http://pubsonline.informs.org/journal/isre

INFORMATION SYSTEMS RESEARCH
Vol. 31, No. 1, March 2020, pp. 76–101
ISSN 1047-7047 (print), ISSN 1526-5536 (online)

Consumption and Performance: Understanding Longitudinal Dynamics of Recommender Systems via an Agent-Based Simulation Framework

Jingjing Zhang,[a] Gediminas Adomavicius,[b] Alok Gupta,[b] Wolfgang Ketter[c,d]

[a] Department of Operations and Decision Technologies, Kelley School of Business, Indiana University, Bloomington, Indiana 47405; [b] Department of Information and Decision Sciences, Carlson School of Management, University of Minnesota, Minneapolis, Minnesota 55455; [c] Department of Management, Economics, and Social Sciences, University of Cologne, 50923 Cologne, Germany; [d] Department of Technology and Operations Management, Rotterdam School of Management, Erasmus University Rotterdam, 3062 PA Rotterdam, Netherlands

Contact: jjzhang@indiana.edu, http://orcid.org/0000-0002-6805-8685 (JZ); gedas@umn.edu, http://orcid.org/0000-0001-5251-5098 (GA); alok@umn.edu, http://orcid.org/0000-0002-2097-1643 (AG); ketter@wiso.uni-koeln.de, http://orcid.org/0000-0001-9008-142X (WK)

Received: August 16, 2016
Revised: March 6, 2018; February 12, 2019
Accepted: April 22, 2019
Published Online in Articles in Advance: March 18, 2020

https://doi.org/10.1287/isre.2019.0876

Abstract. We develop a general agent-based modeling and computational simulation approach to study the impact of various factors on the temporal dynamics of recommender systems' performance. The proposed agent-based simulation approach allows for comprehensive analysis of longitudinal recommender systems performance under a variety of diverse conditions, which typically is not feasible with live real-world systems. We specifically focus on exploring the product *consumption strategies* and show that, over time, user–recommender interactions consistently lead to the longitudinal *performance paradox* of recommender systems. In particular, users' reliance on the system's recommendations to make item choices generally tends to make the recommender system less useful in the long run or, more specifically, negatively impacts the longitudinal dynamics of several important dimensions of recommendation performance. Furthermore, we explore the nuances of the performance paradox via additional explorations of longitudinal dynamics of recommender systems for a variety of user populations and consumption strategies, as well as personalized and nonpersonalized recommendation approaches. One interesting discovery from our exploration is that a certain *hybrid* consumption strategy—that is, where users rely on a combination of both personalized- and popularity-based recommendations, offers a unique ability to substantially improve consumption relevance over time. In other words, for such hybrid consumption settings, recommendation algorithms facilitate the general "quality-rises-to-the-top" phenomenon, which is not present in the pure popularity-based consumption. In addition to discussing a number of interesting performance patterns, the paper also analyzes and provides insights into the underlying factors that drive such patterns. Our findings have significant implications for the design and implementation of recommender systems.

History: Alessandro Acquisti, Senior Editor; Jungpil Hahn, Associate Editor.
Supplemental Material: The online appendix is available at https://doi.org/10.1287/isre.2019.0876.

Keywords: dynamics of recommender systems • agent-based modeling • simulation • consumption strategies • prediction accuracy • consumption diversity • consumption relevance

1. Introduction and Motivation

Recommender systems are commonplace in the electronic marketplace. They provide personalized suggestions to individual consumers to help them discover information and items that are most relevant to their interests and needs. Such systems are especially valuable in complex electronic commerce markets that often involve millions of products to choose from. In practice, recommender systems have demonstrated strong influence on both consumers and retailers. For example, it has been reported that recommendations could account for 35% of product sales at Amazon.com (Marshall 2006). Netflix, the Internet television and movie streaming/rental company, has reported that about 75% of the content watched by its subscribers is suggested by its recommendation system (Amatriain and Basilico 2012).

Much research in the area of recommender systems has focused on developing and improving personalization techniques for making accurate predictions of user preference ratings for individual items. The developed prediction algorithms are typically evaluated in a fixed static setting on an existing rating data file using traditional evaluation approaches from machine-learning and data-mining literature, such as training/testing data split or cross-validation. In other words, a vast number of studies in the recommender systems literature have used existing offline data sets

collected from real-world systems for assessing the performance of algorithms. Such data sets typically represent a one-time snapshot of the system and can help make the best design choices (e.g., find the best-performing algorithm) for that specific situation. However, it often remains unclear how these design choices will affect users' consumption of items and interactions with the system, which in turn will influence the future performance of the system. In prior literature, the nature of long-term consumer–recommender interactions—that is, *temporal dynamics* of recommender systems—has been underexplored. Additionally, the static data sets only contain the actually observed interactions between users and items in a fixed system setting, and, therefore, we do not know the evolution possibilities and longitudinal dynamics of the system under other, alternative settings. There are many "what-ifs" to be explored in a system—for example, what might happen in the long run if the system ranks the recommended items differently? What if users rely more versus less heavily on the system for their item selections? To answer these questions, we need to analyze and understand the longitudinal dynamics of recommender systems.

In addition, real-world recommender systems' implementations have to focus not only on recommendation accuracy, but rather balance a number of objectives, including accuracy, diversity, relevance, novelty, popularity, scalability, adaptability, awareness, explanations, social values, similarity, etc. (Amatriain and Basilico 2012). In other words, most serious real-world recommender systems inevitably have to navigate various tradeoffs in order to balance many (often conflicting) goals. Thus, understanding longitudinal performance tradeoffs that are inherent in the interactions between the users and the system constitutes an important and practical research topic. Such understanding would enable designers to anticipate the temporal changes in the system, make strategic design choices, and maximize the long-term value of the system. This paper demonstrates that *consumption strategies* (based on different degrees of users' reliance on recommendations when choosing which items to consume) turn out to have a major influence on some of these tradeoffs and, more generally, on the longitudinal performance dynamics of the system. Thus, understanding the impact of consumption strategies (and some of the underlying reasons behind it) represents an important contribution of this work.

An ideal approach for studying recommender systems' longitudinal dynamics would be to perform large-scale longitudinal field experiments in real-world settings. Depending on the research question, such live experiments may need to isolate or control for the effects of numerous (potentially confounding) factors, such as the composition of user and item populations (e.g., user population heterogeneity and distribution of item popularity), users' "lifecycle" characteristics (e.g., users' consumption frequency), users' consumption strategies (e.g., how users choose items), and so forth. Controlling for some of these effects may be difficult in real-world settings; thus, conducting such field experiments may not be feasible or could be prohibitively expensive. Therefore, we propose and develop an *agent-based modeling and simulation approach* to investigate the temporal dynamics of recommender systems.

Agent-based modeling methodology allows one to create computational models that simulate the simultaneous actions and interactions of artificial agents, which typically represent individual entities (e.g., users), in an attempt to recreate and predict the appearance of complex phenomena (Miller and Page 2007). In the recommender systems context, agent-based simulation is typically much cheaper and faster than large-scale field experiments with real users and systems. Besides, simulation can provide a rich environment for exploration, allowing for numerous experiments under a variety of settings, which makes it possible to identify, explore, and separate effects of different factors. Simulation can uncover key insights that could then be further analyzed by using additional methodologies (i.e., more targeted field experiments and econometric analyses) and, thus, can be used in conjunction with other methods such as field experiments and analytical modeling. These characteristics of agent-based simulation make it well-suited for our exploration of longitudinal dynamics of recommender systems.

The contributions of the paper are threefold. First, we introduce a general-purpose agent-based modeling framework for exploring the longitudinal dynamics of recommender systems. Little research has systematically examined the longitudinal aspects of recommender systems and how users' interactions with the system influence the system's performance over time. One reason for this is because studying longitudinal dynamics is inherently difficult, as there are so many intertwined factors involved in the user–recommender ecosystem. As mentioned earlier, it is often difficult to isolate (or control for) the effects of various factors analytically or experimentally. Thus, building upon prior studies that demonstrated the usefulness of simulation to examine some specific recommender-systems-related research questions (e.g., Fleder and Hosanagar 2009, Prawesh and Padmanabhan 2014, and Jannach et al. 2015), this research introduces (and advocates for) a general-purpose, comprehensive, low-cost, and risk-free simulation framework that can be used to explore various emerging phenomena resulting from user–recommender

interactions by allowing researchers and practitioners to manipulate all the different aspects of recommender systems as well as simulate various canonical user consumption behaviors.

A second key contribution of this paper is that we use the simulation framework to study how users' *consumption strategies* (specifically, with respect to the level of users' reliance on recommendations) influence the longitudinal performance dynamics of the system. The user's inherent consumption strategy is not something that can be controlled by companies using field experiments; thus, agent-based modeling provides a useful tool for researchers to study the impact of consumption strategies. In our paper, we demonstrate an interesting paradox that users' high reliance on the recommender system actually provides highly suboptimal performance outcomes (i.e., smaller benefits for the system's performance) in the long run. The simulation framework allows us to perform a more in-depth investigation of all the process-oriented metrics to provide an in-depth understanding of why and how this performance paradox occurs. Our analysis suggests that the different consumption strategies (e.g., select items randomly versus based on recommendations) lead to fundamental changes in the structural and value distributions of the rating data, which subsequently serve as inputs for the recommendation algorithms, and therefore affect the future performance of the recommender systems. The theoretical understanding of the longitudinal nature of recommender systems represents a major contribution of this work.

Third, we provide explorations of several additional factors about user populations and types of recommender systems (e.g., personalized versus popularity-based recommendations, nonhomogeneous user populations, top-N, and classification systems). Our exploration reveals some highly interesting patterns. For example, the *hybrid* consumption strategy based on personalized and popularity-based recommendations offers a unique combination that is able to substantially improve the relevance of selected/consumed items over time. It is because recommender systems can discover and popularize the "good-quality" items that are appealing to a significant number of people (but that are not popular at this time), thus helping these items climb up in the popularity rank list. This results in a general increase of item quality in the list of most popular items (and, hence, consumption relevance) over time. In other words, in hybrid consumption settings, recommendation algorithms facilitate the general "quality-rises-to-the-top" phenomenon, which is not present in the pure popularity-based consumption; this is another important finding facilitated by the simulation-based approach. In addition to discussing a number of key performance patterns, the paper also analyzes and provides insights into the underlying factors that drive these patterns.

2. Related Work

2.1. Recommender Systems

Recommender systems make recommendations by inferring users' preferences for items based on users' feedback on previously consumed items. The most common approach to modeling users' preferences for items is via numeric *ratings*. For example, Amazon users are asked to rate the items they have purchased on a 5-star scale (with 1 being "I hate it" and 5 "I love it"). A recommender system then analyzes patterns of users' past ratings and predicts users' preferences on unconsumed items.

There has been a significant amount of research on the design and implementation of recommender system algorithms, with the goal of improving accuracy performance (see Adomavicius and Tuzhilin 2005 for an overview). Among the different types of algorithms, the *neighborhood-based Collaborative Filtering* (CF) approach, which predicts unknown ratings of a user (or item) based on the ratings of the "nearest-neighbor" users (or items) with similar rating patterns (e.g., Resnick et al. 1994, Sarwar et al. 2001, and Linden et al. 2003), represents arguably the most popular recommendation approach. The neighborhood-based CF technique can be user-based, if the ratings of the nearest-neighbor users are used to predict unknown ratings (Resnick et al. 1994, Konstan et al. 1997), or item-based, if the ratings of the nearest-neighbor items are used to predict unknown ratings (Sarwar et al. 2001, Linden et al. 2003). The neighborhood-based CF approaches are widely used in real-world applications; for example, Amazon is a well-known early adopter of the item-based CF algorithm (Linden et al. 2003, Smith and Linden 2017). We use item-based CF as our recommendation algorithm choice in our simulation experiments.

In terms of performance evaluation, prior literature characterizes performance as a multifaceted construct that can be evaluated along a number of dimensions (Herlocker et al. 2004). The evaluation approach can be broadly categorized as either system-centric or user-centric (Herlocker et al. 2004, Pu et al. 2011). The vast majority of recommender systems literature has focused on system-centric evaluation, which considers algorithmic performance and uses measures such as accuracy, diversity, and coverage. In contrast, user-centric evaluation focuses on user experience and uses measures such as relevance, usability, and satisfaction. Our simulation framework measures the performance on several key dimensions—both system-centric and user-centric—including systems'

prediction capability (e.g., accuracy), item-discovery capability (e.g., aggregate item diversity), and users' consumption outcome (e.g., relevance of consumed items). Many real-world systems, including the ones used by Netflix and Amazon, strive to balance different dimensions in their personalization services (see, e.g., Amatriain and Basilico 2012 and Smith and Linden 2017).

In addition to the extensive literature on the design of recommender systems, there is an increasing body of research that has examined the *impact* of recommender systems from various perspectives, thus taking more of a longitudinal view. From the business perspective, research has shown that recommenders can positively affect product sales and Web impressions (e.g., Ansari et al. 2000 and De et al. 2010). Fleder and Hosanagar (2009) show that recommender systems can lead to a "rich-get-richer, poor-get-poorer" effect for products, thus resulting in a decrease in aggregate sales diversity over time. Jannach et al. (2015) further empirically explore the effects of different recommendation algorithms on item popularity as well as several other outcomes. In addition, Prawesh and Padmanabhan (2014) demonstrate that, in non-personalization-based recommender systems, recommending top-N popular items could lead to popularity amplification of these items and make the system susceptible to manipulations. From the user perspective, product recommendations help reduce users' search costs (Brynjolfsson et al. 2011). However, users' interactions with recommender systems may have unintended effects on user preferences and economic behavior. In particular, observing the system-predicted ratings can bias users' subsequent judgements for products (Cosley et al. 2003, Adomavicius et al. 2013), as well as economic behaviors (Adomavicius et al. 2018). Our work also takes a longitudinal perspective of the recommender systems performance, by focusing on the underexplored issue of how the performance is affected by users' consumption strategies over time. The proposed simulation framework not only provides a promising opportunity to perform an in-depth, systematic study of this issue, but also to uncover the potential mechanisms through which these effects may occur.

Lastly, the recommender systems literature that uses the multiarmed bandit—or, more generally, reinforcement learning (Sutton and Barto 1998)—perspective has a potentially relevant connection to this work, as it focuses on the classic exploration–exploitation dilemma of how the system should take actions in order to optimize some longitudinal (or cumulative) aspects of performance. Exploitation implies providing users with the most relevant recommendations (i.e., to gain more value at the current point), whereas exploration means providing users with possibly suboptimal recommendations that can help the recommendation algorithm acquire the most informative new knowledge about users' preferences (i.e., to be able to gain more value in the future). For example, Li et al. (2010) proposed a contextual-bandit approach to personalize the recommendations based on contextual information and user feedback in order to maximize users' clicks on the recommended news articles, and Zeng et al. (2016) further incorporated the time-varying contextual information to maximize the cumulative value of recommendations. In contrast to the multiarmed bandit literature on recommender systems, this study focuses not on new algorithm development, but rather on understanding the longitudinal performance implications of different user behaviors (e.g., consumption strategies) for any given algorithm of interest.

In summary, there has been a substantial amount of work in the area of recommender systems that has focused on algorithmic development and the influence of recommendations on businesses and users. However, little research is available on understanding how the *longitudinal dynamics* of recommender systems themselves is influenced by users' continuous interactions with the system. Our work sets out to address this gap by investigating the impact of users' consumption strategies on the long-term performance of recommender systems. We show that users' reliance on the recommender systems leads to a performance paradox between the consumption relevance and future performance of the recommender systems. We also discuss a number of other longitudinal performance patterns arising from users' consumption strategies and provide analysis of key underlying factors behind these patterns.

2.2. Agent-Based Modeling and Simulation

We use agent-based modeling methodology to simulate the ongoing interactions of multiple agents (i.e., users) with the recommender system to investigate the system's temporal dynamics. The simulation process is one of emergence from the micro (i.e., individual) level to a macro (i.e., aggregated) level. In agent-based modeling, the agents are software entities that carry out some operations on behalf of a user or another program with some degree of independence or autonomy, and, in so doing, employ some knowledge or representation of the user's goals or desires (Wooldridge and Jennings 1995). As summarized by Miller and Page (2007, chapter 6), the advantages of agent-based models over traditional

experiments include being "flexible, process oriented, timely, adaptive, inherently dynamic, heterogeneous, scalable, repeatable, recoverable, constructive, and low cost."

Agent-based systems have been heavily used in the social-science community to model complex adaptive systems. In the Information Systems literature, prior studies have used agent-based models to simulate auction mechanisms to analyze auctioneer and bidder strategies and tradeoffs (e.g., Bapna et al. 2003 and Bapna et al. 2008); to simulate online communities to understand the design factors that lead to online community success (e.g., Ren and Kraut 2014); and to solve trading-agent problems and analyze tradeoffs of trading strategies in complex and uncertain environments (such as a supply chain environment in which agents must compete with each other in both procurement and sales markets while simultaneously managing inventories, fulfillment, and a manufacturing process) (e.g., Ketter et al. 2012 and Ketter et al. 2016a). Prior work on agent-based virtual worlds (ABVWs) brings real-world modeling aspects to the foreground to evaluate possible futures and tradeoffs of high-complexity environments and potential paths to these futures (Chaturvedi et al. 2011).

Our simulation platform for recommender systems can be categorized as a conventional ABVW to perform controlled experiments in pursuit of explanatory theories. The different user consumption strategies that we study are a variant of the Competitive Benchmarking method (Ketter et al. 2016b). We conduct experiments by simulating agent behaviors across many alternative settings and scenarios and observe the longitudinal dynamics of recommendation systems as a result of these agents' actions. In simulation-based research, there is a tension between real-world fidelity and model parsimony and elegance. We mitigate this tension by seeding the simulation environment with real-world application settings, which we discuss in detail in Section 3. Using an agent-based modeling framework offers three benefits for this research. First, we can implement and test various recommendation settings using a simulation approach. Second, heterogeneity in inherent user preferences and consumption strategies can be easily accommodated—that is, controlled for and experimented with. Third, we can readily explore a variety of different settings to disentangle the effects of various factors on the longitudinal dynamics of recommender systems.

3. Simulation Framework for Recommender Systems

Before describing the modeling and simulation choices made for our specific research study, in the next subsection, we present a brief overview of the general simulation framework that can be used to explore a wide variety of issues regarding temporal dynamics of recommender systems.

3.1. General Procedure and Main Components

Our general simulation framework reflects the traditional recommendation scenario, where individual users are able to consume individual items, possibly after observing item recommendations provided by the recommender engine. After consuming an item, each user is able to provide feedback to the system regarding the user's actual preference for that item (i.e., how relevant the item was to the user). This feedback is then used by the engine to further improve its subsequent recommendations to users, leading to new item consumptions, etc. Correspondingly, our framework consists of three major components that model the state of the entire simulated system—*item population*, *user population*, and *recommender engine*—as summarized in Table 1. In addition, our framework includes an *iterative simulation procedure* that advances the system state temporally (step by step), as presented in Algorithm 1. In this subsection, we present key aspects of each major component needed to properly model longitudinal recommender system behavior in a variety of settings; however, each component can be further customized and extended.

When modeling temporal dynamics of recommender systems, key aspects of the item population component include *item arrival*, *item lifespan*, and *item content* (see Table 1(a)). The first two aspects control the changes in item population over time—that is, item additions (e.g., new releases) and departures (e.g., item discontinuation). The third aspect, modeling item content distribution, allows one to control for heterogeneity of items available for consumption (e.g., having similar versus diverse items). Also, when modeling the item population (as well as the user population), it is important to model the key aspects both at the population level and at the individual level, as reflected in Table 1. For example, item content needs to be properly represented in each individual item as well as initialized (seeded) according to population-level considerations.

We model the *user-population* component using seven key aspects of user behavior as related to recommender systems (see Table 1(b)). Specifically, and similarly to the item population modeling, *user arrival* and *user lifespan* model the user population over time—that is, user arrivals to and departures from the system. As not all users typically like all items, modeling individual *user preferences* allows one to control the heterogeneity of the user population. Users' consumption of items is a major element of users'

Table 1. Major Components of the General Simulation Framework

Panel A. Item population

Aspect	Population-level modeling considerations	Individual agent modeling (item agent state)
Item arrival	*Item arrival rate* represents the cadence with which new items are introduced.	ItemArrivalTime$_i$: The date that item i is added to the system.
Item lifespan	*Item lifespan distribution* represents heterogeneity among items in terms of how long they will be available to be recommended or consumed.	ItemLifespan$_i$: The length of time item i will be available in the system.
Item content	*Item content distribution* represents heterogeneity of the item population in terms of their content.	ItemContent$_i$: Content representation of item i, for example, as a vector of item features/characteristics.

Panel B. User population

Aspect	Population-level modeling considerations	Individual agent modeling (user agent state)
User arrival	*User arrival rate* represents the cadence with which new users arrive to the system.	UserArrivalTime$_u$: The date that user u arrives to the system.
User lifespan	*User lifespan distribution* represents heterogeneity among users in terms of how long they will stay with the system (i.e., reflects users' churn rate).	UserLifespan$_u$: The length of time user u will stay with the system.
User preferences	*User preference distribution* represents heterogeneity of user population in terms of their preferences.	UserPrefs$_u$: Representation of user tastes/preferences, for example, as a vector of preferences for different content features.
User consumption frequency	*User consumption period distribution* represents the heterogeneity of users in terms of their consumption frequency (i.e., expected timespan between two consumptions).	UserConsPeriod$_u$: Representation of length of time between user u's two consecutive consumptions. (Also, next scheduled consumption for user u is referred to as UserNextCons$_u$ value.)
User consumption strategy	*User consumption strategy distribution* represents heterogeneity of users based on how they make item selection for consumption.	UserConsStrategy$_u$: User u's consumption strategy—that is, the strategy with which an item is selected for consumption (e.g., out of the available recommendations).
User feedback	*User feedback distribution* represents heterogeneity of users in terms of how they calculate their feedback to the system (i.e., the actual preference rating) upon consuming an item.	UserFeedback$_u$: Function for calculating an actual preference that user u would report after consuming any given item i, typically modeled as a degree of fit between UserPrefs$_u$ and ItemContent$_i$.
User feedback likelihood	*User feedback likelihood* represents heterogeneity of users in terms of how likely they are to provide feedback to the system.	UserFeedbackLikelihood$_u$: Representation of user u's likelihood (probability) of submitting feedback on newly consumed items.

Panel C. Recommender engine

Aspect	Recommender engine modeling
[Functionality] Rating prediction	PredictRatings: Rating prediction algorithm(s), which are used to calculate system-predicted (personalized or nonpersonalized) ratings for not-yet-consumed items.
[Functionality] Recommendation generation	GenerateRecs: Recommendation generator(s)—that is, which are used to produce recommendations from predicted ratings in various ways, including as ranked list of all items, top-N items, classified categories of items (relevant versus irrelevant items), etc.
[Functionality] Performance measurement	Evaluate: Performance metrics, for measuring different aspects of recommender systems performance, for example, accuracy, diversity, novelty, serendipity, revenue, etc.
[System state at time t] Known historical ratings	Available ratings $(R^t)_{ui}$, representing user-item consumption history (i.e., a database of known user-item ratings at time t), which is the main data source for recommendation algorithms. Can be initialized to represent desired historical rating distributions.
[System state at time t] System-predicted ratings	Rating predictions $(P^t)_{ui}$, representing the recommender system's predictions of all (unknown) user–item ratings at time t; P^t = PredictRatings(R^t).
[System state at time t] System recommendations	Recommendations $(L^t)_u$, representing system's recommendations that are available to any given user u at time t; that is, L^t_u = GenerateRecs(P^t_u).

interactions with recommender systems, and, thus, we model both *user-consumption frequency* (i.e., how often individual users tend to consume items) and *user-consumption strategy* (i.e., how individual users select an item for consumption) aspects. Finally, because recommender systems rely on feedback from the users (i.e., how much users liked the consumed items, which serves as the "ground truth" for the recommendation algorithms), it is crucial for the simulation framework to be able to calculate the actual preference rating of any user for any item as needed. This is explicitly modeled as the *user-feedback* aspect, which reflects the degree of fit between the user preferences and item content. The proposed framework also allows one to model *user-feedback likelihood*—that is, how likely users are to provide feedback after any given item consumption. In summary, users are modeled as populations of individual agents, which are initialized according to desired population-level considerations, but each agent then behaving autonomously as prescribed only by their own specific characteristics discussed above.

Our modeling of the recommender engine component (see Table 1(c) for an overview) is based on the traditional notion of *ratings* to represent user preferences for items. The known ratings that a user provided in the past are used as inputs to the recommendation algorithm that estimates ratings for the items that the user has not yet consumed. To define the main elements more formally, let U be the set of all users and I be the set of all possible items that can be recommended. The entire user–item space is then denoted as $S = U \times I$, and R_{ui} represents the rating that user u gave item i. However, at any given time, R_{ui} values are known only for a limited subset of the whole user–item space. Then, the key recommender engine tasks are to *predict* rating values, denoted as P_{ui}, for unknown $(u, i) \in U \times I$ pairs and, based on rating predictions, to provide some recommendations L_u to each user u (e.g., a list of items ordered based on predicted rating value, as was done in this study). Thus, at any given time t, the recommender engine's state can be represented by the following three key elements: set of known historical user–item ratings for previously consumed items (R^t), set of system-predicted ratings for previously unconsumed items (P^t), and the set (or list) of resulting system's recommendations to users (L^t). To manage this state, the recommender engine component is modeled with three main functionalities: *rating prediction*—that is, algorithms for predicting users' preference ratings for not-yet-consumed items (for calculating P^t based on R^t); *recommendation generation*—that is, strategies for producing item suggestions based on rating predictions (for computing L^t); and *performance measurement*—that is, evaluating various aspects of system's performance based on specific metrics of interest. When items or users depart from the system, their ratings can still be used for making predictions, but the system will no longer suggest these items or make recommendations to these users.

We use a discrete-time *iterative simulation procedure* to advance the system state from one time period to the next, as described in Algorithm 1, where each period represents some desired temporal granularity (e.g., day or hour). Prior to the main simulation, the three major components—that is, item population, user population, and recommendation engine—are initialized according to the desired distributional characteristics (i.e., Step 0 in Algorithm 1). The main simulation procedure consists of iterating the following two steps at each time period t. Step 1 is a "user–system interaction" step, where each user that is due for consumption at current time selects an item for consumption using her own consumption strategy, consumes the item, and provides feedback (actual rating) about the consumed item back to the system. Step 2 is a "system update" step, where the recommender system's state is updated based on newly submitted ratings (i.e., rating predictions, recommendations, and performance metrics are recalculated), completing a feedback loop that is critical to the recommender system's use and value; both user and item populations are also updated as necessary (e.g., based on new arrivals or lifespan restrictions).

The general simulation framework represents a comprehensive testbed for studying a rich set of longitudinal recommender system behaviors and, as with any simulation-based approach, provides multiple major benefits. It is readily replicable and can be rerun multiple times to obtain average performance estimates across multiple runs. Also, every aspect of the framework—from the major performance metrics to any internal state variables—can be tracked throughout the simulation, logged, and subsequently inspected and analyzed. Furthermore, it is highly customizable and can be instantiated (as well as expanded or reduced) in numerous ways, depending on the desired problem setting. For example, one could focus on different recommendation algorithms, different performance evaluation metrics, different ways of presenting recommendations to users, different user consumption strategies, etc. Also, one could vary modeling sophistication as needed, by including stochasticity (e.g., randomness in arrival rates or consumption frequencies), explicit temporal trends (e.g., preference evolution of users), context awareness, more sophisticated behavioral heuristics, etc., as part of key aspects of the major components.

Algorithm 1 (Overview of Iterative Simulation Procedure)

INITIALIZATION (at time period $t = 0$)

[Step 0: System Initialization]

Initialize the desired item population based on the item distributional parameters.

Initialize the desired user population based on the user distributional parameters.

Initialize available rating data $\boldsymbol{R}^0$.

Calculate item predictions $\boldsymbol{P}^0 = \text{PredictRatings}(\boldsymbol{R}^0)$ and prepare recommendations for each user u, i.e., $L_u^0 = \text{GenerateRecs}(P_u^0)$.

Evaluate recommendation performance (Evaluate).

MAIN SIMULATION (perform Steps 1 and 2 at each time period $t = 1, 2, 3, \ldots$)

[Step 1: Main Interaction/Consumption/Feedback Process]

Use all currently available rating data at current time t: $R^t = R^{t-1}$.

For each user u who is due for an item consumption at time t (i.e., where $\text{UserNextCons}_u = t$):

User u selects item x for consumption from recommendations L_u^{t-1} in accordance to her consumption strategy ($\text{UserConsStrategy}_u$).

User u consumes item x and calculates her preference rating $r = \text{UserFeedback}_u(x)$.

Determine whether rating r would be submitted as feedback to the recommender system based on $\text{UserFeedbackLikelihood}_u(x)$; if so, update known rating data to include the new rating, i.e., $R_{ux}^t = r$.

Update next consumption time for user u, i.e., $\text{UserNextCons}_u = \text{UserNextCons}_u + \text{UserConsPeriod}_u$.

[Step 2: System Update]

Update the item population based on the item distributional parameters (item arrivals, departures, etc.).

Update the user population based on the user distributional parameters (user arrivals, departures, etc.).

Recalculate item predictions $P^t = \text{PredictRatings}(R^t)$ and prepare recommendations for each user u, i.e., $L_u^t = \text{GenerateRecs}(P_{ui}^t)$.

Evaluate recommendation performance (Evaluate).

The rest of Section 3 discusses the instantiation of this general simulation framework for our specific research study on exploring the impact of consumption strategies on recommender systems performance.

3.2. Modeling Items

A typical way to model the item content is by using a K-dimensional array that represents different features of the item. Using movies as an example, the item features could represent various explicit movie attributes representing, say, the presence (or absence) of different genres, directors, and actors of the movie. For instance, if the first dimension of the array represents the comedy genre, the high value on this dimension would mean that the movie has high comedy content.

In our model, for generalizability and abstraction convenience, we choose to represent item content using an array of continuous *latent* factors (as opposed to observable explicit features). That is, item i is associated with item-feature vector $\textit{Item content}_i = (q_1, \ldots, q_K)$. Latent factors are not directly observed, but are rather inferred through a mathematical model from variables that are directly measured—that is, they are abstract representations of the actual content of the item. To achieve a realistic distribution of item content across the entire item population, the latent item-content vectors are initialized by the results of a matrix factorization model based on real-world rating data sets. We will elaborate on our approach in Section 3.5.

We note that using latent factors is not the only possible item representation, but it is our implementation choice for the following reasons. In particular, using latent factors can reduce the dimensionality of data. A large number of observable variables can be aggregated by a mathematical model into a much smaller set of latent variables to represent the underlying concepts. Also, latent factors are often used in statistical modeling techniques as a mathematical convenience where they often are not of primary interest—that is, where "measuring" them is not a goal. In our simulation framework, item content is used to compute the relationships among entities (such as content similarity between items, or estimated user preference on items). Our framework is intended to be generalizable to different types of products (e.g., movies, songs, and books), and therefore using latent factors allows us to abstract away from the explicit, highly application-specific attributes and to be able to apply the same simulation framework to different types of items.

Finally, for this study, we use a fixed set of items (as well as a fixed set of users) that are available for the entire simulation period—that is, there are no item (and user) arrivals or departures during the intermediate times. The main reason is to focus on isolating the effect of users' consumption strategies (as expressed by different levels of reliance on recommender systems) on the longitudinal performance dynamics and, hence, to avoid complex additional potential interactions, complications to experimental design (e.g., the need to provide proper allowances for these "cold-start" items

and users in recommendation and performance evaluation procedures), and potential confounding effects toward our research question.

3.3. Modeling Users

Users are modeled as autonomous agents in our simulation framework. We model the behaviors of individual agents and observe the collective effect of agent actions. Below, we discuss the specific instantiations of the key aspects of user agent model used in our study: user preferences, user feedback, user consumption frequency, and user consumption strategy.[1]

3.3.1. User Preferences. Similar to an item's content, each user's preference profile is modeled as an array of K values that represent the user's preferences for the corresponding K item features. Using the earlier example, if the first dimension of the array represents the comedy genre, the high value on this dimension would mean that the user has high preference for comedy content. The intuition behind using the same exact set of dimensions to model *both* users and items is that the association (or alignment) between user preferences and item features can then represent how a user rates an item. For example, a user with high preference for comedy is likely to give a high rating to a movie that has lots of comedic content. Similarly to item content, we use *latent* factors to represent user preferences (as opposed to observable explicit features). That is, user u's preferences for items are modeled as a vector of latent factors $User\ prefs_u = (p_1, \ldots, p_K)$. Finally, to achieve realistic distributions of user preferences across the entire user population, the latent user preference vectors are initialized by the results of a matrix factorization model based on real-world rating data sets. We will elaborate on our approach in Section 3.5.

3.3.2. User Feedback. Immediately after item consumption, a user may submit his feedback in the form of rating for the newly consumed item back to the system. A key benefit of user preferences and item content being represented by identically modelled latent vectors is that it allows us to calculate any user's actual ("ground truth") preference rating for each item by computing the association simply as the dot product of the two vectors. That is, the latent vectors of user u (i.e., $User\ prefs_u$) and item i (i.e., $Item\ content_i$) together determine how any user u rates any item i—that is, $R_{ui} = User\ prefs_u \cdot Item\ content_i$. However, prior literature has found that real users are often inconsistent in their ratings (e.g., Cosley et al. 2003 and Amatriain et al. 2009). When users were asked to rerate previously rated items, despite a strong correlation, their new ratings often did not match earlier ratings (Cosley et al. 2003). Prior research shows that users tend to give ratings from a Gaussian probability distribution (Pennock et al. 2000) and that ratings with a drift of ±1 account for more than 90% of the rate–rerate inconsistencies (Amatriain et al. 2009). Hence, our simulation framework introduces a noise factor to model such rating inconsistency of real-world users. Specifically, users' submitted ratings are perturbed with random noise so that user ratings are normally distributed around their true preferences with standard deviation 1.0 (on the rating scale 1–5). Also, the default setting of our specific simulation in this study is that users rate *every* item that they consume (i.e., $User\ feedback\ likelihood_u = 1.0$). Adding some randomness to how often a user rates consumed items only "stretches" the longitudinal patterns in time; this does not affect the key qualitative characteristics in performance patterns of interest.

3.3.3. User Consumption Frequency. Consumption timespan is the average length of time between a user's two consecutive consumptions and is used to model how often a user consumes items. For example, a value of 2 for a user's consumption timespan means that, *on average*, this user consumes one item every two time periods. The timespan between two consecutive consumptions for user u is drawn from a normal distribution with an average of $User\ cons\ period_u$ and a standard deviation of σ_u. Thus, the time for the next scheduled consumption $User\ next\ cons_u$ is updated as:

$$User\ next\ cons_u = User\ next\ cons_u + \Delta,$$
$$\text{where } \Delta \sim \mathrm{N}(User\ cons\ period_u, \sigma_u).$$

In our experiments, we calibrate the average timespan for each user based on the real-world data and normalize the average timespans to the interval [1, 5]. In other words, the most active users will consume one item in each simulation period, whereas the least active users will consume one item every five simulation periods, on average. The standard deviation is set to be one period.

3.3.4. User Consumption Strategy. With respect to consumption strategy, we focus on users' reliance on recommendation lists when they select an item to consume. In our simulation, we assume that the users are able to scroll through all items (ranked by predicted relevance, from highest to lowest) and select any item. This setting is commonly seen in real-world application settings, such as the "Sort by Relevance" setting on Amazon and "Sorted just for you" on Nordstrom's website.

In the simulation framework, the probability of an item being selected reduces significantly as the item rank gets lower. We use a probabilistic model to simulate the diminishing consumption likelihood based on the item's rank in the recommendation list. Following prior literature that suggests an

exponential pattern of the influence of rank (e.g., Carare 2012 and Ursu 2018), we model the consumption probabilities using an exponential-decay function. The decay rate can be parameterized to simulate how fast the likelihood of an item being consumed decreases as the item's rank gets lower in the recommendation list. For an item ranked at the i-th position, its probability of consumption is calculated as:

$$prob_i = k \cdot \alpha^{-i}.$$

Because the cumulative consumption probability of all items should be 1, the parameter k can be expressed as a function of the exponential decay rate α:

$$\lim_{i\to\infty}\sum_i prob_i = \lim_{i\to\infty} p_0 \cdot \frac{1-\alpha^{-i}}{\alpha-1} = k \cdot \frac{1}{\alpha-1} = 1$$
$$\Rightarrow k = \alpha - 1.$$

To simulate diminishing consumption likelihood, the value of exponential-decay parameter α should be equal to or greater than 1. When $\alpha = 1$, this represents a special case where all items have equal probability of being chosen. In other words, this is equivalent to the situation where a user completely ignores the recommendations and selects items randomly. When $\alpha > 1$, larger values of α represent heavier reliance on the recommendations to make item choices—that is, a user is more likely to choose items that are ranked high in the recommendation list. For example, when $\alpha = 2$, it means that the first item has a 50% probability to be chosen, the second item 25%, the third item 12.5%, and the fourth item 6.25%, and all the other items share the remaining 6.25% probability of being consumed.

In the simulation experiments for our specific study, we model the various aspects of user behavior independently. However, we would like to emphasize that the general simulation framework allows investigators to model relationships among the different aspects with any required degree of complexity, in accordance to a specific research question. For example, a user's consumption strategy and consumption frequency could be modeled as functions of the quality of a system's prior recommendations (e.g., how accurate and diverse the recommendations were in recent user–system interactions). Also, a user's likelihood of submitting her feedback for a newly consumed item may well depend on how much she enjoyed the item.

At the *user–population* level, we vary the distribution of consumption strategy by changing the mixture ratio of different types of users to explore the impact of user–population heterogeneity. Within a population, it is possible for users to adopt different consumption strategies—for example, users can rely on the recommender systems to different degrees. We use percentage distribution to denote the size of different types of users within the population. Section 5 provides more details on heterogeneous (in terms of consumption strategy) user populations that we explored in our study.

3.4. Modeling Recommender System

As discussed in Section 3.1, we model the recommender engine as having three key functionalities: rating prediction, recommendation generation, and performance measurement. For recommendation generation, we model that the system provides item suggestions as a list, which is ranked based on some notion on item relevance (or quality), that users can browse through. The item relevance can be determined in various ways, both personalized and nonpersonalized. In our simulation experiments, we use personalized ranking based on system-predicted ratings for each user, as well as nonpersonalized ranking based on item popularity.

Details of rating prediction and performance measurement are described in the next two subsections.

3.4.1. Rating Prediction. The simulation framework can use any recommendation technique or combination of techniques to estimate users' preference ratings for items. As part of this study, we explored multiple recommendation algorithms, including several variations of the item-based collaborative filtering, user-based CF, and matrix factorization approaches. We found the results to be highly consistent across different algorithms. Because of space constraints, this paper focuses on the item-based CF (Item-CF) algorithm that is widely adopted in real-world applications (e.g., Sarwar et al. 2001 and Smith and Linden 2017).

In the item-based CF approach, the prediction of unknown rating for a target user–item pair is computed as the weighted sum of ratings received by the target item's neighbors (i.e., other items that have similar rating patterns across users) from the target user. More specifically, the predicted rating for user u on item i is an aggregate of user u's ratings on previously consumed items that are similar to item i. The similarity between two items is used as a weight to aggregate ratings. The more similar item j and target item i are, the more weight will be carried by the rating provided on item j by user u in the weighted sum when computing the prediction for target item i. Thus, the predicted rating R^*_{ui} is computed as:

$$R^*_{ui} = b_{ui} + \frac{\sum_{j\in N(u,i)} sim_{ij} \times (R_{uj} - b_{uj})}{\sum_{j\in N(u,i)} |sim_{ij}|},$$

where $N(u,i)$ is the set of nearest neighbors (items) to target item i that were previously rated by user u. The computation of recommendations involves a normalization step to remove "global effects" from rating

data (Bell and Koren 2007). Specifically, a *baseline estimate* for each known rating, denoted as b_{ui}, is computed as $b_{ui} = \mu + b_u + b_i$, where μ is the global mean of ratings in the data set, and b_u (b_i) is the average rating deviation of user u (item i) from global mean μ. We compute similarity between items using the Pearson correlation coefficient. The similarity is further adjusted based on the number of common ratings using a shrinkage parameter, as suggested in Bell and Koren (2007), and based on the inverse item popularity, as suggested in Breese et al. (1998). As part of the parameter tuning and sensitivity analysis, we varied the minimum number of required ratings for similarity calculation from 3 to 20 and the number of nearest neighbors used for rating prediction from 20 to 100. The best results are robust across a range of parameter values. For our simulation, we require that each pair of items must be corated by at least three users to compute a similarity score. Each predicted rating R^*_{ui} is calculated based on user u's ratings on a maximum of 50 most similar items to item i.

3.4.2. Performance Measurement. The performance of recommender systems can be evaluated along many different dimensions. In this work, we focus on three important aspects of recommender systems performance: prediction performance, discovery performance, and outcome performance.

Prediction Performance (i.e., Measured by Predictive Accuracy). Prediction performance measures the ability of a recommender system to correctly estimate users' preferences for items (i.e., how well a system can identify relevant versus irrelevant items for each user). We operationalize prediction performance using the *prediction-accuracy* metric. Prediction accuracy measures how close a recommender system's predicted rating of items for a user differs from the user's true preference. One widely used accuracy metric is root mean squared error (RMSE):

$$RMSE = \sqrt{\sum_{(u,i)\in T} \left(R^*_{u,i} - R_{u,i}\right)^2 / |T|}.$$

Here, $R^*_{u,i}$ represents the system-predicted rating for user u and item i; $R_{u,i}$ is the actual rating, and T is the set of user–item pairs (u,i) used for performance evaluation. In our simulation, we use arguably the most comprehensive set of evaluation ratings, which includes all the available unconsumed elements in the rating space, because we know the ground truth for every possible consumption in our simulation.[2]

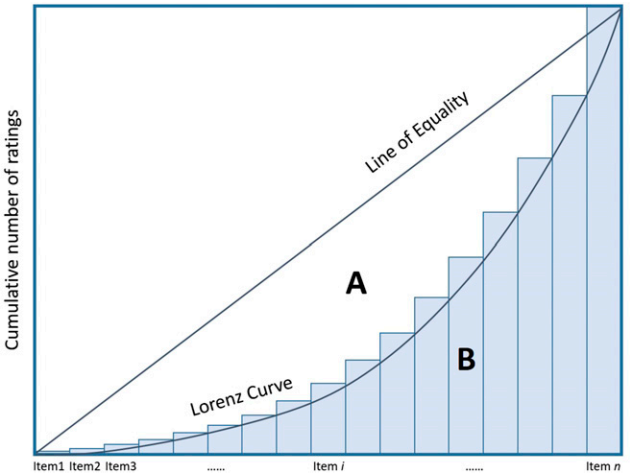
Discovery Performance (i.e., Measured by Consumption Diversity). Discovery performance measures a system's capability to provide users with personalized (perhaps even idiosyncratic) suggestions—in other words, how well the system can narrow down the choice set for the user from the items that are predicted to be relevant to a much smaller list of "good," individualized recommendations that are tailored to match the user's preferences. We measure discovery performance using the *aggregated consumption diversity* of items. Providing the same item (or the same small choice set) to a diverse, realistic population of users can also be viewed as representing a low level of personalization. Therefore, consumption diversity may serve as a proxy of the system's capability in discovering relevant and personalized items.

We use the distributional inequality metric *Gini coefficient* (Gini 1921) to measure aggregate diversity of item-consumption distribution. Gini is computed as the ratio of the area between the line of equality and the Lorenz curve (shown in Figure 1) that represents cumulative frequency of items arranged in the ascending order based on popularity (i.e., area A), over the total area under the line of equality (i.e., area $A + B$). For discrete distributions, Gini is defined as:

$$Gini = \frac{A}{A+B} = 1 - \frac{B}{A+B} = 1 - 2\sum_{i=1}^{n}\left(\frac{n+1-i}{n+1}\right) \times \left(\frac{x_i}{total}\right),$$

where x_i is the popularity of item i (i.e., the number of consumptions of item i), n is the total number of items available in the system, and *total* is the total number of item consumptions. Thus, a value of 0 represents total equality (all items are equally popular among users), and a value of 1 represents maximal inequality (e.g., all users consuming the same exact bestselling item).

Figure 1. (Color online) Lorenz Curve



Outcome Performance (i.e., Measured by the Consumed Item Relevance). Outcome performance focuses on the consumption outcomes resulting from the user–recommender interactions and measures what the user actually ends up *consuming* as a result of his interactions with the system. In this paper, we use the *consumption relevance* metric, the basic idea of which is that an item that is highly rated by the user (after consumption) is considered to be more relevant than an item that is lowly rated. Formally, we measure relevance as the average of user-submitted ratings on newly consumed items:

$$Relevance = \left(\sum_{(u,i)\in C} R_{u,i}\right)/|C|,$$

where $R_{u,i}$ represents the actual rating given by user u on items i, and C is the set of user–item pairs (u,i) used for relevance evaluation—that is, the set of all new item consumptions by users. The relevance metric is designed to provide a complementary, user-centric view of recommender systems' performance that captures quality of users' actual consumption experiences/outcomes rather than quality of displayed recommendations.

Overall, prediction performance, discovery performance, and outcome performance represent three different key dimensions of recommender systems. The first two—that is, prediction and discovery—are about the *capabilities* of the system—that is, measuring the system's capability to provide good *recommendations*. The last one, outcome performance, is about users' consumption experience—that is, serving as a proxy of the user's satisfaction, experience, etc. In the remainder of the paper, for simplicity, we refer to these three performance measurements using their corresponding quantitative metrics. That is, we refer to prediction performance as predictive accuracy, discovery performance as consumption diversity, and outcome performance as consumed item relevance. Although the three metrics used in the study represent different performance aspects, they are not completely orthogonal to each other. For example, prior studies have shown that improvements in diversity may come at a cost of accuracy reduction (e.g., Adomavicius and Kwon 2012).

3.5. Additional Comments on Simulation Initialization and Validation

The proper initialization of the three major simulation framework components—item population, user population, and recommendation engine—represents a crucial component of the simulation procedure (indicated by Step 0 in Algorithm 1). As part of the initialization, we need to seed the simulation with a set of items with realistic item features, a set of users with realistic user preferences, and some initial ratings representing realistic user–item consumptions prior to the starting time $t = 0$. In our experiments, we initialize the item and user populations simultaneously based on real-word data (taken from a real-life recommendation application). Given an actual data set with users U, items I, and containing a set of known ratings R, we first initialize the simulation system to contain the same amount of users and items. To initialize the item content and user preference vectors, we apply a matrix factorization algorithm induced by singular value decomposition (SVD) (e.g., Funk 2006 and Koren et al. 2009) on real-world ratings data R to derive the latent features that best represent users' preferences and items' features. Specifically, the SVD model decomposes the rating matrix containing R into two low-rank submatrices, so that each user u and each item i is represented by a vector of latent factors learned from known rating data—that is, $User\ prefs_u$ and $Item\ content_i$. These vectors are then used throughout the entire simulation and, as discussed earlier, allow us to calculate a realistic preference rating of any user u for any item i (upon item consumption) as a simple dot product: $R_{ui} = User\ prefs_u \cdot Item\ content_i$. More details about the matrix factorization procedure are in Online Appendix D.

Next, we seed the simulation (for the starting time $t = 0$) with $|R|$ preexisting ratings, each of which is readily computed based on the initialized user latent preferences and item latent features, as mentioned above. To ensure that the *distribution* of these initial ratings (i.e., which user–item pairs within $U \times I$ contain known ratings at $t = 0$) is also realistic, we again use the real-world rating data set; specifically, each user's preexisting item consumptions must correspond to the known ratings in the original real-world data set. In summary, we take advantage of a real-world data set to obtain an *initial* simulation environment (users, items, and initial ratings) that is representative of realistic situations.

We make several additional assumptions for this specific study. First, each user consumes and rates an item at most once—that is, no repeated consumption or rating for the same item. Second, all items in the system have identical cost, and, therefore, there are no monetary factors involved when users select items or when the system makes recommendations. Thus, a system's recommendations and a user's selection of an item are based essentially on the item's fit with a user's preferences. Third, items do not have inventory restrictions and can be simultaneously consumed by multiple users. Additionally, there is one central recommender system that serves all users—that is, the recommendations to each user are generated by the same system. These assumptions apply to many real-world settings, especially in marketplaces of digital goods. Example domains include movies,

songs, news, e-books, etc. However, to model the application domains where some of these assumptions are violated, the simulation framework can be configured and extended to incorporate additional factors, such as item price heterogeneity, repeated consumptions, and inventory constraints.

Our simulation is validated and verified in a number of ways. First, the simulation is seeded by real-world data that represent actual user behaviors and item-consumption distributions. This ensures that the model is a reasonable representation of the real-world user and item populations. Second, the latent factors used to model users and items are extracted from real preference data using one of the best-performing matrix factorization algorithms. This assures that the modeled rating process (i.e., deciding what actual rating the simulated user should give to an item upon consumption) is representative of the realistic user's preference formation. Third, our implementations of the recommendation algorithms used in the simulation were validated by comparing their outputs with the outputs of the well-known LensKit recommender package (http://lenskit.org/) on the same set of input data sets and observing highly consistent performance. Lastly, we employed a variety of testing techniques to ensure that the implementation was correct and that it matched the specifications of our conceptual model; also, the reasonableness and robustness of the outputs was examined under a variety of parameter settings.

4. Main Simulation Results: The Performance Paradox

Our primary objective of this study is to understand how the performance of recommender systems evolves with the user population exhibiting different degrees of reliance on recommendations. The main set of experiments focuses on exploring *homogenous* user populations using personalization-based recommender systems (RecSys), where all the users receive the same type of recommendations and rely on these recommendations to the same degree for making their item selections. Following the main results, Section 5 provides additional explorations with *heterogeneous* user populations.

4.1. Application Settings

We performed simulation experiments on multiple user and item populations with homogeneous consumption behavior in order to understand the baseline effects of various pure consumption strategies. Specifically, we simulated two different application settings by seeding our simulation testbed with publicly available data sets, which come from different application domains and have different data characteristics. These data sets include: a subset of the Netflix 100M data set (Bennett and Lanning 2007), from which we randomly sampled 3,000 movies and extracted a random sample of 3,000 users who rated at least one of these movies; and a subset of Yahoo! Music rating data (from http://webscope.sandbox.yahoo.com/), from which we randomly sampled 4,000 songs and extracted a random sample of 6,000 users who rated at least one of these songs. All ratings in the Netflix and Yahoo! Music data sets are integer values between 1 and 5, where 1 represents the least-liked items and 5 represent the most-liked items. The data sets and their main characteristics are summarized in Table 2. Using these data sets, we generated two sets of user and item populations, and the number of ratings represents the initial state (i.e., $t = 0$) of our simulations.

4.2. Key Longitudinal Performance Patterns: The Paradox of Recommender Systems

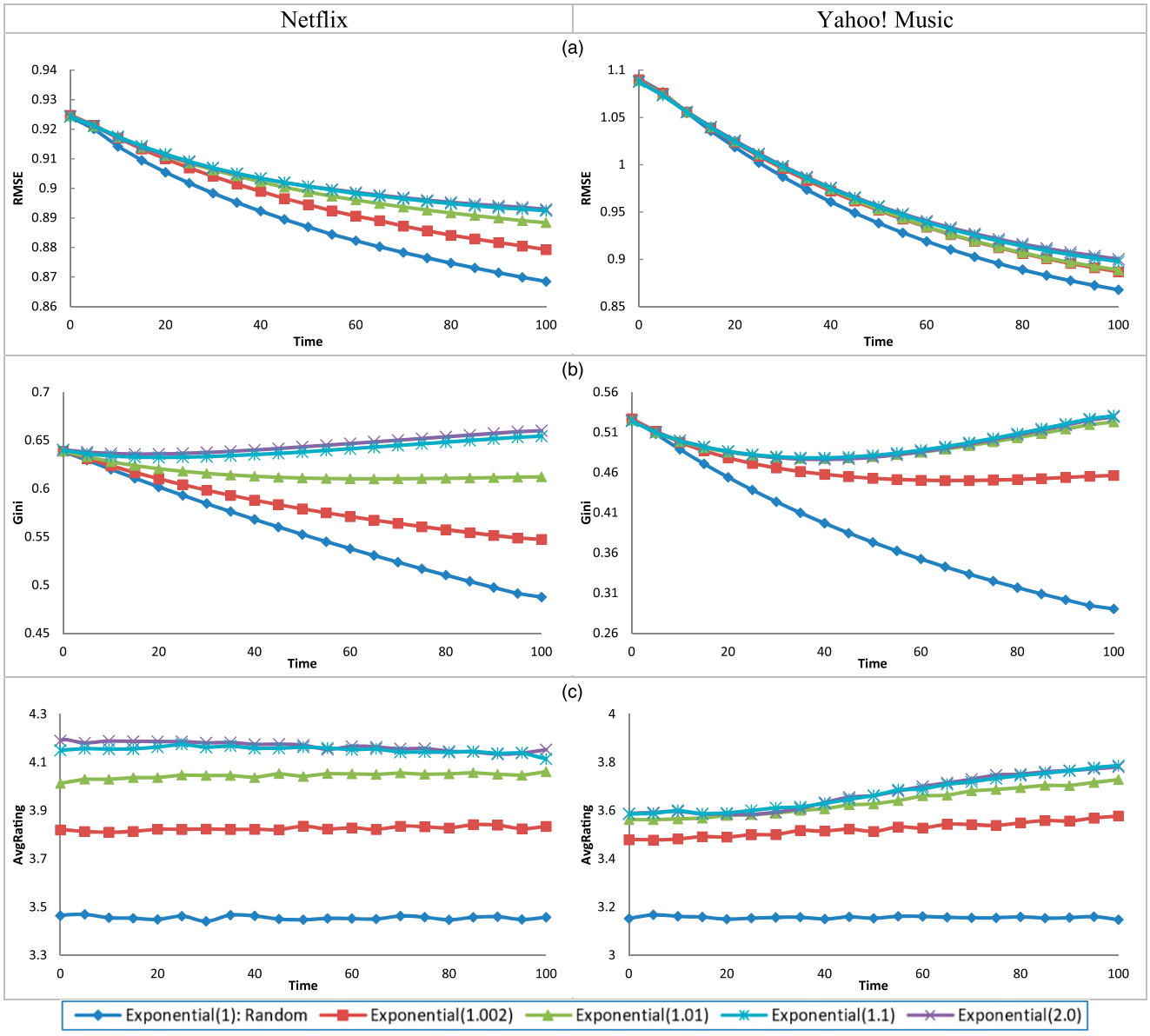
We vary the exponential decay rate parameter from 1 to 2 to simulate the degree of reliance. As mentioned earlier, at one extreme, exponential decay of 1 represents the "Random" consumption strategy, in which the user uniformly at random chooses among the items that he has not consumed before (i.e., completely ignores personalized recommendations). In contrast, exponential decay of 2 represents a consumption strategy that focuses very heavily just on the very top items recommended by the system.

Each user population is endowed with a homogeneous consumption strategy (i.e., every user in the population adopts the same exponential decay parameter), and we compared the impact of consumption strategy on several important recommendation outcomes using the popular item-based collaborative filtering approach. We repeated each simulation five times and report the average results in the paper. Across multiple runs, the seeding of user/item populations and rating distribution is the same, so the variance comes from the randomness in the users' consumption choices, the noise in users' submitted ratings, and timespan between consumptions. We find the longitudinal

Table 2. Summary of Experimental Application Settings

Data set	Description	Users	Items	Ratings	Density
Sample of Netflix	Movie ratings distributed by Netflix	3,000	3,000	344,021	3.82%
Sample of Yahoo! Music	Song ratings released by Yahoo! Music services	6,000	4,000	230,773	0.96%

Figure 2. (Color online) Impact of User Consumption Strategies on Key Recommendation Outcomes Across 100 Time Periods



Notes. (a) Accuracy. (b) Diversity. (c) Relevance.

patterns to be highly robust across runs (variance across five runs is very small). Figure 2 illustrates the temporal changes in aggregate item consumption diversity (measured by Gini coefficient), recommender system's predictive accuracy (measured by RMSE), and relevance of consumed items (measured by average user-specified rating of consumed items) for homogeneous user populations with different consumption strategies, generated from two real-world application settings.

The experimental results suggest that the degree of users' reliance on recommendations significantly affects the longitudinal dynamics of a recommender system. Naturally, the precise magnitude of various performance metrics will vary depending on the specific application domain and/or recommendation algorithm; however, qualitatively, dynamics of recommender systems demonstrate high consistency across application settings. With respect to systems' predictive accuracy, as more ratings get submitted to the system over time, not surprisingly, the RMSE generally keeps decreasing—that is, accuracy is improved. The degree of the accuracy improvement is, however, significantly affected by the consumption behavior of the user population. Somewhat paradoxically, recommender systems achieve the highest accuracy over time when users completely ignore the recommender system and choose to consume items at random. In contrast, as the reliance on personalized recommendations increases (as reflected by higher

exponential decay parameters), the accuracy improvement over time becomes substantially smaller.

With respect to aggregate consumption diversity, we observed largely consistent patterns in the changes of consumption diversity values across different application settings. The more heavily users rely on recommendations, the more concentrated (i.e., less diverse and less personalized) the aggregate consumption of the entire user population becomes over time, as shown by a larger Gini coefficient. Specifically, Gini decreases (i.e., consumption diversity increases) when users either ignore recommendations (i.e., when exponential decay is 1) or use recommendations only to a very small extent—for example, to avoid really "bad" items (e.g., when exponential decay is 1.002). In contrast, when users heavily rely on a system's recommendations to make item selections (e.g., when exponential decay is larger than 1.01), Gini remains relatively unchanged or even increases, meaning that many users end up consuming the same items.

In terms of the recommendation relevance, not relying on personalized recommendations leads to the consumption of lower-relevance items on average, whereas higher reliance on the recommendations leads to more relevant items. Such results are expected and serve as evidence that the recommendation algorithms are effective in finding relevant items for users. Interestingly, the relevance of many consumption strategies remains relatively unchanged longitudinally. The exception is the Yahoo! Music setting, in which we observe increasing recommendation relevance for the consumption strategies with more reliance on the recommender system. The reason is that, unlike Netflix rating data, Yahoo! Music data are much sparser during the initial simulation periods. For neighborhood-based recommendation algorithms, such sparse data are typically not sufficient for accurate predictions, as evidenced by a much higher error rate for Yahoo! Music data at time $t = 0$, as compared with Netflix data (RMSE 1.09 versus. 0.92). In addition, sparse rating data might not allow one to predict *all* unknown ratings—that is, some highly relevant items would remain unidentified. In other words, the *coverage* of neighborhood-based CF is known to suffer in sparse settings (e.g., Ge et al. 2010). With ratings becoming denser over time, both coverage and accuracy improve, leading to more relevant recommendations.

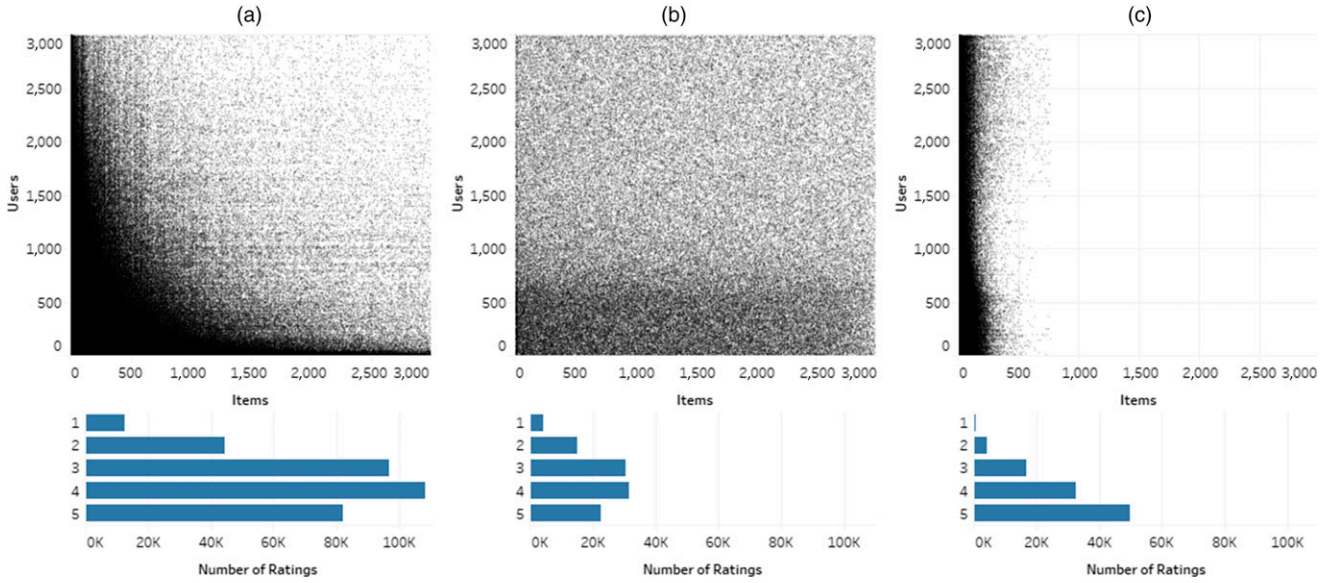
Combining observations based on all three metrics (i.e., diversity, accuracy, and relevance), we observe the *performance paradox* in the longitudinal dynamics of recommender systems. Although random selection strategy leads to the highest system accuracy and item consumption diversity, this improvement in performance is coupled with the lowest relevance of consumed items. Also, as importantly, although system-generated recommendations can effectively help users to find relevant items, the reliance on recommendations will typically result in lower recommendation accuracy improvements and will actually reduce users' aggregate consumption diversity (i.e., a potential indicator of the degree of personalization that the system provides, as discussed earlier) in the long run—both are important issues that may affect users' willingness to continue using recommender systems. In summary, *the observed performance paradox shows that, although higher (as compared with lower) reliance on recommendations leads to higher consumed item relevance over time, the trend is opposite for the other performance metrics; that is, higher (as compared with lower) reliance on recommendations leads to lower future prediction and discovery performance of a recommender system.*

One reason for the performance paradox lies in the changes of rating data distribution over time. Although random consumption provides additional rating data that is more uniformly representative of the underlying data space, the system-based consumption leads to more skewed rating values and consumption patterns. The results are consistent with prior observations that recommendations can push users toward the same items, and thus make users more similar in their consumptions, leading to a poor-get-poorer and rich-get-richer phenomenon (Fleder and Hosanagar 2009). Such a biased data distribution, in turn, impairs the system's capability in learning users' preferences, because it is usually advantageous to learn predictive models from more balanced, less skewed data sets (e.g., Adomavicius and Zhang 2012). The next subsection provides a more in-depth discussion and analysis of the key longitudinal patterns.

4.3. Analysis of the Longitudinal Performance Through Process Metrics

To obtain a more in-depth understanding of the potential underlying mechanisms that drive the different longitudinal patterns in recommender systems performance under different consumption strategies, we analyze the simulation process and all the intermediate results. Specifically, we examine and contrast the changes in various data characteristics and distribution metrics over time for the two most distinct consumption strategies explored in Section 4.2—that is, Exponential(1) and Exponential(2). As discussed earlier, Exponential(1) describes the scenario where users completely ignore the recommender system and consume items at random, whereas under the Exponential(2) scenario, users heavily rely on recommender systems to make consumption choices. For the convenience of reference, we will refer to Exponential(1) as the "*Random*" condition and to Exponential(2) as the "*RecSys*" condition.

Figure 3. (Color online) Rating Distribution Scatterplots and Rating Value Histograms; Netflix Setting



Notes. (a) The initial ratings at time 0. (b) The newly added ratings based on Random consumption strategy over 100 time periods. (c) The newly added ratings based on RecSys consumption strategy over 100 time periods.

Overall, we observe that the two distinct consumption strategies—Random and RecSys—substantially change the fundamental structure and distribution of the underlying rating data over time. Figure 3 illustrates the distribution of ratings and rating values at the initial stage of simulation and the distribution of newly consumed ratings added to the rating data over the 100 simulation periods in the Netflix application setting.[3] Ratings are plotted as a rating matrix, with each column representing an item and each row representing a user. Dark dots represent the positions of known ratings in the matrix. The rows and columns of the rating matrices are sorted according to the rating frequency of users and items—that is, the first column represents the most rated item, the first row represents the user who provided the most ratings, and so on. Figure 3(a) illustrates the initial rating matrix before the simulation process—that is, the rating data at $t = 0$. As would be expected, the initial rating matrix represents a snapshot of a realistic, long-tail-type setting, where some users have consumed a lot and many users have consumed a much smaller number of items, and where some items (e.g., best-sellers) have been consumed quite heavily, but most have been consumed much less. The bottom part of Figure 3(a) is the histogram of rating values in the initial matrix (i.e., the number of ratings equal to 1, 2, 3, 4, and 5). The initial rating data are slightly skewed toward high rating values, with a rating mean of 3.54, as is common with real-world rating data sets.

Starting from the initial rating data, the two strategies—Random and RecSys—result in highly different consumption distributions, illustrated in Figure 3, (b) and (c), respectively. More specifically, this figure depicts the structural distribution (as a scatterplot) and value distribution (as a histogram) of the *newly added ratings* for Random or RecSys consumptions over the entire span of 100 simulation periods. Note that, because users consume at the same pace in both Random and RecSys conditions, the two incremental rating matrices in Figure 3, (b) and (c) contain the same amount of total ratings (i.e., the same number of dark dots). The distributions (both structural and rating value) of these newly added ratings, however, are notably different for the two consumption strategies. Specifically, we see that the RecSys strategy results in a concentrated consumption of a relatively small set of items, as shown by the dense ratings distributed to a small fraction of items in Figure 3(c). In contrast, the Random strategy leads to a more balanced data set, in which ratings are evenly distributed to all the items, as shown in Figure 3(b). Over the entire simulation (100 periods), the RecSys-based consumptions spread across only 770 unique items, whereas the Random-based consumptions spread across the entire set of 3,000 items. In addition, the rating value distributions are quite different between Random- and RecSys-based consumptions. Ratings resulting from the Random strategy follow a normal distribution, with a mean of 3.46, whereas the ratings based on the RecSys strategy are highly skewed toward high values, with a mean of 4.17.

Overall, during the entire simulation period, both Random and RecSys consumption strategies provided the same number of additional ratings that recommendation algorithms can use to further improve their performance; as was shown in Figure 2(a),

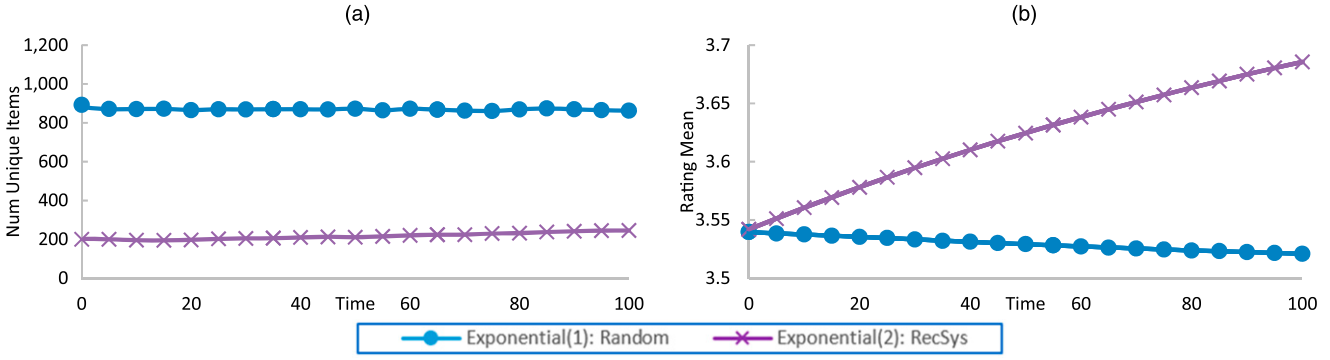
the recommender system's accuracy has increased for *both* Random and RecSys populations. However, as discussed above, while the Random-based consumptions provided additional rating data that are more representative of the underlying data space in terms of structural user and item distribution (e.g., offers new data on the entire pool of items and users), as well as in terms of rating value distribution, the RecSys-based consumptions add new data only for a small fraction of items, as well as with highly skewed rating values. As a result, RecSys consumptions offer very little benefit to help improve predictions on these unrecommended items, and, correspondingly, Figure 2(a) reflects consistently better predictive accuracy performance of Random (over RecSys).

In addition to a general understanding of rating data dynamics in Random and RecSys conditions, we have also calculated a variety of intermediate process metrics (e.g., the rating overlap between pairs of items, the average number of neighbors of each item, and the distribution of ratings for items) for the purpose of exploring the potential mechanisms through which these characteristics may affect outcomes of recommendation algorithms (i.e., item-based collaborative filtering in our case). Prior studies have suggested that data set characteristics significantly impact the accuracy of recommendation algorithms. For example, prior work has examined how connections between users and items (e.g., either purchased or not purchased) affects the accuracy of collaborative filtering recommendation algorithms (Huang and Zeng 2011). Using a bipartite graphical representation of user–item interactions, the authors demonstrated that the topological characteristics of binary data sets could explain why certain CF algorithms work for the given data sets. Their results show that CF algorithms provide higher accuracy on data sets with larger deviations of clustering coefficients. In another study, Adomavicius and Zhang (2012) show that the accuracy performance of CF algorithms is largely driven by the characteristics of rating data. A small number of rating characteristics, including density, rating value distribution, and the frequency distribution of user and item ratings, could explain the majority of the variance in predictive accuracy of CF algorithms.

Following these prior studies, we examine a variety of rating data set characteristics at each simulation period. For example, Figure 4(a) shows that the Random strategy results in consumptions of substantially more unique items than the RecSys strategy. Among the average of 1,031 total consumptions in each simulation period, only about 200 unique items are consumed in the RecSys condition, whereas about 900 unique items are consumed in the Random condition. That is, reliance on recommender systems for item selection leads to an increasingly more concentrated consumption on a small number of items, which in turn leads to more concentrated (i.e., less diverse) data for algorithms to train on. This is highly consistent not only with our observed trends in consumption diversity (Figure 2(a)) and underlying rating data (Figure 3(c)), but also with the observations in prior literature about the concentration effects (or the rich-get-richer effects) of recommender systems that have been reported in research literature (e.g., Adomavicius and Kwon 2012 and Fleder and Hosanagar 2009), and we confirm this with our simulations as well, as discussed below.

Furthermore, when people rely on recommendation lists to make their item selections (as opposed to consuming items at random), they are more likely to consume good items that match well with their personal preferences, and thus their subsequent ratings on these newly consumed items are high. Therefore, over time, we see an increasing difference between rating means of consumed items in the Random versus RecSys

Figure 4. (Color online) Plots of the Number of Unique Items Consumed by Users and the Average Rating Value of the Cumulative Rating Data Set in Each Simulation Period; Netflix Setting



conditions in Figure 4(b), which is also highly consistent with our observed trends in recommendation relevance (Figure 2(c)) and the more granular distributional information about the submitted rating values for these two conditions reported in Figure 3, (b) and (c).

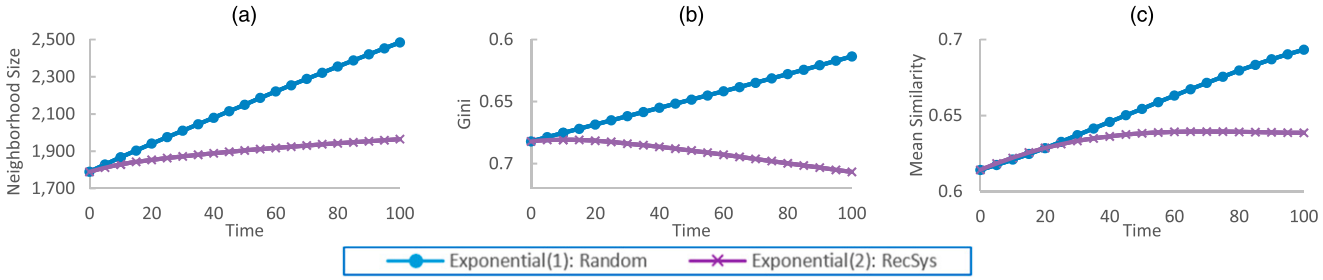
Additionally, because the neighborhood-based collaborative filtering algorithms largely depend on the characteristics of neighborhoods to make predictions, we further examined the temporal trends of the item neighborhoods. For any given item, the size of its neighborhood is defined as the number of other items that have an overlap of at least k ratings with this item (i.e., there are at least k users who rated *both* the focal item and the "neighbor" item). As an informative indicator of temporal dynamics, we computed the average size of item neighborhood. The value of k can be parameterized to adjust the requirements for neighbors: The higher the value of k, the more restrictive it is to be neighbors. We varied the value of k in our explorations and found the patterns to be highly robust. Figure 5(a) illustrates the temporal changes in the neighborhood size for the overlap of $k = 3$, which is what we used in our simulations, in the Random and RecSys conditions.

Figure 5(a) shows that, unsurprisingly, over time (i.e., as more ratings get added to the data), the average neighborhood size in both the Random and RecSys conditions increases. However, this trend is much more prominent in the Random condition as compared with RecSys. The reason for this can be seen from Figure 5(b), which displays the change of distributional inequality in rating overlaps (as measured by Gini coefficient) over time—that is, the RecSys consumption (as compared with Random) leads to a substantially more skewed distribution of rating overlaps among item pairs. In other words, because the RecSys strategy leads to concentrated consumptions on a small fraction of items, the rating overlaps among this small set of items increase significantly. But many of these items, however, are already neighbors (i.e., they have already had an overlap of 3 or more ratings), and, therefore, additional consumptions of these items only increase the overlaps among existing neighbors, but offer little help toward expanding the size of the neighborhood. In contrast, the Random strategy leads to evenly distributed consumptions on all items, and, as a result, we observe an increase in average rating overlap between pairs of items across the board. Because of the increasing overlap among all items, the number of neighbors that share at least k ratings also increase steadily for *all* items. Therefore, we observe a steady increase of the average neighborhood size in the Random condition, as shown in Figure 5(a).

For neighborhood-based algorithms (such as item-based CF) that rely on the nearest neighbors for calculating predicted ratings, a larger neighborhood size enables the algorithms to make more accurate predictions. This is because, with more neighbors, a given item is more likely to have "better," more informative neighbors—that is, items that share more similar rating patterns. This is demonstrated in Figure 5(c), which shows temporal changes of the average similarity score of the 10 nearest-neighbor items used for each rating prediction. As can be seen from the graph, the average similarity score increases significantly more in the Random condition as compared with RecSys, which shows that Random-based consumptions allow the item-based CF algorithm to use more similar items in the rating predictions.

In summary, combining the insights from the different analyses and explorations, our results show that the different consumption strategies—Random and RecSys—change the underlying structure of users' self-reported rating data (spatial and value distributions as well as key structural patterns, such as item overlaps and neighborhoods) over time in

Figure 5. (Color online) Plots of Average Size of Item Neighborhoods, Gini Coefficient of Distribution of Item Rating Overlaps, and Average Similarity of Top 10 Nearest Neighbors to the Focal Item That are Previously Rated by the Focal User in Each Simulation Period; Netflix Setting



Notes. (a) Average number of items with ≥3 common ratings. (b) Gini coefficient of the distribution of rating overlaps. (c) Average similarity of 10 nearest neighbors to the focal item.

significantly different ways. Because these data subsequently serve as input to the recommendation algorithm, they influence the system's performance in the future iterations. We would also like to reiterate that, in all the experiments described in Section 4, the users' RecSys-based consumption strategy was modeled as selecting an item from a scroll-down list ranked based on personalized predicted preference rating, where the probability of item selection decreases exponentially with each list position. There exist several other formats for presenting recommendations to users (e.g., based on a fixed-length top-N list or on a binary classification given a rating threshold) and corresponding ways to model item selection. We explored some of these formats as part of our study and found the longitudinal performance patterns to be highly consistent with the ones discussed in this section. For completeness, we include these additional results in Online Appendix B.

5. Exploring Heterogeneous User Populations and Consumption Patterns

The simulation experiments described in Section 4 represent the first set of steps of investigating recommender system dynamics, where we focused on "canonical" user populations that use simple (i.e., nonhybrid) and homogeneous consumption patterns—that is, every user had the same item-consumption strategy. The agent-based simulation approach allows us to extend our investigation naturally to more complex, more sophisticated consumption strategies, including *hybrid* strategies representing a mixture of multiple canonical strategies. For example, a hybrid approach would allow us to model situations where a user might heavily rely on recommendations at some times but would largely ignore recommendations at other times. In this section, we demonstrate several natural extensions of our analysis by exploring nonhomogeneous user populations and more sophisticated consumption patterns. The first set of experiments examines heterogeneous user populations with different degrees of *reliance* on the personalized recommendations—that is, populations that make some consumptions via zero reliance and some via heavy reliance on the recommender system. The second set of experiments examines the user population mix based on the *source* of recommendations. We explore populations that make some consumptions based on personalized recommendations and some based on popularity-based recommendations.

5.1. User Populations with Heterogeneous Patterns of Reliance on Recommendations

We start by exploring the heterogeneity in users' reliance on personalized recommender systems. It is a natural extension of the main experiments, and it introduces the mixtures of the two extreme reliance strategies explored in the main experiments. We closely follow the design of the main experiment described in Section 4, except for making two adjustments to the user population (resulting in two different simulations):

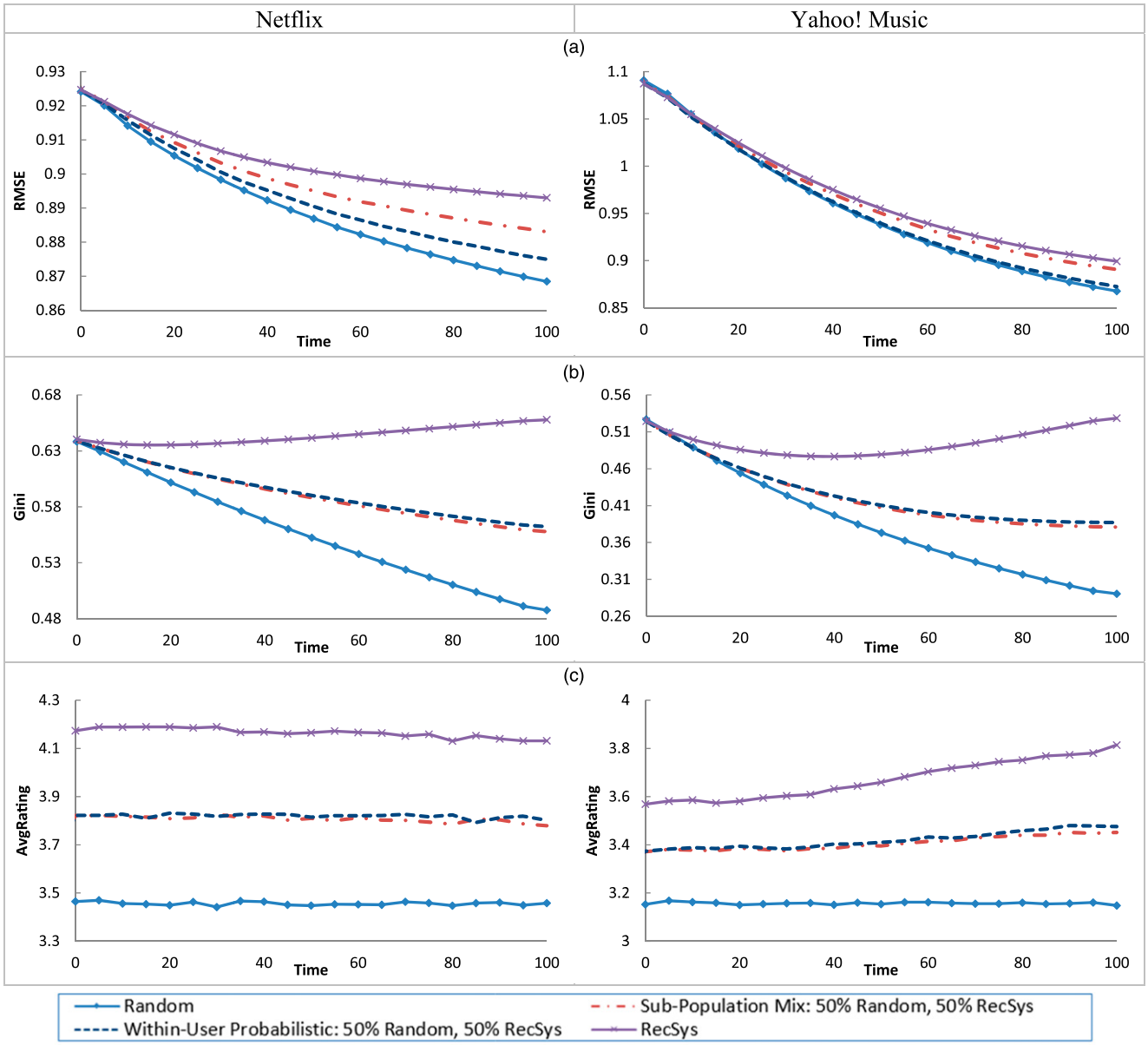
- *Sub-Population Mixture Model* (or cross-user consumption heterogeneity): Instead of all users using the same consumption strategy, we model an equally split user population, where half of the population always chooses items at random—that is, is modeled using Exponential(1)—and the other half always exhibits heavy reliance on top recommended items—that is, is modeled using Exponential(2).
- *Within-User Probabilistic Model* (or within-user consumption heterogeneity): Here, all users have the same (but more sophisticated, i.e., hybrid) consumption strategy; that is, for each consumption, each user chooses an item either at random (with 50% probability) or based on the recommender system (50%).

In other words, these two more complex user populations represent two different approaches (i.e., cross-user and within-user) to combine Random and RecSys canonical consumption strategies. Results for the two more complex user populations based on the Netflix application setting are presented in Figure 6, labeled as "Sub-Population Mix" and "Within-User Probabilistic," respectively. For the ease of comparison, the graphs also include the temporal dynamics of the two homogeneous populations—based on Random (i.e., Exponential(1)) and RecSys (i.e., Exponential(2)) consumption strategies—from earlier experiments.

An interesting observation is that, in terms of diversity and relevance, both nonhomogeneous populations, although defined fairly differently, demonstrate similar performance (to each other) over time. Their longitudinal diversity and relevance performance seems to be approximately in the middle between the performance of the two canonical populations (i.e., Random and RecSys), exactly as might be expected, given the 50%–50% compositions of our nonhomogeneous populations. However, with respect to system accuracy, the within-user probabilistic population consistently exhibits higher accuracy (lower RMSE) than the cross-user subpopulation mixture. For both Netflix and Yahoo! Music settings, the within-user probabilistic populations achieve high longitudinal accuracy, similar to the one of the homogeneous Random population, whereas the performance of cross-user subpopulation mixture is substantially worse, although still outperforming the homogeneous RecSys population.

In summary, exploring even a simple 50%–50% combination of Random and RecSys consumption strategies (either within-user or cross-user) provided novel and interesting insights—that is, that both nonhomogeneous user populations were able to

Figure 6. (Color online) Longitudinal Performance Patterns in Hybrid Consumption Settings



Notes. (a) Accuracy. (b) Diversity. (c) Relevance.

gain about half of the benefits in diversity and relevance, and the within-user probabilistic population was able to gain nearly all the benefits in accuracy, when compared with homogeneous populations. These findings point to important implications for recommender systems design, some of which we discuss in Section 6.

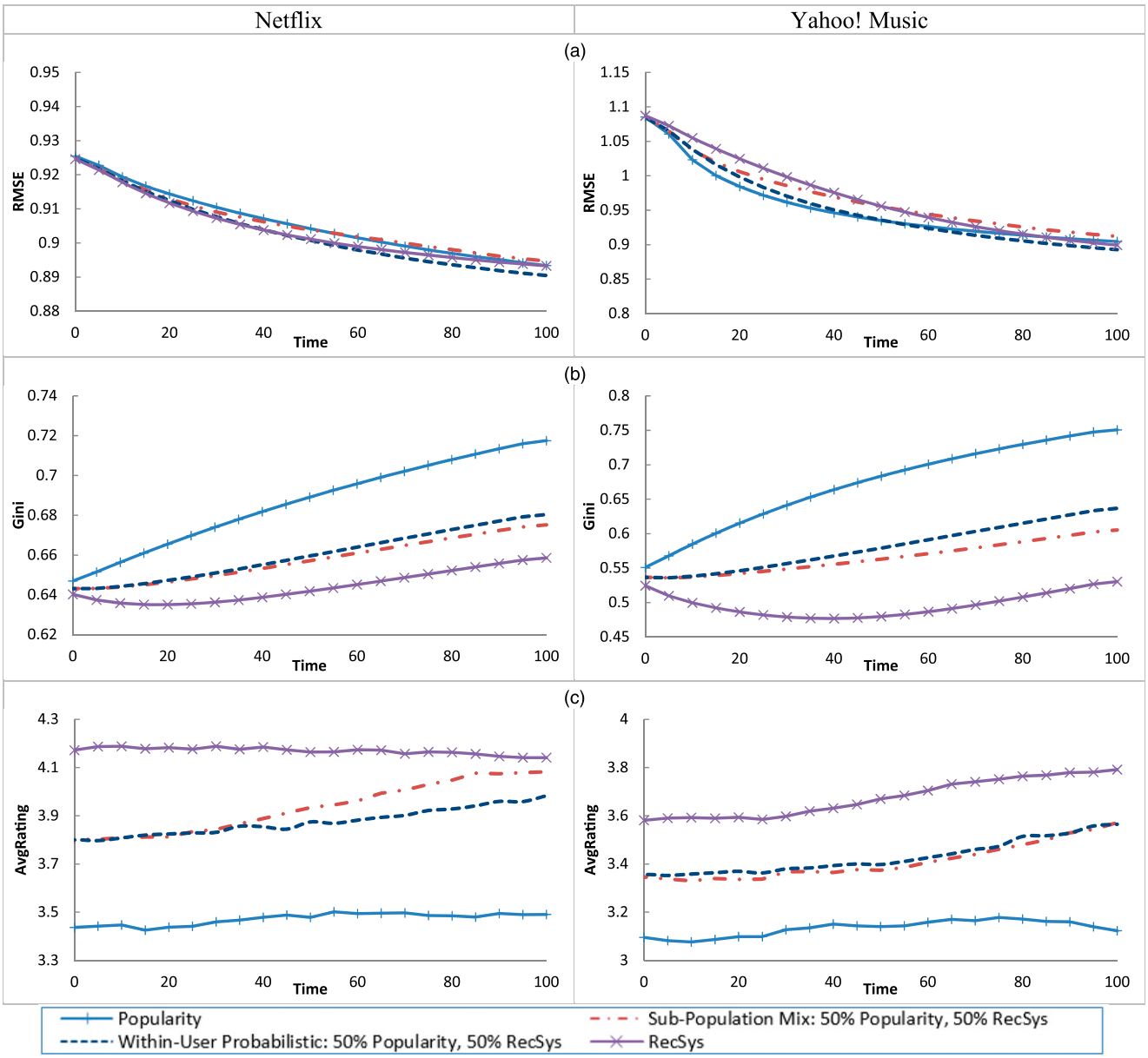
5.2. User Populations Relying on Heterogeneous Sources of Recommendations

In addition to the recommendations generated by personalized recommender systems, real-world users also receive and consider information from many other sources during their decision-making process. One such important source is the *popularity* of items (e.g., from various lists of bestsellers), which represents a key simple type of external information that is easily accessible and can be helpful to users. Popular items tend to receive higher exposure and get selected and consumed more often than less popular items, yielding artificial amplification in the popularity of top-recommended items (e.g., Prawesh and Padmanabhan 2014). Popularity-based recommendations are commonly seen in real-world applications, and, because the popularity rankings are not personalized to individual users, they might lead to different temporal dynamics of recommender systems performance. Therefore, in this experiment, our

objective is to compare the longitudinal dynamics of personalized versus popularity-based systems.

We examine and compare the longitudinal performance of two homogeneous populations with pure consumption strategies based either (i) on the recommendation algorithm (i.e., denoted as RecSys), where the user heavily relies on the personalized recommendation list (ranked by the predicted rating value) to make item choices; or (ii) on popularity (i.e., denoted as "Popularity"), where the user heavily relies on the popularity rank for consumption. Item popularity is measured in a standard manner as the number of users who consumed this item up to this point in time—that is, $Popularity_i$ = number of consumptions of item i. In parallel with prior experiments, we set the exponential decay parameter to be 2 in both RecSys and Popularity conditions to simulate users' reliance on these two systems. Thus, given a ranked item list, either personalized or popularity-based, the users heavily rely on the ranked list to make their choices. In addition, we examine how the hybrid consumption strategies based on *both* personalized and non-personalized (i.e., popularity-based) recommendations affect longitudinal recommender system's dynamics. Similarly to experiments in Section 5.1, we again consider a *heterogeneous user subpopulation mixture* (i.e., denoted as Sub-Population Mix), where half of the users always rely heavily on the popularity

Figure 7. (Color online) Populations with Mixtures of Popularity-Based and Personalized Consumption



Notes. (a) Accuracy. (b) Diversity. (c) Relevance.

rank to choose items, whereas the other half always rely heavily on the personalized recommendations. And we also consider the *within-user probabilistic mixture* (i.e., denoted as Within-User Probabilistic) such that, for each consumption, each user chooses an item either based on the item's popularity rank (with 50% probability) or based on the recommender system (50%). Again, the two more complex user populations represent two different ways (i.e., cross-user and within-user) to combine Popularity and RecSys canonical consumption strategies. Experimental results are presented in Figure 7.

Comparing all four populations, pure popularity-based consumption leads to the lowest diversity (highest Gini) and relevance. When users always consume the most popular items, the aggregate consumption diversity of the user population inevitably decreases substantially (as indicated by the increasing Gini). Comparatively, pure RecSys-based consumption leads to the highest diversity and relevance. In terms of diversity and relevance, consumption strategies that are based on 50%–50% mixtures of RecSys and Popularity tend to exhibit longitudinal performance that is somewhat closer to the performance of pure RecSys-based (as opposed to Popularity-based) consumption. The accuracy performance is comparable across four populations.

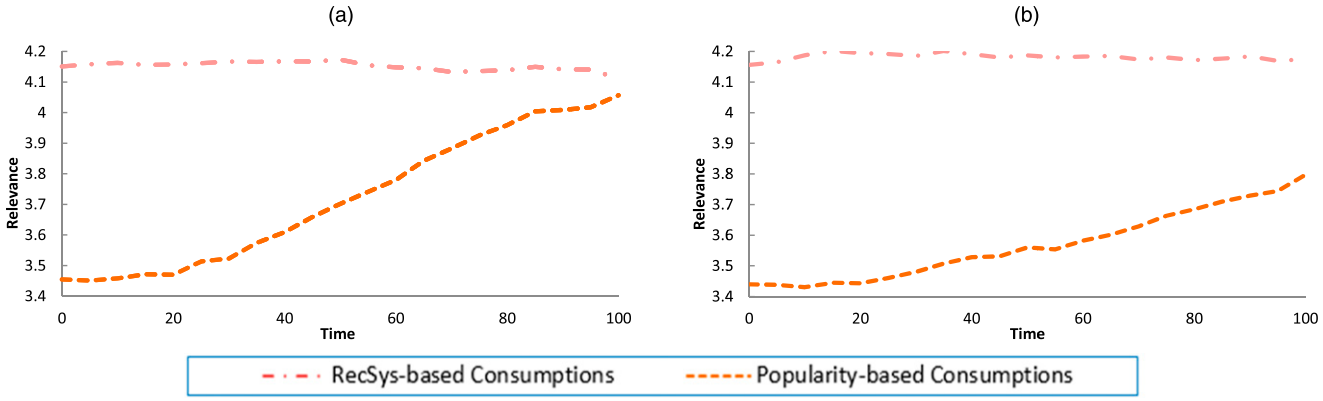
Most interestingly, in the *relevance* graphs, we consistently observe a clear upward trend for the two heterogeneous RecSys–Popularity mixture populations across different application settings. The relevance of items consumed by both subpopulation mixture and within-user probabilistic populations consistently increases over time toward the most advantageous relevance level exhibited by the pure RecSys strategy. In retrospect, such an upward trend was not consistently observed in different application settings (e.g., both Netflix and Yahoo! Music) in other consumption strategies, including the pure RecSys and Popularity-based strategies (*RecSys* and *Popularity* in Figure 7), the two RecSys–Random mixtures (Figure 6), or the Classification and Top-N recommendation strategies (Figure B1 in Online Appendix B).

We conducted further analysis to understand the unique increasing trend of relevance for the heterogeneous populations. Because the two heterogeneous RecSys–Popularity user populations apply a hybrid strategy to select items, we thereby categorized all the item consumptions in both populations into two halves based on how the items were selected by users—that is, one half of the consumptions (denoted "50%-RecSys") is a result from the RecSys strategy, and the second half ("50%-Popularity") is from the Popularity-based strategy. We further calculated the average relevance of consumed items separately for these two halves, and Figure 8 shows how it changed over time in the Netflix setting (results were analogous for the Yahoo! Music setting presented in Online Appendix A).

Interestingly, decomposing the consumption relevance of hybrid RecSys–Popularity user populations (i.e., Sub-Population Mix and Within-Population Probabilistic Mix) into the consumption relevance from the two underlying canonical consumption strategies (50%-RecSys and 50%-Popularity) reveals that the overall upward trend in relevance is driven entirely by the consumptions from the Popularity strategy. More specifically, Figure 8, (a) and (b) shows that, within both heterogeneous RecSys–Popularity user populations, the average relevance of items selected by 50%-RecSys did not change much over time and stayed about 4.15. On the other hand, the average relevance of items selected by 50%-Popularity initially did not exhibit significant change, but later started to increase steadily (after about 20 simulation periods).

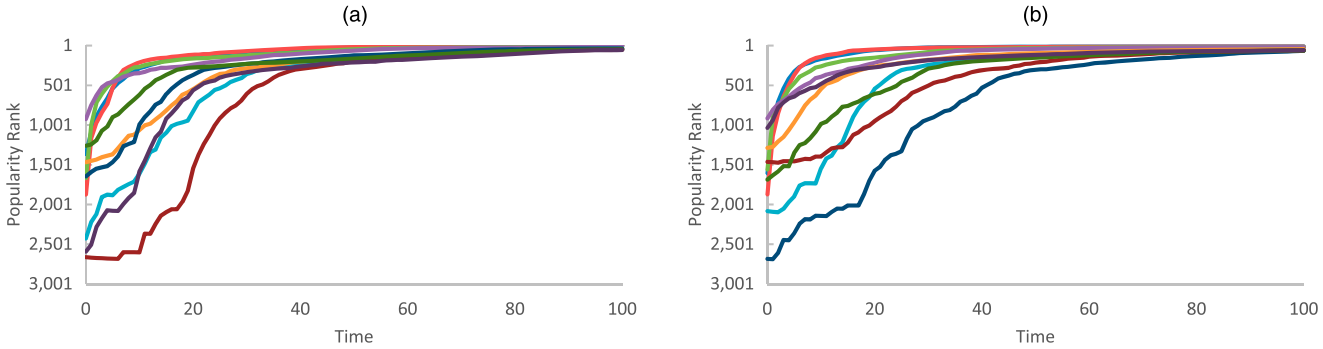
To understand the relevance increase among Popularity-based consumptions within heterogeneous populations, we further analyzed the item popularity rank and

Figure 8. (Color online) Relevance of the Consumed Items Based on the RecSys and Popularity Strategies in the Two RecSys–Popularity Mixture Populations; Netflix Setting



Notes. (a) Subpopulation mix: 50% popularity, 50% RecSys. (b) Within-user probabilistic: 50% popularity, 50% RecSys.

Figure 9. (Color online) Changes of Popularity Rank Positions of the 10 Most-Consumed Items Selected Based on the RecSys Consumption Strategy; Netflix Setting



Notes. Each line represents the popularity rank dynamics for a given item. (a) Subpopulation mix: 50% popularity, 50% RecSys. (b) Within-user probabilistic: 50% popularity, 50% RecSys.

consumption patterns in each simulation period. We found that, over time, the popularity rank of items was materially altered by the RecSys-based consumptions in a way that is advantageous for Popularity-based consumptions: (i) Following RecSys strategy (as part of the mixture) leads to concentrated consumptions of good-quality items, some of which are not (yet) popular; (ii) this leads to more ratings on these good items and improves their popularity rank; (iii) eventually some of these items climb up and make their way near the top of the popularity chart; and (iv) following Popularity consumption strategy (as part of the mixture) takes advantage of these newly popularized items that are of high quality, leading to an upward trend of consumption relevance of Popularity-based consumptions.

Figure 9 illustrates this by plotting the changes of the popularity rank position for the 10 *most-consumed* items selected by the RecSys strategy. At initial period $t = 0$, all of these items are ranked low (e.g., below 925 out of 3,000 total items in the Netflix setting)—that is, the good items (i.e., items that are highly predicted to many users) that are not popular. Over time, these items are highly recommended to users by the recommender system and, thus, get consumed by many as part of their RecSys consumptions; hence, the popularity of these items increases across the overall data set. Eventually, several of these items make it all the way up to the list of the most popular items, which then get consumed even more as part of the Popularity consumptions. In other words, hybrid RecSys–Popularity strategies exhibit highly symbiotic tendencies (which do not manifest for the homogeneous Popularity strategy alone), where the personalized recommendation system helps discover and popularize items that are likable and relevant to many users; this helps increase the average quality of the items at the top of the popularity charts, and therefore subsequently increases the relevance of the Popularity-based consumptions. In other words, the personalized recommendation algorithms facilitate a general quality-rises-to-the-top phenomenon, which is another important finding that the agent-based simulation approach enabled to uncover.

We also note that the degree of increase in relevance of Popularity-based consumptions is associated with the degree of change in the top popularity rank list; for example, by comparing Figure 9, (a) and (b) for the Netflix application setting, we see that the Sub-Population Mix has more dramatic changes in the top-10 popular item list than the Within-User Probabilistic mixture, which leads to the different upward slopes of relevance over time, as observed in Figure 8.

It is clear that both the composition of user population and the nuances of within-user consumption patterns substantially impact a recommender system's long-term dynamics. The analysis of simple mixtures of personalized–random or personalized–popularity consumption strategies already uncover a number of interesting patterns of the longitudinal dynamics, which further highlights the usefulness and importance of the agent-based simulation approach. An interesting future research direction would be to systematically explore the impact of more complex populations and consumption strategies on recommender systems' outcomes and performance, including the impact of more sophisticated mixtures of users' reliance on different recommendation approaches as well as additional sources of nonpersonalized recommendations (e.g., item likability, critics' reviews, and word-of-mouth).

6. Discussion and Conclusions

6.1. Summary of Findings

Understanding temporal dynamics of recommender systems represents an important yet underexplored research problem. In this research, we developed a general-purpose agent-based simulation and modeling

approach to analyze how user–recommender interactions affect recommender systems in the long run. The proposed simulation framework can be used to systematically investigate the impact of various population configurations and item consumption patterns (as well as other characteristics of recommendation applications that may be difficult to control for in real-world settings and system deployments) on the longitudinal recommender system's performance.

Our explorations with the agent-based simulation framework provide a number of insights that have potentially significant implications for both the designers and the users of recommender systems. In particular, we identify an interesting (and perhaps somewhat paradoxical) phenomenon that has not been explored in research literature: *Even though such systems are designed to help users find relevant items, the more users use the system to make their choices, generally the less improved the system becomes in the future*. More specifically, users' reliance on the recommender system, although helping users discover relevant items, actually hurts the diversity of the items that are recommended and consumed as well as impairs the system's learning pace to improve predictive accuracy in the future. In contrast, in the long run, the system's accuracy and diversity would improve the most under the hypothetical scenario where users entirely ignore a system's recommendations and randomly select items to consume. Such a performance paradox is consistently observed across several application settings. We further present a thorough analysis to explain why and how this performance paradox occurs. We show that users' higher reliance on recommender systems leads to a less advantageous structural distribution (i.e., more concentrated on a small number of items) and value distribution (i.e., more skewed toward high values) of the rating data, which subsequently serve as inputs to the recommender systems and, thus, impair the system's capability to provide accurate and personalized recommendations in the long run. The in-depth analysis (enabled by the simulation approach) of the underlying process leading to the performance paradox represents another contribution of this research.

Additional explorations of longitudinal dynamics of recommender systems under a variety of simulation settings (e.g., personalized versus popularity-based recommendations, nonhomogeneous user populations, top-N, and classification systems) consistently suggest the existence of a recommender system's performance paradox, but with nuanced temporal patterns. For example, a certain *hybrid* consumption strategy—that is, where users rely on both personalized- and popularity-based recommendations—offers a unique combination in which recommender systems facilitate the general quality-rises-to-the-top phenomenon by discovering the good-quality items (with potential mass appeal, but not yet popular) and popularizing them over time.

6.2. Implications and Future Research Directions

Our findings have brought to light several potentially significant issues in the design and implementation of recommender systems. As the main general implication, it is necessary to have a more holistic view about recommender systems performance. Although accuracy, diversity, and relevance are all useful performance measurements, they represent very different dimensions of recommendation quality. Based on their specific application context, the system's design needs to weigh the relative importance of these three dimensions on the desired overall system utility. Even more importantly, it is not sufficient to develop recommendation algorithms by optimizing them for recommendation accuracy based on a snapshot of historical rating data—that is, using the traditional predictive modeling process—as it may not lead toward a favorable performance trajectory, even when users simply keep following the system's recommendations (as would be the intention). In other words, a longitudinal awareness of a system's performance is important, given the findings of this study.

In terms of more specific implications, because users' consumption strategies can significantly influence the longitudinal performance of recommender systems, it is important for system designers to explicitly account for this. Conducting in-depth analysis of the system's recommendations and users' consumption history to infer users' consumption strategies would allow the system to anticipate a user's item choices and strategically adjust the system's parameters according to its long-term objectives. In other words, various recommender system's components (e.g., rating-prediction algorithms, interface designs, or rating mechanisms) that play a role in users' interactions with the system may need to be reconsidered—for example, to account for the performance paradox. If users' reliance on recommendations can impair the system's future performance, then how to balance the short-term usefulness of the popular and widely used standard ranking approach (i.e., recommending highest predicted items to users) with its implications on the longitudinal system's performance? Reinforcement learning methodologies, as mentioned in Section 2.1, provide promising opportunities for researchers and practitioners to develop recommendation approaches that best assist users while taking into account the user consumption strategies and long-term performance dynamics of the system. Also, some beneficial consumption strategies

could be proactively promoted through a system's design. The quality-rises-to-the-top phenomenon discovered in this work demonstrates the advantages of hybrid (i.e., personalized and popularity-based) consumption and, thus, provides rich opportunities for developing and testing various integrative popularity-based and personalized recommendation designs.

This work opens up a multitude of additional interesting directions for further research. Although this paper represents initial explorations using the proposed agent-based simulation framework to study the collective behaviors of users on recommender systems, future research is needed to expand the research scope to explore more factors that might affect the dynamics of such systems. For example, our current experiments always initialized consumption distribution based on real rating data sets to simulate real-word behaviors. Future work can examine how alternative initialization states (e.g., varying skewness of item popularity or distributions of initial user consumptions—more uniformly distributed versus more concentrated) affect recommender systems' dynamics. Another potential extension is modeling new item introduction over time in the simulation and examining how different factors (e.g., arrival rates or different strategies of dealing with "cold-start" problem) affect the dynamics of the system.

Another direction for future research is to investigate the effect of heterogeneity of user populations in a more granular manner. In real-world systems, users' consumption strategies are exogenous, and the exact nature of effects of the consumption strategies largely depends on the population heterogeneity. Findings on simple 50%–50% mixtures of consumption patterns explored in this paper already provide some important insights and implications, which could lead to interesting research questions about more sophisticated population and consumption mixtures. For example, future research is needed to investigate the various degrees of population heterogeneity (e.g., any proportion of x% versus $(100 - x)$% mixtures of two consumption strategies) and the exact nature of their longitudinal effects on recommender system's performance. Future work should also investigate more sophisticated population mixtures with multiple levels of reliance on personalized and nonpersonalized recommendations.

Although this paper simulates some noise in users' submitted ratings to provide more realism, the noise is added as a random perturbation that is normally distributed around the user's true preference. However, prior research literature shows that user populations often exhibit *systematic* biases when providing online reviews (Moe and Schweidel 2012, Luca and Zervas 2016) and ratings (Cosley et al. 2003, Adomavicius et al. 2013). Thus, using the proposed framework, one could simulate systematic biases in users' submitted preference ratings due to various user–system interactions. For example, biases can be introduced by the exposure to system's predicted ratings, other users' reviews and ratings, and item popularity information. Understanding the longitudinal effects of systematic biases in users' ratings on the temporal dynamics of recommender systems performance constitutes another promising avenue for future work.

In summary, longitudinal dynamics of recommender systems represents a problem-rich area, and the experiments in this paper represent just one set of explorations using the agent-based simulation framework. These experimental findings would arguably be more difficult to uncover using other methodologies, such as field studies and laboratory experiments, because inherent user behavior (e.g., consumption strategies and system usage) is extremely hard to control. This further emphasizes the value of agent-based simulation as a systematic, flexible, low-cost, and low-risk methodology for performing comprehensive explorations as well as in-depth analyses of recommender systems dynamics. Insights from simulation experiments can lead to new theoretical or algorithmic developments as well as more targeted field and laboratory experiments. Therefore, this paper advocates for embracing simulation technology in recommender systems research.

Endnotes

[1] As discussed earlier, in this study, we used a fixed set of users that were available for the entire simulation period—that is, the aspects of user arrival and user lifespan were not considered.

[2] In terms of robustness checks, we have evaluated the accuracy performance based on several other sets of rating data: (i) a random hold-out sample of ratings extracted from real-world rating data; and (ii) the subset of highly predicted ratings of the unknown rating space. We found that the accuracy performance evaluated on these rating samples still demonstrates the same overall longitudinal patterns, consistently with the accuracy on the overall set of unknown ratings.

[3] Because of space limitations, we present the results from the Netflix application setting in this paper to discuss the underlying mechanisms of the performance paradox of recommender systems. The results from the Yahoo! Music settings are provided in Online Appendix A and are consistent with those from the Netflix setting.

References

Adomavicius G, Kwon Y (2012) Improving aggregate recommendation diversity using ranking-based techniques. *IEEE Trans. Knowledge Data Engrg.* 24(5):896–911.

Adomavicius G, Tuzhilin A (2005) Toward the next generation of recommendation system: A survey of the state-of-the-art and possible extensions. *IEEE Trans. Knowledge Data Engrg.* 17(6): 734–749.

Adomavicius G, Zhang J (2012) Impact of data characteristics on recommender systems performance. *ACM Trans. Management Inform. Systems* 3(1):Article 3.

Adomavicius G, Bockstedt J, Curley S, Zhang J (2013) Do recommender systems manipulate consumer preferences? A study of anchoring effects. *Inform. Systems Res.* 24(4):956–975.

Adomavicius G, Bockstedt J, Curley S, Zhang J (2018) Effects of online recommendations on consumers' willingness to pay. *Inform. Systems Res.* 29(1):84–102.

Amatriain X, Basilico J (2012) Netflix recommendations: Beyond the 5 stars: Part 1. *The Netflix Tech Blog*. Accessed March 8, 2018, https://medium.com/netflix-techblog/netflix-recommendations-beyond-the-5-stars-part-1-55838468f429.

Amatriain X, Pujol JM, Oliver N (2009) I like it … I like it not: Evaluating user ratings noise in recommender systems. Houben G-J, McCalla G, Pianesi F, Zancanaro M, eds. *Conf. User Modeling Adaptation Personalization* (Springer, Berlin), 247–258.

Ansari A, Essegaier S, Kohli R (2000) Internet recommendation systems. *J. Marketing Res.* 37(3):363–375.

Bapna R, Goes P, Gupta A (2003) Replicating online Yankee auctions to analyze auctioneers' and bidders' strategies. *Inform. Systems Res.* 14(3):244–268.

Bapna R, Goes P, Gupta A, Karuga G (2008) Predicting bidders' willingness to pay in online multiunit ascending auctions: Analytical and empirical insights. *INFORMS J. Comput.* 20(3):345–355.

Bell RM, Koren Y (2007) Improved neighborhood-based collaborative filtering. *Proc. KDD Cup Workshop 2007* (ACM, New York), 7–14.

Bennett J, Lanning S (2007) The Netflix prize. *Proc. KDD Cup Workshop 2007* (ACM, New York), 51–52.

Breese JS, Heckerman D, Kadie C (1998) Empirical analysis of predictive algorithms for collaborative filtering. Cooper GF, Moral S, eds. *Proc. 14th Conf. Uncertainty Artificial Intelligence* (Morgan Kaufmann Publishers, San Francisco), 43–52.

Brynjolfsson E, Hu Y, Simester D (2011) Goodbye Pareto principle, hello long tail: The effect of search costs on the concentration of product sales. *Management Sci.* 57(8):1373–1386.

Carare O (2012) The impact of bestseller rank on demand: Evidence from the app market. *Internat. Econom. Rev.* 53(3):717–742.

Chaturvedi AR, Dolk DR, Drnevich PL (2011) Design principles for virtual worlds. *Management Inform. Systems Quart.* 35(3):673–684.

Cosley D, Lam S, Albert I, Konstan JA, Riedl J (2003) Is seeing believing? How recommender interfaces affect users' opinions. *CHI 2003 Conf.* (ACM, New York), 585–592.

De P, Hu Y, Rahman MS (2010) Technology usage and online sales: An empirical study. *Management Sci.* 56(11):1930–1945.

Fleder D, Hosanagar K (2009) Blockbuster culture's next rise or fall: The impact of recommender systems on sales diversity. *Management Sci.* 55(5):697–712.

Funk S (2006) Netflix update: Try this at home. Accessed March 3, 2016, http://sifter.org/~simon/journal/20061211.html.

Ge M, Delgado-Battenfeld C, Jannach D (2010) Beyond accuracy: Evaluating recommender systems by coverage and serendipity. *4th ACM Conf. Recommender Systems* (ACM, New York), 257–260.

Gini C (1921) Measurement of inequality and incomes. *Econom. J.* 31(121):124–126.

Herlocker JL, Konstan JA, Terveen K, Riedl JT (2004) Evaluating collaborative filtering recommender systems. *ACM Trans. Inform. Systems* 22(1):5–53.

Huang Z, Zeng DD (2011) Why does collaborative filtering work? Transaction-based recommendation model validation and selection by analyzing bipartite random graphs. *INFORMS J. Comput.* 23(1):138–152.

Jannach D, Lerche L, Kamehkhosh I, Jugovac M (2015) What recommenders recommend: An analysis of recommendation biases and possible countermeasures. *User Modeling User-Adapted Interaction* 25(5):427–491.

Ketter W, Peters M, Collins J, Gupta A (2016a) Competitive benchmarking: An IS research approach to address wicked problems with big data and analytics. *Management Inform. Systems Quart.* 40(4):1057–1080.

Ketter W, Peters M, Collins J, Gupta A (2016b) A multiagent competitive gaming platform to address societal challenges. *Management Inform. Systems Quart.* 40(2):447–460.

Ketter W, Collins J, Gini M, Gupta A, Schrater P (2012) Real-time tactical and strategic sales management for intelligent agents guided by economic regimes. *Inform. Systems Res.* 23(4):1263–1283.

Konstan J, Miller B, Maltz D, Herlocker J, Gordon L, Riedl J (1997) Grouplens: Applying collaborative filtering to Usenet news. *Commun. ACM* 40(3):77–87.

Koren Y, Bell R, Volinsky C (2009) Matrix factorization techniques for recommender systems. *IEEE Comput. Soc.* 42(8):30–37.

Li L, Chu W, Langford J, Schapire RE (2010) A contextual-bandit approach to personalized news article recommendation. *Internat. WWW Conf.* (ACM, New York), 661–670.

Linden G, Smith B, York J (2003) Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Comput. Soc.* 7(1): 76–80.

Luca M, Zervas G (2016) Fake it till you make it: Reputation, competition, and Yelp review fraud. *Management Sci.* 62(12): 3412–3427.

Marshall M (2006) Aggregate knowledge raises $5M from Kleiner, on a roll. Accessed April 29, 2012, http://venturebeat.com/2006/12/10/aggregate-knowledge-raises-5m-from-kleiner-on-a-roll/.

Miller JH, Page SE (2007) *Complex Adaptive Systems: An Introduction to Computational Models of Social Life* (Princeton University Press, Princeton, NJ).

Moe WW, Schweidel DA (2012) Online product opinions: Incidence, evaluation, and evolution. *Marketing Sci.* 31(3):372–386.

Pennock DM, Horvitz E, Lawrence S, Giles CL (2000) Collaborative filtering by personality diagnosis: A hybrid memory and model-based approach. Boutilier C, Goldszmidt M, eds. *Conf. Uncertainty Artificial Intelligence* (Morgan Kaufmann Publishers, San Francisco), 473–480.

Prawesh S, Padmanabhan B (2014) The "Most Popular News" recommender: Count amplification and manipulation resistance. *Inform. Systems Res.* 25(3):569–589.

Pu P, Chen L, Hu R (2011) A user-centric evaluation framework for recommender systems. *Proc. 5th ACM Conf. Recommender Systems* (ACM, New York), 157–164.

Ren Y, Kraut R (2014) Agent-based modeling to inform online community design: Impact of topical breadth, message volume, and discussion moderation on member commitment and contribution. *Human Comput. Interaction* 29(4):351–389.

Resnick P, Iacovou N, Suchak M, Bergstrom P, Riedl J (1994) Grouplens: An open architecture for collaborative filtering of netnews. *Proc. 1994 Conf. Comput. Supported Cooperative Work* (ACM, New York), 175–186.

Sarwar B, Karypis G, Konstan JA, Riedl J (2001) Item-based collaborative filtering recommendation algorithms. *Proc. 10th Internat. Conf. World Wide Web* (ACM, New York), 285–295.

Smith B, Linden G (2017) Two decades of recommender systems at Amazon.com. *IEEE Internet Comput.* 21(3):12–18.

Sutton RS, Barto AG (1998) *Reinforcement Learning: An Introduction* (MIT Press, Cambridge, MA).

Ursu R (2018) The power of rankings: Quantifying the effect of rankings on online consumer search and purchase decisions. *Marketing Sci.* 37(4):530–552.

Wooldridge M, Jennings NR (1995) Intelligent agents: Theory and practice. *Knowledge Engrg. Rev.* 10(2):115–152.

Zeng C, Wang Q, Mokhtari S, Li T (2016) Online context-aware recommendation with time varying multi-armed bandit. *Proc. 22nd ACM SIGKDD Internat. Conf. Knowledge Discovery Data Mining* (ACM, New York), 2025–2034.