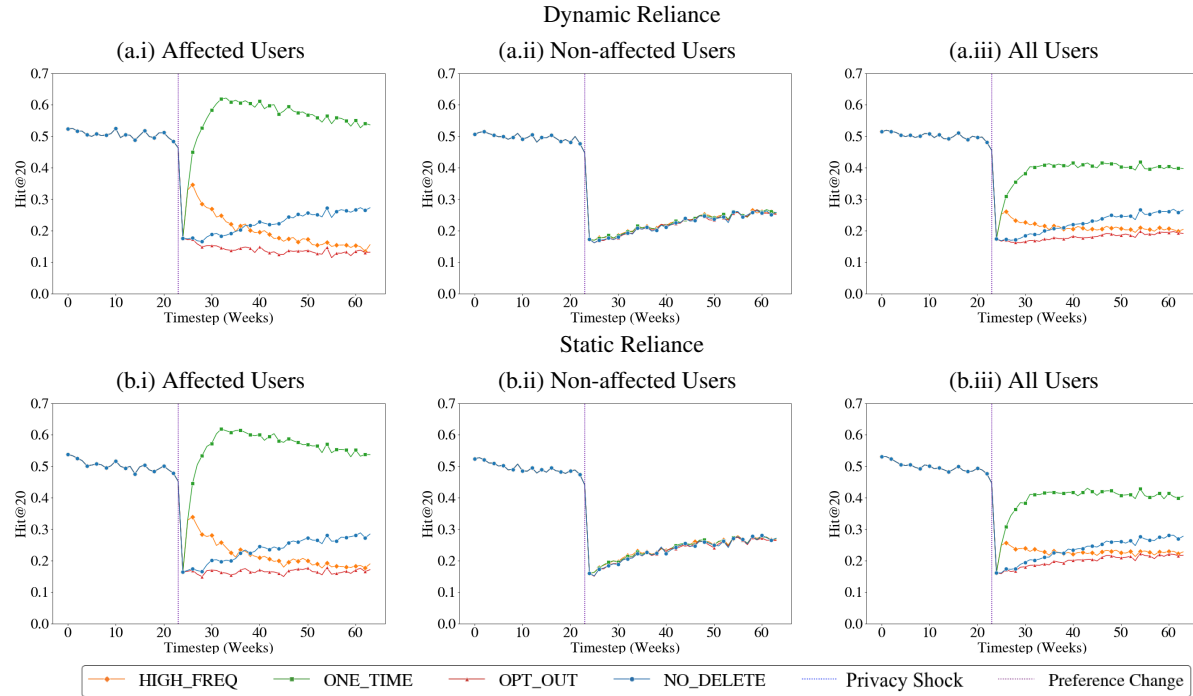
Figures U52 and U53 compare simulation results for dynamic vs. static RS reliance settings under stable and dynamic user preferences, respectively. The general performance patterns remain highly consistent

Figure U52 Impact of Privacy Practices on RS Performance (Hit@20) with Dynamic vs. Static User Reliance (Static User Preferences, Multi-Channel Retailer Dataset)



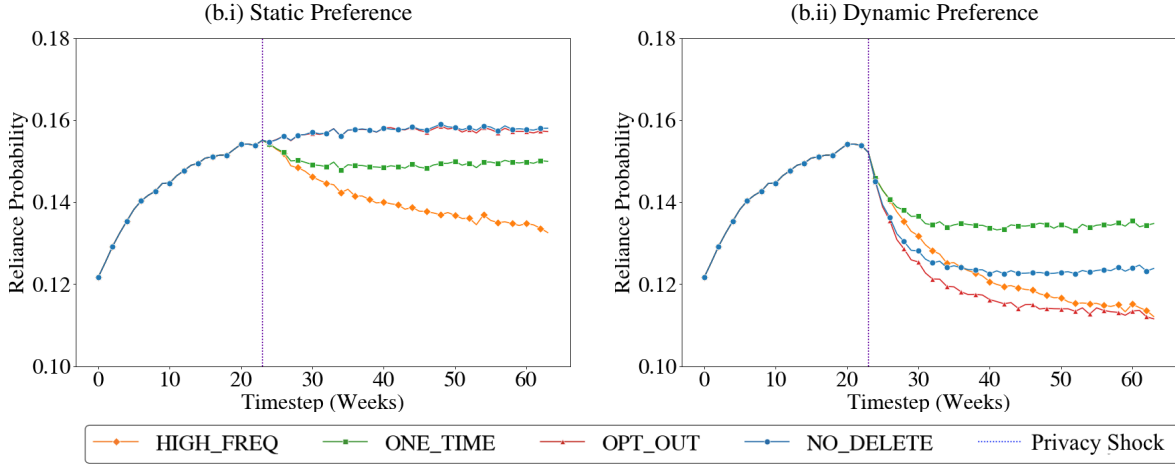
The results are under a setting where 50% of users are affected, algorithm is BPR, and the user preference changes together with $T_{DataReq}$.

Figure U53 Impact of Privacy Practices on RS Performance (Hit@20) with Dynamic vs. Static User Reliance (Dynamic User Preferences, Multi-Channel Retailer Dataset)



The results are under a setting where 50% of users are affected, algorithm is BPR, and the user preference changes together with $T_{DataReq}$.

Figure U54 Dynamic User Reliance by Timesteps (Static vs. Dynamic User Preferences, Multi-Channel Retailer Dataset)



The results are under a setting where 50% of users are affected, algorithm is BPR, and the user preference changes together with $T_{DataReq}$.

across the the dynamic vs. static RS reliance settings under both stable and dynamic user preferences. We also plot the average dynamic user reliance over time in Figure U54. We observe that the user reliance increases gradually before the privacy shock because the system is able to recommend more relevant items with the accumulation of more interaction (training) data. Then, the user reliance drops (by different degrees for different privacy practices) after the privacy shock because the system performance degrades. This observation provides further support that the dynamic user reliance modeling properly reflects the changes in performance-based user trust in the system.

Appendix V: Modeling New Item Arrival

We extend our simulation framework to incorporate the arrival of new items. To accurately reflect the real-world item dynamics, we seeded the simulated item arrival schedule based on the arrival patterns drawn from the original dataset. Since the dataset does not directly record item creation dates, we used the item's *first-seen* date (the date it was first consumed by any user) as a practical proxy for its arrival on the platform.

Figure V55 shows the number of newly arrived items in each week for the Multi-Channel Retailer seeding dataset. On average, around 416 items arrive each week.⁵ As mentioned earlier, we consider the first-seen date of an item as its arrival date. However, some items may have actually arrived before the period covered by our dataset; therefore, the number of arrivals during the early portions of our data period could be

⁵The dataset reflects the period from June 1, 2018 to July 1, 2019. The 25th week in this period was the week of Black Friday, when the sales volume increased dramatically and the number of first-seen items also spiked strongly.

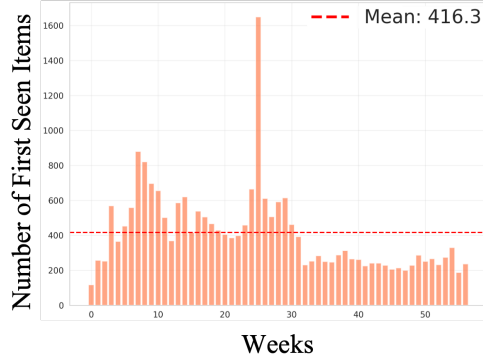


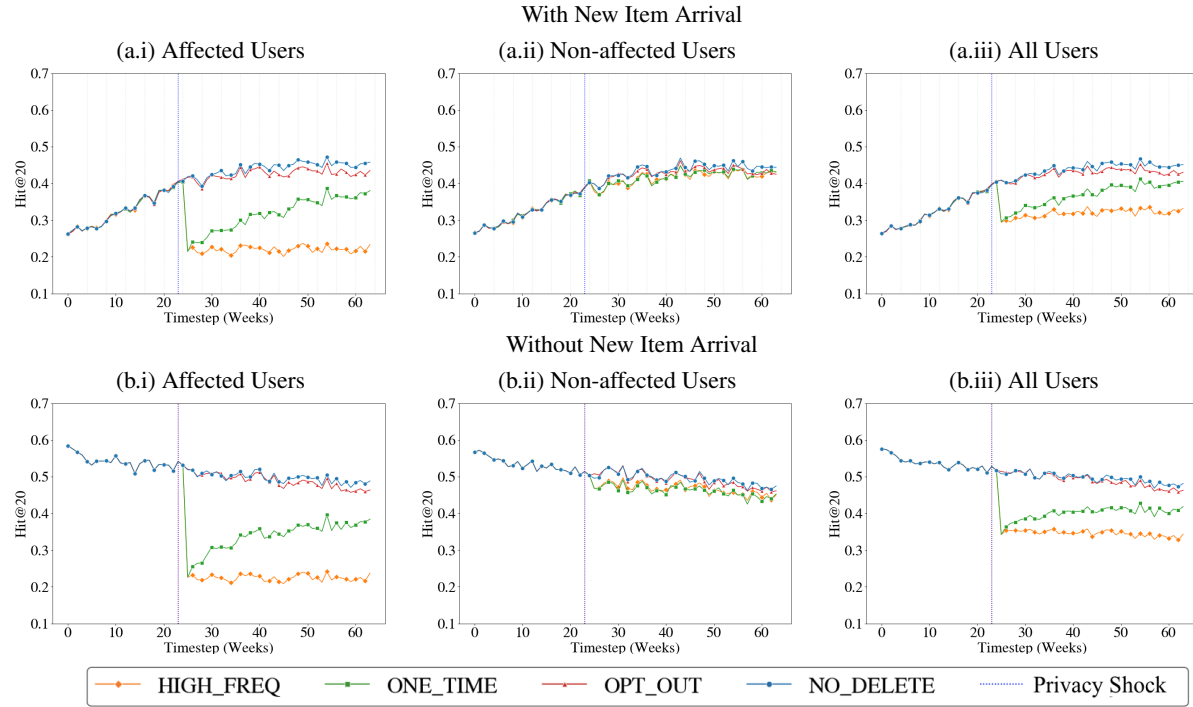
Figure V55 Number of First-Seen Items by Week

overestimated because of that. For this reason, we perform the arrival rate estimation only based on the later portion of our data period (i.e., after 32nd week). Figure V55 shows that the number of first-seen items in each week is almost uniformly distributed around the mean of 205, which is roughly 0.8% of items in the entire item population of the seeding dataset. Thus, in our simulation, we also adopt a uniform item arrival schedule with 0.8% of the entire item population arriving each week (i.e., 20 new items arrive uniformly each week for 65 weeks). To avoid the strict cold-start problem (i.e., where there is zero interaction information about the item), these new items arrive in the system with 5 consumption records.⁶

In Figures V56 and V57, we compare simulation results for the settings with vs. without new item arrival under stable and dynamic user preferences, respectively. The general performance patterns are highly consistent with one nuance. We observe that the baseline system performance with new item arrival has a general tendency to increase over time, because new item arrival continuously reveals richer user preference information, which the system learns to improve its performance gradually. In contrast, the baseline system performance without new item arrival has a general tendency to slightly decrease over time. This reflects the fact that, over time, the most-preferred items get consumed by users and, without new item arrival, the quality of the remaining items will inevitably slowly decrease.

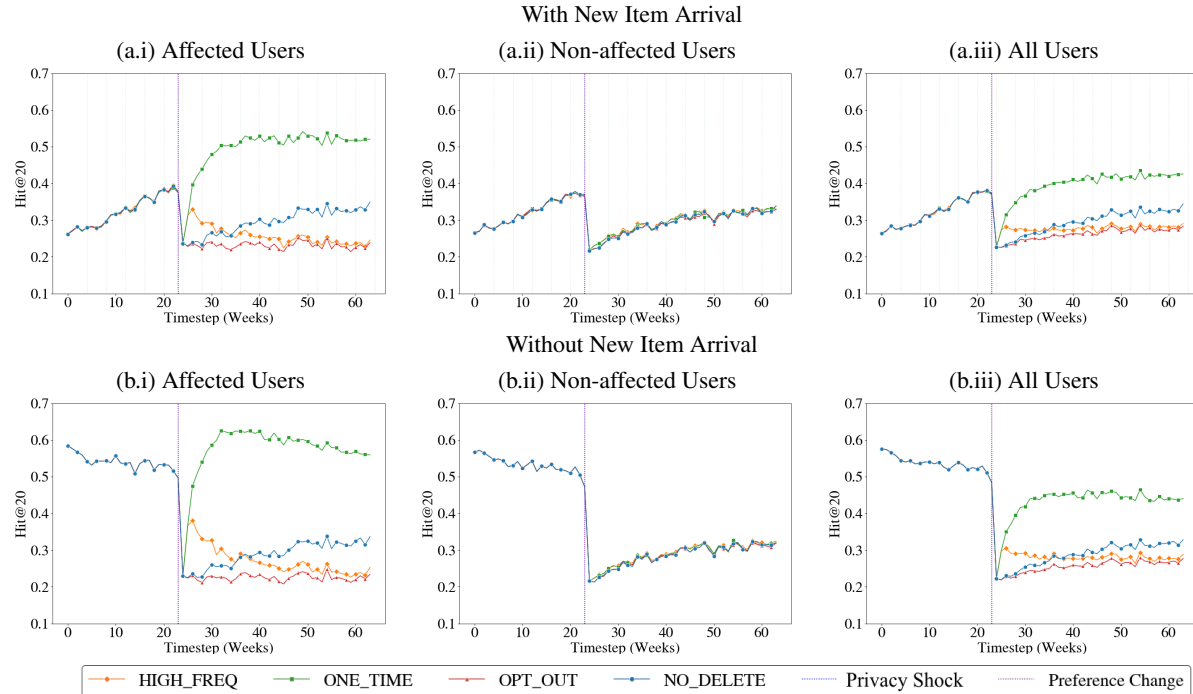
⁶This is consistent with the real-world practice where strict cold-start items are excluded from RS until they have accumulated some minimal number of interactions that can then be used for personalization purposes.

Figure V56 Impact of Privacy Practices on RS Performance (Hit@20) with vs. without New Item Arrival Modeling (Stable User Preferences, Multi-Channel Retailer Dataset)



The results are under a setting where 50% of users are affected, algorithm is BPR, and the user preference changes together with $T_{DataReq}$.

Figure V57 Impact of Privacy Practices on RS Performance (Hit@20) with vs. without New Item Arrival Modeling (Dynamic User Preferences, Multi-Channel Retailer Dataset)



The results are under a setting where 50% of users are affected, algorithm is BPR, and the user preference changes together with $T_{DataReq}$.

Appendix W: Impact of Data Privacy Practices on Consumption Diversity

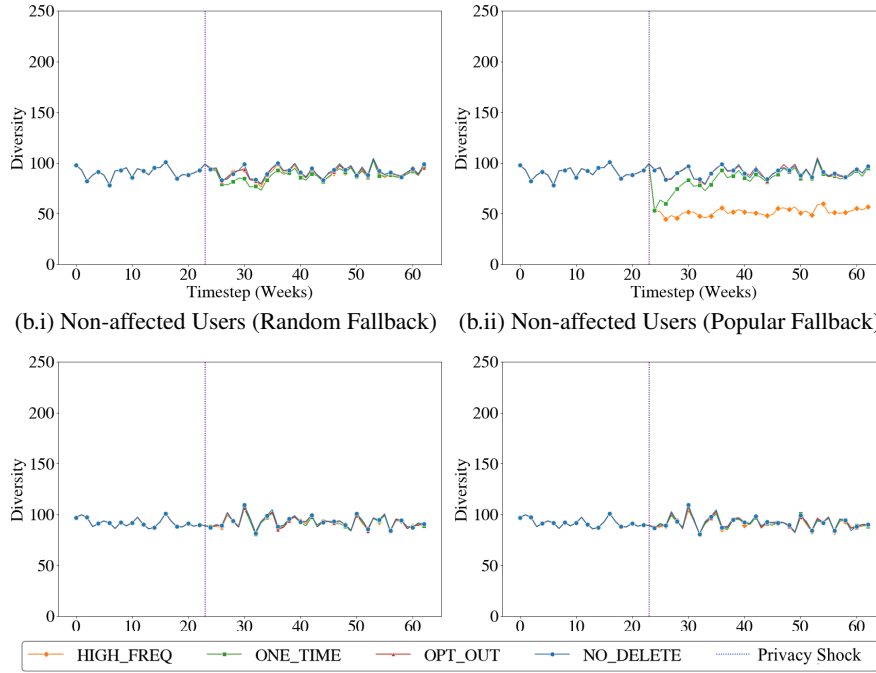
Although the main focus of our study is the impact of data privacy practices on the RS accuracy performance, data privacy practices may influence various item consumption patterns as well. Therefore, in this section we explore the impact of data privacy practices on the diversity of consumed items.

In particular, we focus on a popular measure of *aggregate diversity*, which is often measured by the total number of distinct consumed items (Adomavicius and Kwon 2011, Zhang et al. 2020), and we use a period of one week as a temporal granularity at which we measure consumption diversity over time.⁷ As part of our explorations under the stable user preference setting, as shown in Figure W58, we find that the consumption diversity does not change significantly under OPT.OUT. However, the impact of deletion-based practices (ONE.TIME and HIGH.FREQ) can vary substantially, primarily based on what *fallback* methods are used.⁸ We illustrate this using two fallback methods, *random* fallback and *popular* fallback, in Figure W58. Specifically, to users who do not have sufficient historical consumption data (due to their enacted privacy actions), random fallback recommends random items, while popular fallback recommends the top-20 items that are currently most popular across the entire user population (based on the historical consumption data that is available to RS). For users who have sufficient consumption data, we use the intended RS algorithm to provide personalized recommendations.

⁷To isolate the effect on consumption diversity solely due to RS, we focus only on the items that were consumed due to recommendations; other (e.g., need-based) consumptions are not influenced by RS.

⁸A fallback method is commonly used when the primary recommendation algorithm cannot produce meaningful results, typically because there is insufficient data about a user or item.

Figure W58 Impact of Data Privacy Practices on Consumption Diversity, Stable User Preferences
(a.i) Affected Users (Random Fallback) (a.ii) Affected Users (Popular Fallback)

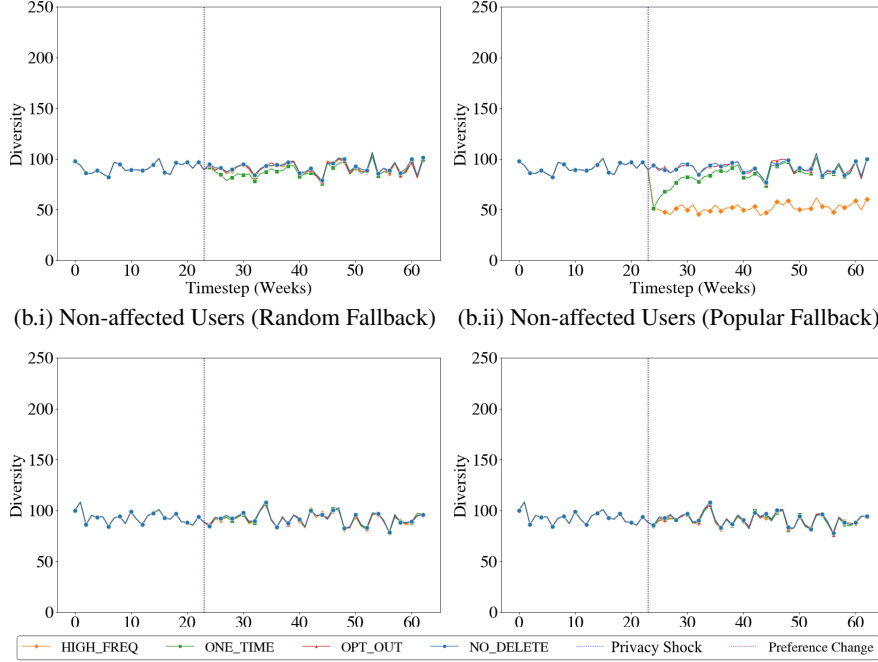


Note that across all figures, the algorithm is BPR, 50% of users are affected, and users' reliance on recommendations is 20%.

Results in Figure W58(a) show that, with random fallback, consumption diversity under HIGH_FREQ and ONE_TIME does not change substantially after the privacy shock; however, consumption diversity decreases significantly with popular fallback, as shown in Figure W58(b). This decrease is attributed to more concentrated recommendations on popular items due to the use of popular fallback, leading to more focused consumption on a smaller subset of items. Following the decrease, consumption diversity under ONE_TIME gradually recovers, as the use of popular fallback is reduced due to the alleviated data sparsity with gradual accumulation of personal data. In contrast, random fallback maintains the diversity near the maximum possible level throughout the simulation period due to completely random recommendations.

We also note that consumption diversity for non-affected users does not change substantially throughout the simulation period, as shown in W58(c-d), indicating a minimal spillover effect. Moreover, consumption diversity patterns are highly consistent across the stable and dynamic user preference settings, as can be seen by comparing Figures W58 and W59.

Figure W59 Impact of Data Privacy Practices on Consumption Diversity, Dynamic User Preferences
(a.i) Affected Users (Random Fallback) (a.ii) Affected Users (Popular Fallback)



Note that across all figures, the algorithm is BPR, 50% of users are affected, and users' reliance on recommendations is 20%.

Appendix X: Impact of Data Privacy Practices on Long-Tail Item Consumption

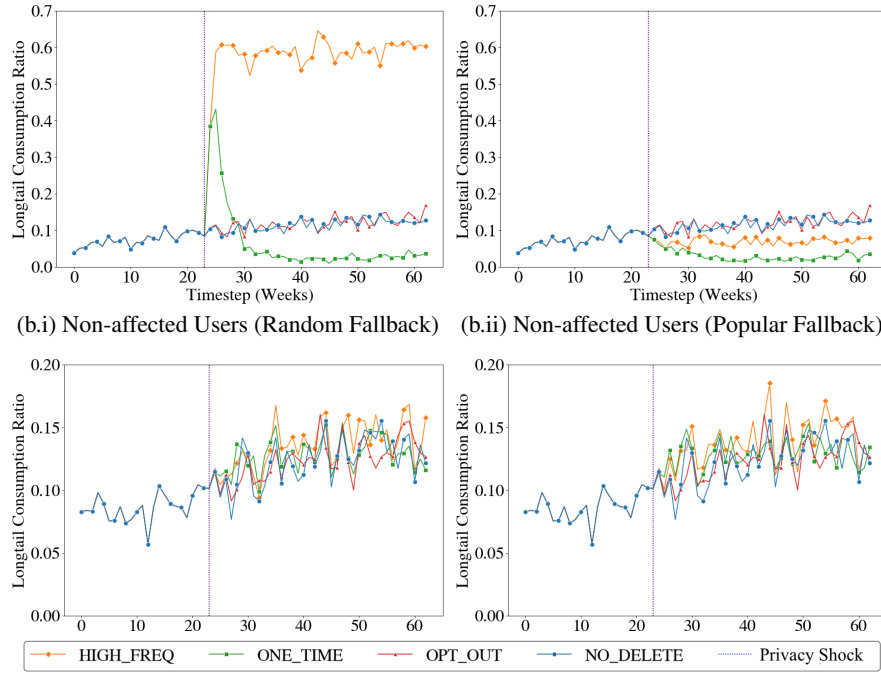
We define the set of long-tail items $I_{longtail}^t$ as the 80% least popular items in the training data of the RS at time t .⁹ We measure the long-tail consumption ratio (LCR) as follows:

$$LCR_t = \frac{\sum_{u \in U} |TrainData_u^t \cap I_{longtail}^{t-1}|}{\sum_{u \in U} |TrainData_u^t|}.$$

Note that item consumptions at t (i.e., $TrainData_u^t$) are made under the potential influence of RS, which was trained on data collected up to (and including at) $t - 1$. Thus, in the above definition, we categorize items as “long-tail” based on the training data collected at $t - 1$ to reflect what RS perceives as long-tail.

⁹We also explored using 70% and 90%, with results remaining qualitatively consistent.

Figure X60 Impact of Data Privacy Practices on Long-Tail Item Consumption, Static User Preferences
(a.i) Affected Users (Random Fallback) (a.ii) Affected Users (Popular Fallback)



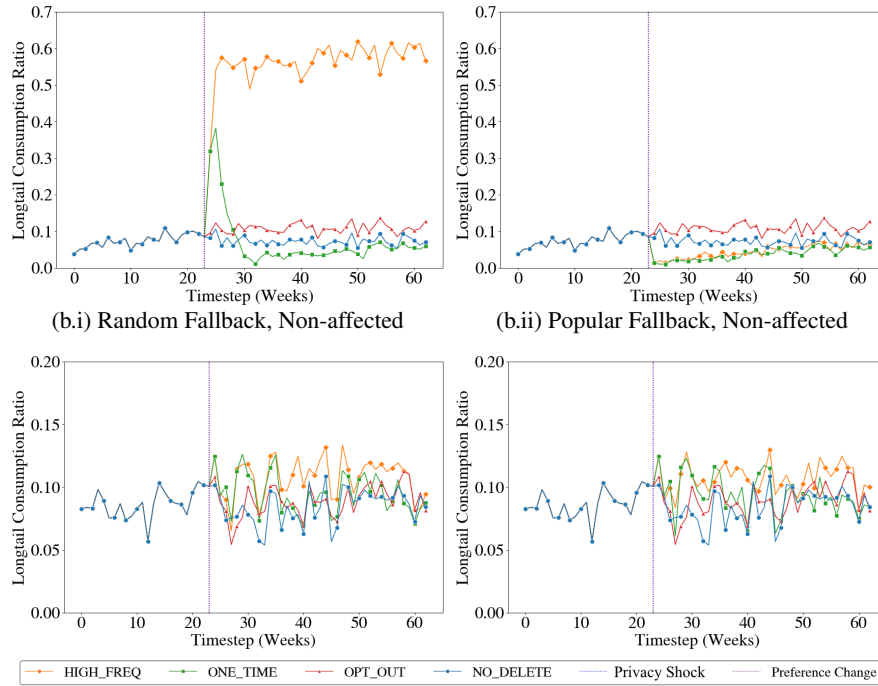
Note that across all figures, the algorithm is ALS, 50% of users are affected, and users' reliance on recommendations is 20%.

Figure X60 shows the long-tail consumption ratio under the stable user preference setting with two different fallback methods: random and popular. We find that the consumption longtailness does not change significantly under OPT_OUT, which remains consistent with the performance patterns measured by recommendation accuracy and consumption diversity. However, the impact of deletion-based practices (ONE_TIME and HIGH_FREQ) varies depending on the fallback method that is used by a given RS algorithm. With random fallback, long-tail item consumption of affected users under HIGH_FREQ and ONE_TIME rises right after the privacy shock. It is because RS loses personal data of affected users, and random fallback by definition will recommend items that have 80% probability of being long-tail. The long-tail item consumption ratio under ONE_TIME in Figure X60(a) gradually decreases after the initial surge as the personal data accumulates, due to the fact that many RS algorithms tend to recommend more popular items when having smaller amounts of data for personalized inference. In Figure X60(b), the long-tail item consumption drops significantly under ONE_TIME and HIGH_FREQ immediately after the privacy shock because popular fallback recommends popular items.

We also note that long-tail item consumption by non-affected users is not significantly impacted by data privacy practices, as shown in Figures X60b.i and X60b.ii, indicating a minimal spillover effect.

Long-tail item consumption under the dynamic user preference setting share highly similar patterns, as shown in Figure X61.

Figure X61 Impact of Data Privacy Practices on Long-Tail Item Consumption, Dynamic User Preferences
(a.i) Random Fallback, Affected (a.ii) Popular Fallback, Affected



Note that across all figures, the algorithm is ALS, 50% of users are affected, and users' reliance on recommendations is 20%.

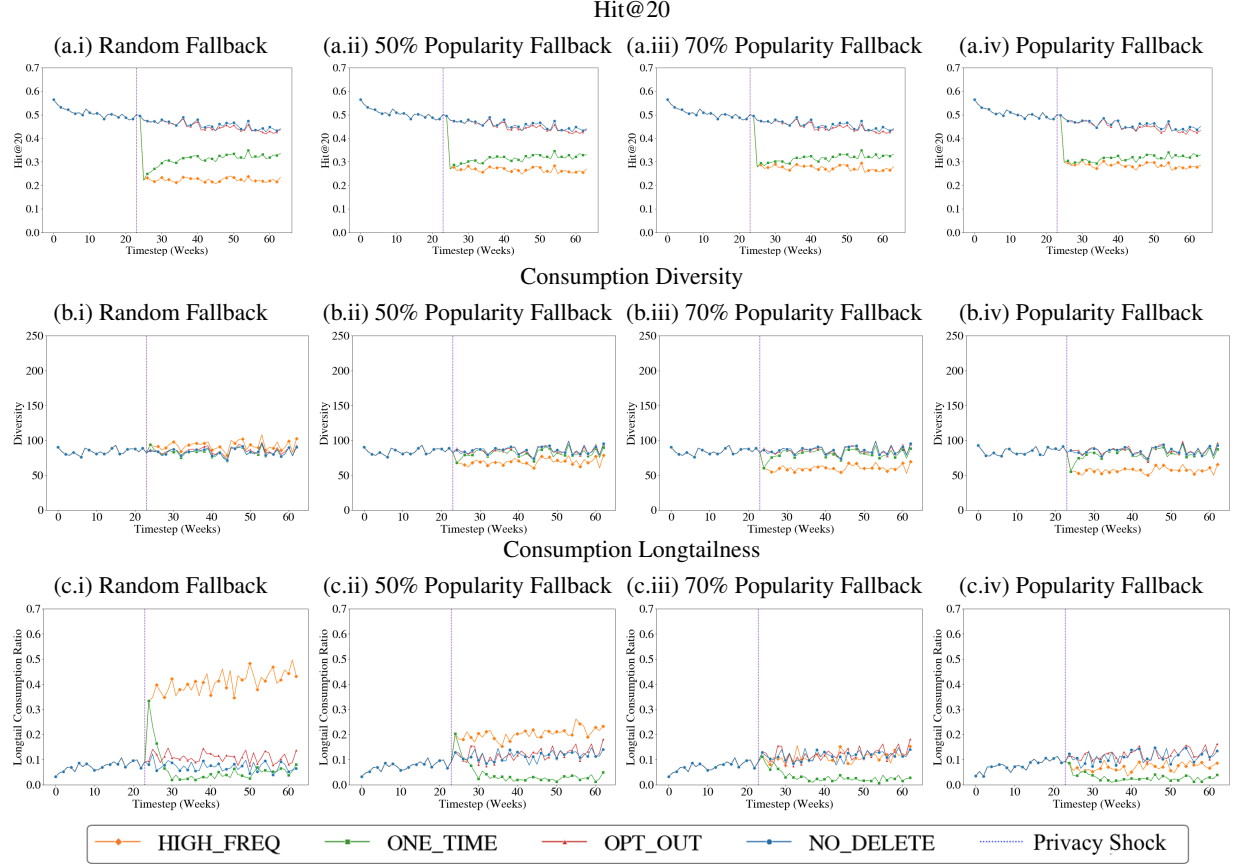
Appendix Y: Hybrid Fallback Methods

Appendices W and X show that the consumption diversity and longtailness can be strongly affected by the fallback methods, such as random fallback or popularity fallback. In this appendix, we further explore hybrid fallback strategies and assess the impact of different fallback strategies on key metrics of system performance (i.e., Hit@20) and consumption dynamics (i.e., diversity and longtailness).

In particular, we examine 4 types of fallback methods for users who do not have sufficient personal data for personalized recommendations: the Random Fallback (i.e., 100% of recommendations are randomly selected items), 50% Popularity Fallback (i.e., 50% of recommendations are popular items and 50% random items), 70% Popularity Fallback (i.e., 70% popular and 30% random), and Popularity Fallback (i.e., 100%

popular). We run simulations with the Multi-Channel Retailer seeding dataset and report results in Figures Y62 and Y63.

Figure Y62 Impact of Privacy Practices on RS Performance (Hit@20) with Different Fallback Methods (Stable User Preferences, Affected Users, Multi-Channel Retailer Dataset)



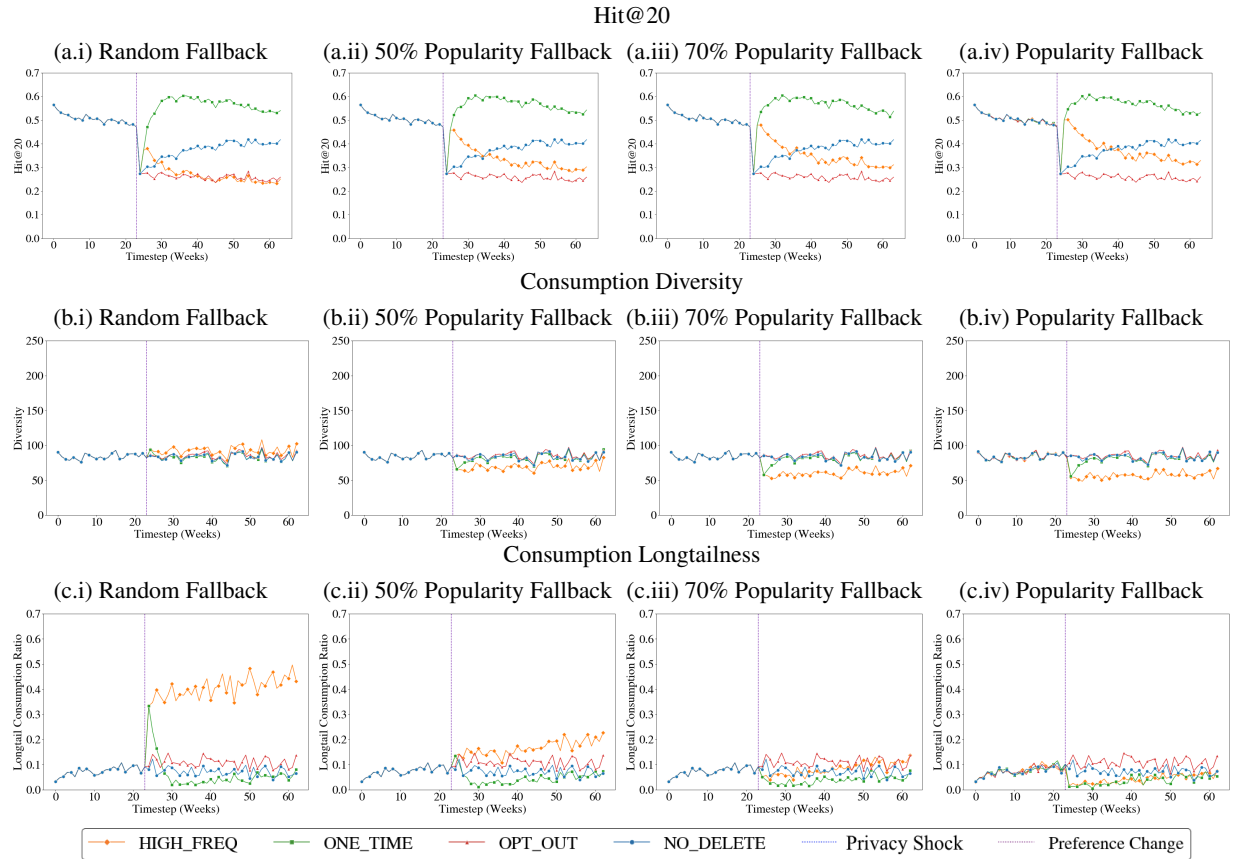
The results are under a setting where 50% of users are affected, algorithm is BPR, and the user preference changes together with $T_{DataReq}$.

Figure Y62 shows the simulation results under stable user preferences for different fallback methods. We observe that the impact of fallback methods mainly differs for the HIGH_FREQ practice, under which most affected users do not have any personal data, and thus the system needs to rely on fallback methods for recommendations. The choice of fallback methods hardly has any impact on system performance and consumption dynamics under other privacy practices because the system rarely rely on fallback methods in those scenarios. Importantly, moving from column 1 to column 4, we observe that the system performance under HIGH_FREQ improves as fallback methods include increasingly larger portion of popular items. In contrast, the consumption diversity and longtailness of HIGH_FREQ decrease as more popular items

are recommended. This is consistent with previous observations in Appendices W and X, confirming that popularity fallback methods tend to improve the system performance at the cost of consumption diversity and longtailness.

Figure Y63 shows the simulation results under dynamic user preferences for different fallback methods. The aforementioned HIGH_FREQ performance patterns are highly consistent across the stable and dynamic preference settings, validating robustness of the fallback-method-related findings.

Figure Y63 Impact of Privacy Practices on RS Performance (Hit@20) with Different Fallback Methods (Dynamic User Preferences, Affected Users, Multi-Channel Retailer Dataset)



The results are under a setting where 50% of users are affected, algorithm is BPR, and the user preference changes together with $T_{DataReq}$.

Appendix Z: External Validity of Simulated User Consumption Behavior

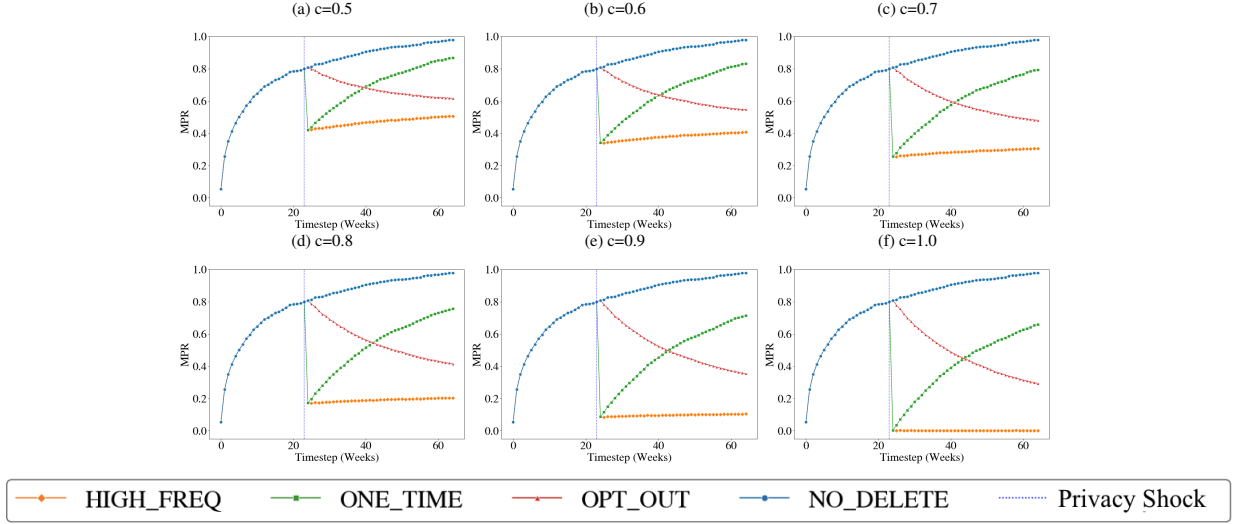
In this appendix, we show that the simulated user consumption behavior is a faithful reflection of real-world consumption patterns. We examine two fundamental aspects of consumption behavior: users' global ranking by their consumption frequency and items' global ranking by popularity. Together, these aspects provide an additional assessment of the external validity of the simulated user consumption behavior.

We compare the aforementioned consumption aspects in (a) the real-world seeding dataset vs. (b) the simulated dataset at time 0, before the intervention of a recommender system (i.e., with simulated dataset $ConsData_u^0$, the simulated consumption records without recommendations). Specifically, we obtained the users' global ranking by consumption frequency on these two datasets and assessed the rank correlation between them; same analysis is performed for items' global ranking by popularity. The ranking correlation (Spearman's r) of users' global ranking by consumption frequency is 0.8968 and that of items' global ranking by popularity is 0.8084, both statistically significant with $p\text{-value} < 0.001$. This shows that the simulated user consumption behavior strongly preserves the relative consumption frequency and item popularity from the seeding dataset on a global scale. These statistics together demonstrate that the simulated consumption behavior comprehensively reflects the real-world consumption behavior (observed in the seeding dataset), building strong external validity into the consumption dynamics of the simulation environment.

Appendix AA: Parameterizable Privacy Practices: Impact of Niche Threshold on the MPR Metric

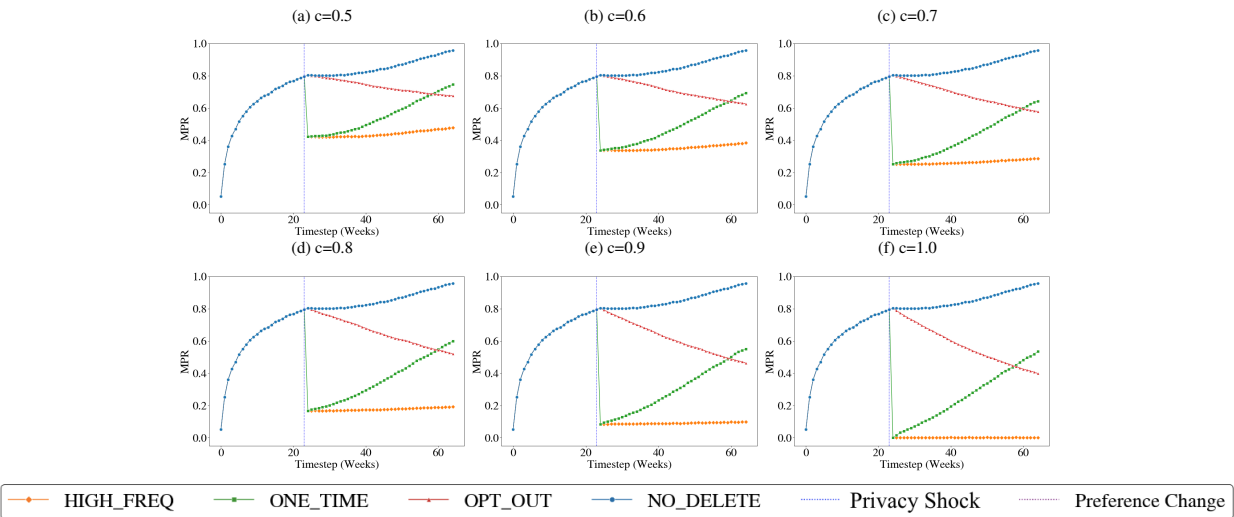
In Figures AA64 and AA65, we show longitudinal trends of the Mean Pairwise Revelation (MPR) metric for different values of niche threshold c under stable and dynamic user preference settings, respectively. The results remain highly consistent with Figure 9 in the main paper.

Figure AA64 MPR Trends for Data Privacy Practices with Varying Niche Thresholds, Stable User Preferences



Across all figures, 50% of users are affected, and the user reliance on RS is 20%. The algorithm is BPR. Setting $c = 1.0$ represents canonical, fully enacted (i.e., non-niche-based) data privacy practices.

Figure AA65 MPR Trends for Data Privacy Practices with Varying Niche Thresholds, Dynamic User Preferences

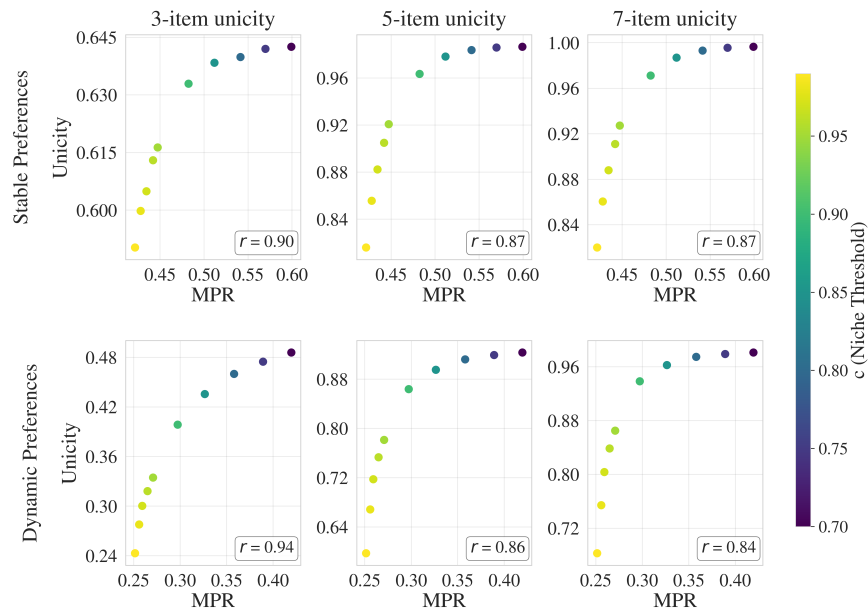


Across all figures, 50% of users are affected, the user reliance on RS is 20%, and the user preference change co-occurs with the privacy shock. The algorithm is BPR. Setting $c = 1.0$ represents canonical, fully enacted (i.e., non-niche-based) data privacy practices.

Appendix AB: Correlation between MPR and Re-identification Risk

Figure AB66 plots the Mean Pairwise Revelation (MPR) against a well-established re-identification risk metric, unicity. Unicity measures the average probability that a random set of s items in the user profile singles out that user from the other users in the data (De Montjoye et al. 2015, 2013). Thus, higher unicity means that the user is more likely to be singled out (i.e., re-identified) from the rest of the users on average, indicating a larger privacy risk. In our experiments using the multi-channel retailer dataset, we instantiate the s -item unicity measure with $s = 3, 5, 7$, and we use RP_ONE_TIME as a representative parameterizable privacy practice to illustrate the relationship between MPR and unicity across different niche threshold values in Figure AB66.¹⁰ For different values of niche threshold c , we report the average unicity and MPR across the post-privacy-shock simulation window (i.e., weeks 25 through 65, consistent with the main experiments in this study). Notably, MPR is strongly positively correlated with unicity. This empirically validates that MPR is an informative measure that is consistently reflective of re-identification risk. Also, as would be expected, the s -item unicity values are higher with larger s , because, on average, it is easier to re-identify a user with knowledge of more items in the user's profile.

Figure AB66 s -Item Unicity vs. Mean Pairwise Revelation (MPR) under RP_ONE.TIME



Note: r denotes the Pearson correlation between MPR and s -item unicity.

¹⁰Note that the MPR under stable and dynamic preferences in Figure AB66 correspond to the same MPR calculation under the same two preference setups in Figure 10(a-b.i) of the main paper.

References

- G. Adomavicius and Y. Kwon. Improving aggregate recommendation diversity using ranking-based techniques. *IEEE Transactions on Knowledge and Data Engineering*, 24(5):896–911, 2011.
- Y.-A. De Montjoye, C. A. Hidalgo, M. Verleysen, and V. D. Blondel. Unique in the crowd: The privacy bounds of human mobility. *Scientific reports*, 3(1):1376, 2013.
- Y.-A. De Montjoye, L. Radaelli, V. K. Singh, and A. S. Pentland. Unique in the shopping mall: On the reidentifiability of credit card metadata. *Science*, 347(6221):536–539, 2015.
- M. Deshpande and G. Karypis. Item-based top-n recommendation algorithms. *ACM Transactions on Information Systems (TOIS)*, 22(1):143–177, 2004.
- B. J. Dietvorst, J. P. Simmons, and C. Massey. Algorithm aversion: people erroneously avoid algorithms after seeing them err. *Journal of experimental psychology: General*, 144(1):114, 2015.
- X. He, H. Zhang, M.-Y. Kan, and T.-S. Chua. Fast matrix factorization for online recommendation with implicit feedback. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 549–558, 2016.
- J. M. Logg, J. A. Minson, and D. A. Moore. Algorithm appreciation: People prefer algorithmic to human judgment. *Organizational Behavior and Human Decision Processes*, 151:90–103, 2019.
- T. Lohrenz, K. McCabe, C. F. Camerer, and P. R. Montague. Neural signature of fictive learning signals in a sequential investment task. *Proceedings of the National Academy of Sciences*, 104(22):9493–9498, 2007.
- M. Ludewig and D. Jannach. Evaluation of session-based recommendation algorithms. *User Modeling and User-Adapted Interaction*, 28(4-5):331–390, 2018.
- G. Takács, I. Pilászy, and D. Tikk. Applications of the conjugate gradient method for implicit feedback collaborative filtering. In *Proceedings of the fifth ACM conference on Recommender systems*, pages 297–300, 2011.
- K. Yu, S. Berkovsky, R. Taib, D. Conway, J. Zhou, and F. Chen. User trust dynamics: An investigation driven by differences in system performance. In *Proceedings of the 22nd international conference on intelligent user interfaces*, pages 307–317, 2017.
- K. Yu, S. Berkovsky, D. Conway, R. Taib, J. Zhou, and F. Chen. Do i trust a machine? differences in user trust based on system performance. In *Human and machine learning: Visible, explainable, trustworthy and transparent*, pages 245–264. Springer, 2018.
- J. Zhang, G. Adomavicius, A. Gupta, and W. Ketter. Consumption and performance: Understanding longitudinal dynamics of recommender systems via an agent-based simulation framework. *Information Systems Research*, 31(1):76–101, 2020.
- Z. Zhang. Improved adam optimizer for deep neural networks. In *2018 IEEE/ACM 26th international symposium on quality of service (IWQoS)*, pages 1–2. Ieee, 2018.