Modeling Recommendation Systems as Two-Sided Markets

Arthur Nghiem

May 2024

1 Introduction

A recommendation system is an algorithm which provides users with suggestions as to which items are most relevant to them. These systems have become ubiquitous in online services. Companies such as Amazon and Spotify use recommendation systems to increase customer engagement. By gathering data directly from the user or by observing their behavior, such systems learn about the user's preferences for future use. In this work, we concern ourselves primarily with recommendation systems that pair users with creators. Such systems can be described as two-sided markets. The streaming service Twitch is one example. The user chooses between different channels to watch, possibly influenced by the provided recommendations, each of which is hosted by a specific creator.

Intuition would suggest that the recommendation system should simply match each user with the creators who most closely align with their exact preferences. We refer to this approach as "user-centric". User-centric recommendation systems may suffer due to their failure to address creators' incentives [5]. Just as users are agents whose satisfaction depends on the quality of their recommendations, so too are creators agents whose satisfaction instead depends on the quantity of engagement they receive. Taking this fact into account, it could be beneficial to use a recommendation algorithm which sometimes provides "sub-optimal" recommendations from the user's perspective for the sake of keeping creators content. The remainder of this work explores this idea, and the conditions under which it may apply.

2 Original Model

This paper is greatly indebted to Huttenlocher et al., whose work "Matching of Users and Creators in Two-Sided Markets with Departures" [3] was the basis and inspiration for this work. Hereafter, we refer to this paper as HLLOS. Its authors describe recommendation systems as two-sided markets in which both users and creators have the agency to leave the platform if they so choose. In particular, their model has the following characteristics:

- Each user is assigned a fixed number of recommendations, denoted as K. We assume that the user will accept all of these suggestions. Such a pairing is called a *match*.

- Each user and creator on the platform has a *type*, which is a multi-dimensional unit vector which defines the kind of content they prefer or produce respectively. User i's type and creator j's type are denoted as u_i and c_j respectively.

- The engagement achieved from a specific user-creator pairing is defined as the *similarity* between the user's type and the creator's type. For an arbitrary pairing between user i and creator j, this is defined as $e_{ij} = u_i^T c_j$.

- All creators have a specific threshold $\bar{a}$ on the number of users they are matched with. If a creator's engagement falls below this threshold, they leave the platform. Otherwise, they stay on the platform.

- All users have a specific threshold $\bar{e}$ for the quality of their recommendations. Each of the user's recommendations must achieve this level of engagement at minimum. The user leaves if any of their recommendations fail to do so and stays otherwise.

- $\bar{e}$ is same across all users and $\bar{a}$ is the same across all creators.

- There are no new users or creators arriving after the platform initially launches.

Given these conditions, the authors prove that some algorithms will outperform the user-centric algorithm in terms of long-term engagement. While this is a novel result, it must be stated that some aspects of this model appear rather unrealistic. For example, the user will not decide which creators to interact with solely based on the recommendations they are given. In some cases, they will make such decisions independently. The model also assumes homogeneity of users and creators in some aspects where it certainly does not apply. For example, more active users will have higher capacity for recommendations, more popular creators will have higher expectations for their level of engagement, etc.

However, the most pressing issue is the discontinuity of the agents' decision making models. Both the users and the creators have hard thresholds, above which they stay on the platform with 100% probability and below which they stay on the platform with 0% probability. It would be far more accurate to characterize their behaviors more smoothly, in which probability of staying on the platform is positively related to quantity of engagement or quality of recommendations for the creator and the user respectively. We refer to this setting as the "continuous setting". HLLOS proposes two algorithms that perform better than the user-centric algorithm in their model. We now determine whether such algorithms still exist under a more realistic model of user and creator behavior.

3 Simulations

We investigate the efficacy of the user-centric algorithm versus possible alternatives through Python simulations. Each simulation represents the lifespan of a platform which uses a recommendation algorithm. The simulations implement HLLOS's model of recommendation systems as two-sided markets while relaxing its most unrealistic assumption.

The platform begins at $t = 0$ with a specified number of users and creators (U_{init} and C_{init} respectively). Each user and creator is initialized with a randomly generated type. These types are D-dimensional unit vectors which fully express the user or creator's preferences.

From here, a recommendation algorithm is used to assign users to creators. Each user has capacity K, meaning that they can be assigned to K different creators at any given time. We retain the assumption that users will rely on the recommendation algorithm, engaging with creators if and only if they have been recommended to them. Four different recommendation algorithms are tested in this simulation (user-centric, local clustering, creator ranking, and filtered greedy), each of which is later described in detail. In any case, the recommendation algorithm generates a $U_{init} \times C_{init}$ matrix R, where $R_{ij} = 1$ if creator j is recommended to user i and $R_{ij} = 0$ otherwise.

After the matching at any time t, each user and creator now individually decides whether or not to remain on the platform considering the outcomes of the recommendation algorithm. The user makes this decision based on how closely their recommendations fit with their type. Higher quality recommendations correspond with a higher probability of continuing to use the platform. In particular, given global parameters x_{user} and k_{user}, the user uses the following logistic function to determine this probability:

$$p_u(x) = \frac{1}{1 + e^{-k_{user}(x - x_{user})}}$$

x represents the quality of the user's recommendations. For user i, this is computed as:

$$x_i = \sum_{j=1}^{C_{init}} e_{ij} R_{ij}$$

x_{user} is the threshold at which the user is equally likely to stay or depart, and k_{user} characterizes the steepness of the probability vs. quality curve. Each creator likewise uses a logistic function to determine their probability of staying on the platform. Whereas users are concerned with the quality of recommendations, creators instead decide based on the quantity of recommendations they receive.

$$p_c(y) = \frac{1}{1 + e^{-k_{creator}(y - y_{creator})}}$$

y represents the number of users the creator is matched with. This computed for creator j as:

$$y_j = \sum_{i=1}^{U_{init}} R_{ij}$$

$y_{creator}$ is the threshold at which the creator is equally likely to stay or depart, and $k_{creator}$ characterizes the steepness of the probability vs. quantity curve.

After each user and creator makes a decision, the platform reaches $t = 1$ with a subset of its users and creators from the previous iteration. The process is then repeated - the recommendation algorithm is applied with the remaining users and creators, then users and creators observe the recommendation outcomes and may depart depending on the results.

At each iteration, the total engagement across all users is recorded. These logs are used to derive the performance metrics which allow for comparison between different combinations of parameters and algorithms. The simulation continues for a maximum of T iterations, or once the platform has zero users and creators, whichever is earlier. This entire process may be replicated N times to obtain more reliable results.

3.1 Algorithms

The simplest algorithm that can be used is the **user-centric algorithm**. This algorithm recommends each user to the K creators closest to their preferences. The user-centric algorithm will always be the best choice for satisfying each individual user's preferences within this model, although its disregard for the creators' incentives means that it does not necessarily maximize long-term engagement.

The first algorithm proposed by HLLOS is the **local clustering algorithm**. This algorithm works by picking a random user and determining the number of creators and users sufficiently close in type. If these numbers exceed K and $x_{creator}$ respectively, then within the "neighbourhood" all unassigned users are assigned to K creators. This is repeated until is is no longer possible to satisfy more users. Creator j is said to be in user i's neighborhood if and only if $u_i^T c_j \geq \bar{e}$. In other words, this means that creator j can be assigned to user i without immediately causing user i to leave the platform.

The second algorithm they propose is the **creator ranking algorithm**. This algorithm works by determining the potential audience size of each creator. A user is part of a creator's potential audience if the user has capacity for more recommendations and the creator is in the user's neighborhood. Creators whose potential audience size cannot possibly reach their quantity threshold are effectively removed from the system. From there, the creator with the lowest potential audience size is assigned the appropriate viewers. This is repeated until all creators are either satisfied or deliberately ignored.

Both of these algorithms provide performance guarantees over the user-centric algorithm in the discontinuous setting described by HLLOS. However, translating these algorithms into the continuous setting renders them less useful. The original versions of both of these algorithms do not provide all users with K recommendations. This is ill-advised in the continuous setting, so they have been modified to fill in the remaining recommendation slots by referring to the user-centric model when needed.

This work proposes the **filtered greedy algorithm** specifically for use in the continuous case. It begins by matching each user with their K closest creators as defined by cosine similarity $e_{ij} = u_i^T c_j$. The algorithm then follows by evaluating which creators are likely to leave the platform given these recommendations. If a creator's probability of retention is less than a specified threshold, they are removed from all recommendations. This work uses threshold $\epsilon = 0.1$. The pairing of users with nearest neighbors is now redone without such creators. The removal of creators could continue in an iterative fashion, but this is not explored in this work. The filtered greedy algorithm can be seen as an effort to combine desirable features of both the user-centric and creator ranking algorithms.

3.2 Metrics

These simulations require clear and easy to interpret indicators of how the platform performed given a certain set of parameters and choice of recommendation algorithm. The program can display the total engagement E_t generated at any given time $t = 1, 2, ..., T$, but we must seek to summarize these findings to facilitate comparison between different scenarios. The following metrics could be used for this purpose:

- HLLOS uses **long-term engagement** as their metric. This is the value of total user engagement as the number of time iterations becomes arbitrarily large. Unfortunately, it is not logical to use this measure in the continuous case. Theory dictates that, since all agents have a non-zero probability of leaving at any given iteration, this metric will be 0 in all cases. This is in contrast with the HLLOS's discontinuous model, in which the platform can reach a "stable set" of users and creators who will never leave the platform.

$$\lim_{t \to \infty} E_t$$

- **Final engagement** can be used as an approximate stand-in for long-term engagement. This is defined as the level of engagement at $t = T$, where it is assumed that T is sufficiently large such that engagement is no longer rapidly decreasing.

$$E_T$$

- **Survival time** is the number of iterations that pass before the platform has zero engagement. One may recall that the simulation only runs up to T iterations before terminating to limit computation time. If the platform has non-zero engagement at this point, then the survival time is declared to be T.

$$\max(t|E_t > 0)$$

- **Total engagement** is the sum of engagement over all time iterations. This metric is one of the more appealing options since it summarizes performance over the entire lifespan with the platform. Total revenue from the platform would be most closely correlated with this metric.

$$\Sigma_{t=1}^{T} E_t$$

- **Turnover rate** is a metric requiring a slight modification to the code structure. The model is changed such that creators who leave the platform are immediately replaced by another creator with the same type in the next iteration. The proportion of creators who leave in each iteration is recorded and then averaged.

3.3 Procedure

The groundwork has now been laid to discuss the experimental procedure. The goal is to determine the conditions under which other algorithms can outperform the user-centric algorithm, if any. We find that U_{init} and C_{init} (the initial number of users and creators respectively) are of particular interest. All combinations of $U_{init} \in \{100, 110, 120, ..., 300\}$ and $C_{init} \in \{15, 16, 17, ..., 25\}$ are tested. The other parameters take on the same value for all simulations as given below:

- Dimensions of the type vectors: $D = 5$
- User capacity: $K = 6$
- Maximum number of time iterations per simulation: $T = 20$
- User's threshold for the quality of recommendations: $x_{user} = 1.5$
- Steepness of the user's quality vs. retention curve: $k_{user} = 6.5$
- Creator's threshold for the quantity of recommendations: $x_{user} = 60$
- Steepness of the creator's quantity vs. retention curve: $k_{user} = 0.2$

All $|U_{init}| \times |C_{init}| = 231$ parameter sets are simulated $N = 100$ times using the user-centric algorithm. Average total engagement is computed for each parameter set. This process is repeated for the local clustering algorithm, the creator ranking algorithm, and the filtered greedy algorithm. We can compare the performance of two of these algorithms by plotting the ratio of average total engagement between the two for each parameter set.

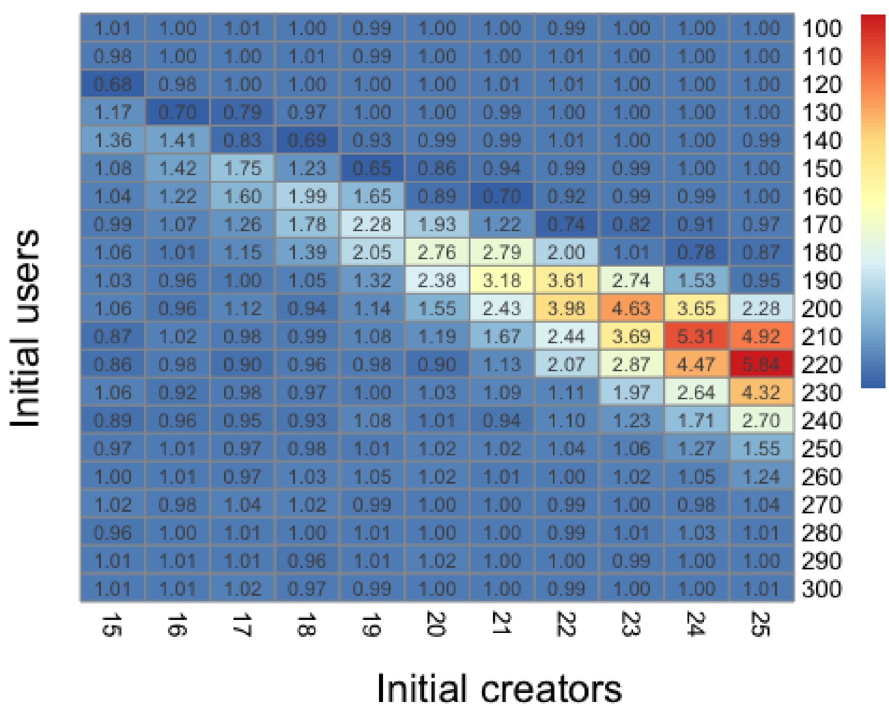
To better compare the user-centric algorithm with the filtered greedy algorithm, we run additional simulations using different metrics. Every parameter set is simulated an additional 100 times using the user-centric algorithm to compute average survival time for each parameter set. The filtered greedy algorithm is tested in the same way. This entire process is repeated to compute average final engagement and average turnover rate at each parameter set for both algorithms.

4 Results

4.1 User-Centric Algorithm vs. Filtered Greedy Algorithm

Average total engagement (rounded to the nearest whole number) for each every combination of initial users and initial creators while using the user-centric algorithm is displayed in Table 1 of the Appendix. Likewise, average total engagement for each every combination of initial users and initial creators while using the filtered greedy algorithm is displayed in Table 2. To visualize the difference in performance between the two algorithms, we plot a heat map of the ratio of total engagement using the filtered greedy algorithm to total engagement using the user-centric model.

Figure 3: Ratio of average total engagement, filtered greedy over user-centric



We make the following novel observations. For most ordered pairs (U_{init}, C_{init}), the ratio approximately equals 1. This implies roughly equal performance between the two algorithms. However, when $0 \leq 10C_{init} - U_{init} < \kappa$ for small κ relative to U_{init}, we observe vastly superior performance by the filtered greedy algorithm compared to the user-centric algorithm. This evidence supports HLLOS's claim that the user-centric algorithm is not always the best choice for maximizing overall engagement on the platform.

These results have a convincing theoretical explanation. The quantity $U_{init}K$ represents the collective capacity of the users, whereas the quantity $C_{init}x_{creator}$ represents the collective expectations of the creators. If $U_{init}K > C_{init}x_{creator}$, then there are more than enough users to keep all creators satisfied. The filtered greedy algorithm will function essentially the same as the user-centric algorithm, as there will be no need to "prune" creators from the system. Inversely, if $U_{init}K < C_{init}x_{creator}$, it is mathematically impossible to meet the expectations of all creators on the platform. It may then be advantageous to exclude some creators from receiving recommendations so as to reserve them for other creators who could be convinced to stay on the platform with the additional engagement. However, if $U_{init}K \ll C_{init}x_{creator}$, then the platform will not be sustainable regardless of the recommendation algorithm used, and there is therefore little difference in performance between the two.

Considering the values $K = 6$ and $x_{creator} = 60$ used in this experiment, one can easily correspond the theoretical inequalities to the empirically observed results. In general, the filtered greedy algorithm is superior if the system is constrained by creator departures, whereas the user-centric algorithm is equal or slightly advantageous if the system is constrained by user departures. Another fact which illustrates this point is that the filtered greedy algorithm disproportionately suffers from an increase in user threshold relative to the user-centric algorithm.

We may also compare the user-centric algorithm with the filtered greedy algorithm using the other performance metrics. Refer to Table 4 through Figure 12 in the appendix for data pertaining to survival time, final engagement, and turnover rate respectively. These results are largely consistent with the the preceding discussion.

4.2 Local Clustering and Creator Ranking Algorithms

We now investigate whether or not the two algorithms presented in HLLOS can be advantageous compared to the user-centric algorithm in the continuous case. Average total engagement data for the local clustering algorithm is presented in Table 13 of the Appendix, and data for the creator ranking algorithm is presented in Table 14. We generate two heat maps with this data. The first compares the local clustering algorithm with the user-centric algorithm, and the second compares the creator ranking algorithm to the user-centric algorithm.

Figure 15: Ratio of average total engagement, local clustering over user-centric

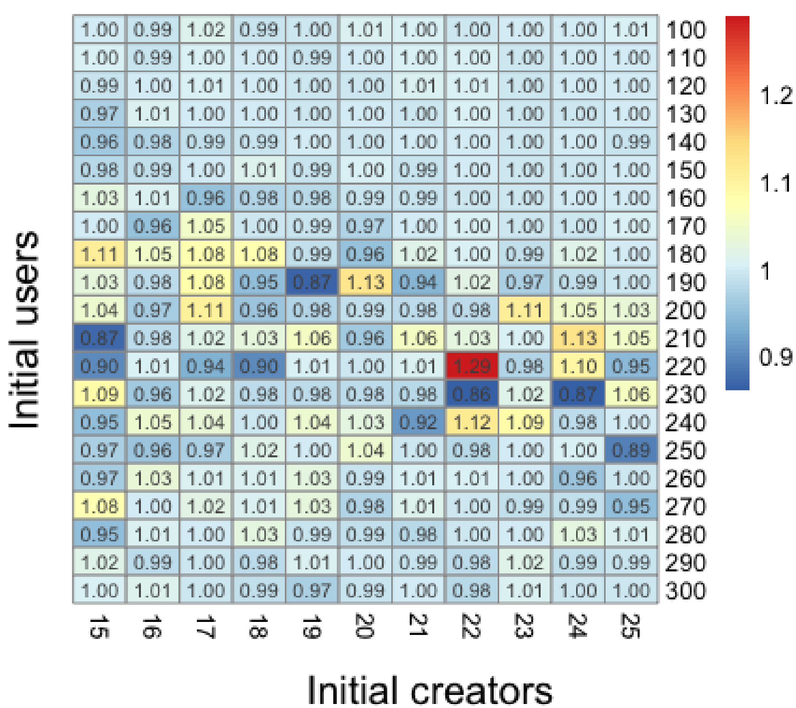
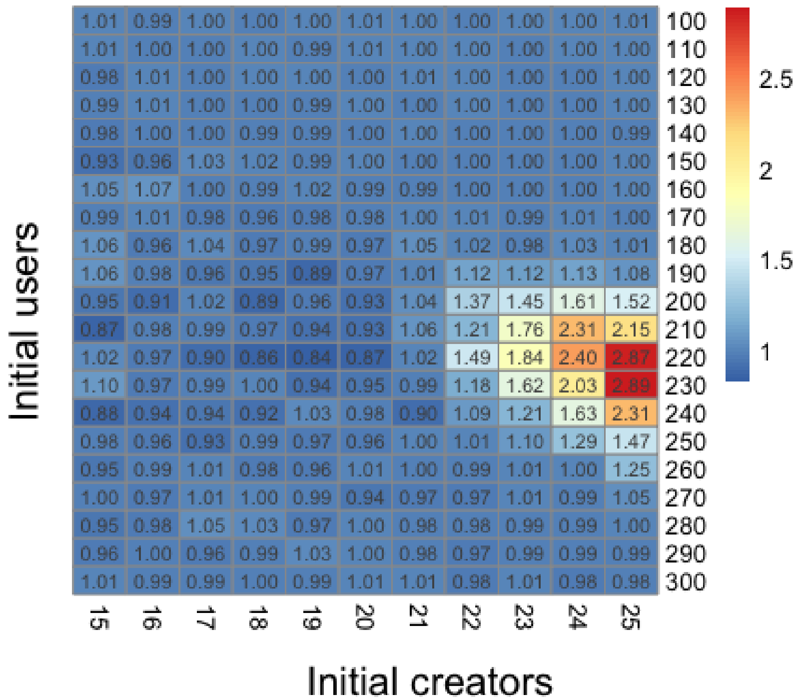


Figure 16: Ratio of average total engagement, creator ranking over user-centric



The local clustering algorithm does not seem to perform significantly better than the user-centric algorithm. In cases where the number of users is insufficient for the number of creators, for example when $(U_{init}, C_{init}) = (130, 23)$, the two algorithms provide nearly identical performance. There are deviations between the two in other cases, but these seem to be a result of statistical noise rather than a systemic difference between the two algorithms.

The creator ranking algorithm achieves greater average total engagement than the user-centric algorithm in some cases. In particular, it is most advantageous when the number of initial creators is higher and when the number of users is slightly insufficient to satisfy these creators, for example when $(U_{init}, C_{init}) = (230, 25)$. The creator ranking algorithm is similar to the filtered greedy algorithm in this regard. However, the filtered greedy algorithm is generally more successful, especially when the initial number of creators is relatively low.

5 Discussion

The work draws the same overarching conclusion as HLLOS. In some cases, the user-centric model fails to maximize engagement over the long-run. However, this work uses a more empirical and simulation-based approach compared to HLLOS, which uses mathematical proofs to guarantee the superiority of certain algorithms over the user-centric algorithm. Furthermore, this work draws a broader conclusion than HLLOS by considering the continuous setting. In the continuous setting, changes to the user-centric model cannot be made without increasing the probability of user departures. Despite this, we find that such changes may still be a net benefit to the system as a whole. Contrast this with HLLOS, whose work in the discontinuous setting allows for some recommendation reassignments to be made without impacting user retention. Their proposed local clustering and creator ranking algorithms take advantage of this fact.

5.1 Limitations

This work does not perfectly simulate how recommendation systems function as two-sided markets. Many of the assumptions made in the original model have been retained. For example, users and creators still have homogeneous decision-making models, and users always follow the suggestions of the recommendation algorithm. The model may be robust to violations of these particular assumptions, and verifying this by modifying the simulations could be an avenue of future research. However, there are more foundational limitations which must be addressed.

Our model assumes that user and creator types are independently and randomly distributed. Though this assumption is convenient computationally, it would not hold for a realistic recommendation system. One would expect to observe groups of users and creators with similar types. In fact, recommendation systems are often designed to identify and leverage such clusters [1]. Our simulations also imply that user and creator types do not change over time. This assumption is also unfounded. Users' taste evolve over time, and creators are incentivized to change their content to meet popular demand. This dynamic is left unaccounted for in our simulations.

Furthermore, our model assumes that users are merely concerned with the accuracy of their recommendations. A user's probability of staying on the platform only depends on the similarity between their type and the recommended creators. Although high accuracy is generally desirable, it is not the only quality that influences the user's satisfaction. They also value serendipity, meaning that they may appreciate less obvious recommendations even if there are alternatives which are strictly speaking "more accurate" [7].

5.2 Application

How would these results apply in practice? This depends on the state of the platform in question. If the primary concern is maintaining the user base (that is to say $U_{init}K > C_{init}x_{creator}$), then the platform should continue operating with a typical user-centric recommendation system. However, if departure of creators is more pressing, then the recommendations may benefit from taking the creators' agency into account just as the filtered greedy algorithm does. The challenge would then be to identify which creators would be most sensitive to an increase in recommendations. In this model, every creator has an identical and well-characterized decision making model. This is not the case in reality, so observations must be made to tease out this information from creators just as conventional recommendation systems learn the preferences of individual users.

Note that the filtered greedy algorithm increases long-term engagement by limiting recommendations to certain creators. This may be accomplished by restricting creators whose content poses social costs. For example, a recommendation system could decrease recommendations for articles containing vaccine misinformation [2], which would have the additional benefit of "freeing up eyeballs" for less problematic content. In other cases, the exclusion of certain creators from receiving recommendations may prove ethically challenging. Doing so could introduce unfairness, defined as "harmful disparity in experiences with a system" [6], into the recommendation system. Creators deemed likely to leave the platform could receive reduced engagement through no direct fault of their own.

Though the aforementioned technical limitations and ethical concerns may limit the applicability of the proposed filtered greedy algorithm, this work certainly illustrates the importance of the balance between supply and demand in a recommendation system. The filtered greedy algorithm would be potential solution if demand is insufficient compared to supply. If supply is insufficient instead, a different solution should be implemented such as the one proposed by Jobson [4].

6 Appendix

Table 1: Average total engagement under the user-centric algorithm.

	C_{init}										
U_{init}	15	16	17	18	19	20	21	22	23	24	25
100	88	92	93	96	97	97	98	99	99	99	99
110	98	101	103	105	107	107	108	108	109	109	109
120	108	110	113	114	116	117	117	118	119	119	119
130	121	120	123	124	126	127	128	128	129	129	129
140	151	136	133	135	136	137	137	138	139	139	140
150	229	183	151	145	146	146	148	148	149	149	149
160	312	257	205	175	162	159	159	158	159	159	159
170	454	407	316	255	203	181	170	168	169	169	169
180	579	592	477	380	291	233	197	186	183	178	179
190	759	838	777	687	538	339	292	222	207	195	191
200	1008	1149	1045	1045	848	648	452	309	246	219	207
210	1382	1321	1420	1479	1268	1092	785	602	401	277	259
220	1639	1712	2008	2137	1971	1863	1428	898	694	477	363
230	1758	2193	2304	2482	2581	2522	2208	2004	1289	1008	650
240	2435	2639	2912	3080	3025	3156	3316	2778	2388	1779	1204
250	2683	3066	3435	3503	3664	3744	3757	3704	3440	2832	2386
260	3180	3360	3756	3933	4009	4165	4271	4331	4265	4084	3319
270	3427	3845	4026	4259	4438	4679	4675	4727	4775	4738	4493
280	3881	4139	4368	4594	4855	4945	5028	5088	5128	4989	4978
290	4147	4471	4794	4999	4917	5144	5312	5346	5325	5462	5405
300	4366	4700	4982	5234	5399	5468	5503	5649	5598	5657	5688

Table 2: Average total engagement under the filtered greedy algorithm.

	C_{init}										
U_{init}	15	16	17	18	19	20	21	22	23	24	25
100	89	92	94	96	96	97	98	98	99	99	99
110	96	101	103	106	106	107	108	109	109	109	109
120	73	108	113	114	116	117	118	119	119	119	119
130	142	84	97	120	126	127	127	128	129	129	129
140	205	192	111	93	126	135	136	139	139	139	139
150	247	259	265	178	95	125	139	147	148	149	149
160	325	314	329	348	268	141	111	146	157	158	159
170	451	435	397	454	462	350	207	124	138	153	164
180	614	600	548	528	598	644	549	372	184	138	155
190	784	807	775	718	711	806	929	802	567	299	181
200	1067	1103	1166	986	964	1003	1100	1230	1138	799	471
210	1209	1350	1391	1468	1373	1302	1314	1467	1480	1471	1274
220	1414	1676	1804	2057	1929	1680	1614	1856	1992	2134	2119
230	1858	2028	2253	2409	2584	2592	2397	2219	2535	2657	2805
240	2172	2531	2770	2861	3274	3181	3112	3054	2936	3042	3246
250	2597	3097	3321	3449	3700	3812	3840	3835	3638	3596	3688
260	3172	3398	3645	4054	4217	4267	4304	4338	4331	4293	4126
270	3503	3758	4199	4344	4392	4657	4670	4691	4758	4655	4678
280	3722	4144	4396	4603	4893	4939	5003	5030	5165	5149	5043
290	4169	4516	4819	4780	4959	5271	5310	5327	5277	5457	5403
300	4409	4737	5061	5084	5361	5450	5528	5587	5618	5667	5717

Table 4: Average survival time under the user-centric algorithm.

						C_{init}					
U_{init}	15	16	17	18	19	20	21	22	23	24	25
100	1	1	1	1	1	1	1	1	1	1	1
110	1	1	1	1	1	1	1	1	1	1	1
120	1.04	1	1	1	1	1	1	1	1	1	1
130	1.22	1	1	1	1	1	1	1	1	1	1
140	1.79	1.3	1.06	1	1	1	1	1	1	1	1
150	2.71	2.08	1.32	1.12	1.01	1	1	1	1	1	1
160	3.48	3.14	2.32	1.73	1.26	1.08	1.02	1	1	1	1
170	4.51	4.16	3.22	2.73	2.03	1.46	1.1	1.06	1	1	1
180	5.48	5.52	4.8	3.79	2.81	2.34	1.53	1.32	1.06	1.04	1.02
190	6.54	6.88	7.02	5.55	4.11	3.24	2.39	1.78	1.4	1.15	1.08
200	7.52	8.03	8.01	7.98	6.53	5.46	4.39	2.93	2.37	1.48	1.21
210	8.84	9.65	10.2	10.03	9.52	7.78	6.63	4.77	3.53	3.06	1.82
220	10.09	11.57	12.41	12.63	13.27	11.63	10.01	7.04	6.13	4.42	3.36
230	13.01	14.11	14.49	14.73	15.96	15.45	14.75	11.73	10.39	7.25	5.06
240	14.88	15.01	16.28	17.27	17.3	17.54	17.62	16.14	14.35	11.68	8.34
250	16.01	17.13	17.78	18.31	18.86	19.16	19.11	18.8	18.5	16.7	14.32
260	17.83	17.78	19.1	18.95	19.55	19.62	19.7	19.77	19.46	19.11	17.99
270	17.91	18.81	19.35	19.65	19.83	19.93	19.94	19.91	20	19.99	19.82
280	18.72	19.79	19.97	19.82	19.94	19.97	20	19.96	20	20	20
290	19.64	19.95	19.91	19.98	19.98	19.96	20	20	20	20	20
300	19.78	19.95	19.97	20	20	20	19.95	20	20	20	20

Table 5: Average survival time under the filtered greedy algorithm.

						C_{init}					
U_{init}	15	16	17	18	19	20	21	22	23	24	25
100	1	1	1	1	1	1	1	1	1	1	1
110	1.01	1	1	1	1	1	1	1	1	1	1
120	1.57	1.02	1.01	1	1	1	1	1	1	1	1
130	2.26	1.9	1.3	1.06	1	1	1	1	1	1	1
140	2.86	2.9	2.14	1.51	1.09	1.02	1.01	1	1	1	1
150	3.11	3.54	3.49	2.51	1.98	1.41	1.02	1	1	1	1
160	3.83	3.75	3.99	3.99	3.36	2.4	1.72	1.19	1.05	1.02	1
170	4.6	4.45	4.27	4.6	5.05	4.59	3.04	2.19	1.47	1.17	1.05
180	5.83	5.89	5.26	5.4	5.19	6.13	5.51	3.79	2.88	1.88	1.4
190	6.94	6.64	6.96	6.28	6.26	6.62	7.16	6.94	5.43	3.68	2.61
200	8.06	8.04	8.68	8.24	8.17	8.18	8.29	8.77	9.05	7.14	4.84
210	9.83	9.77	10.73	10.89	9.66	9.97	9.55	10.47	11.7	11.45	9.38
220	11.01	11.26	12.25	13	13.05	12.17	11.91	11.8	13.2	13.35	13.59
230	11.88	13.37	14.48	15.8	16.17	15.73	15.4	14.43	14.72	15.44	16.4
240	14.66	16.3	15.49	16.73	17.73	17.65	17.39	17.15	16.51	17.87	17.27
250	16.12	17.71	18.16	18.02	18.72	18.75	19.28	19.02	18.75	18.18	17.7
260	16.72	18.29	18.82	19.32	19.34	19.79	19.78	19.65	19.67	19.63	19.57
270	18.51	18.98	19.64	19.68	19.88	19.95	19.96	20	20	19.97	19.93
280	19.01	19.39	19.55	19.91	19.97	19.96	20	20	20	20	20
290	19.4	19.88	19.98	19.97	19.91	20	20	20	20	20	20
300	19.85	19.99	20	20	20	20	20	20	20	20	20

Figure 6: Ratio of average survival time, filtered greedy over user-centric

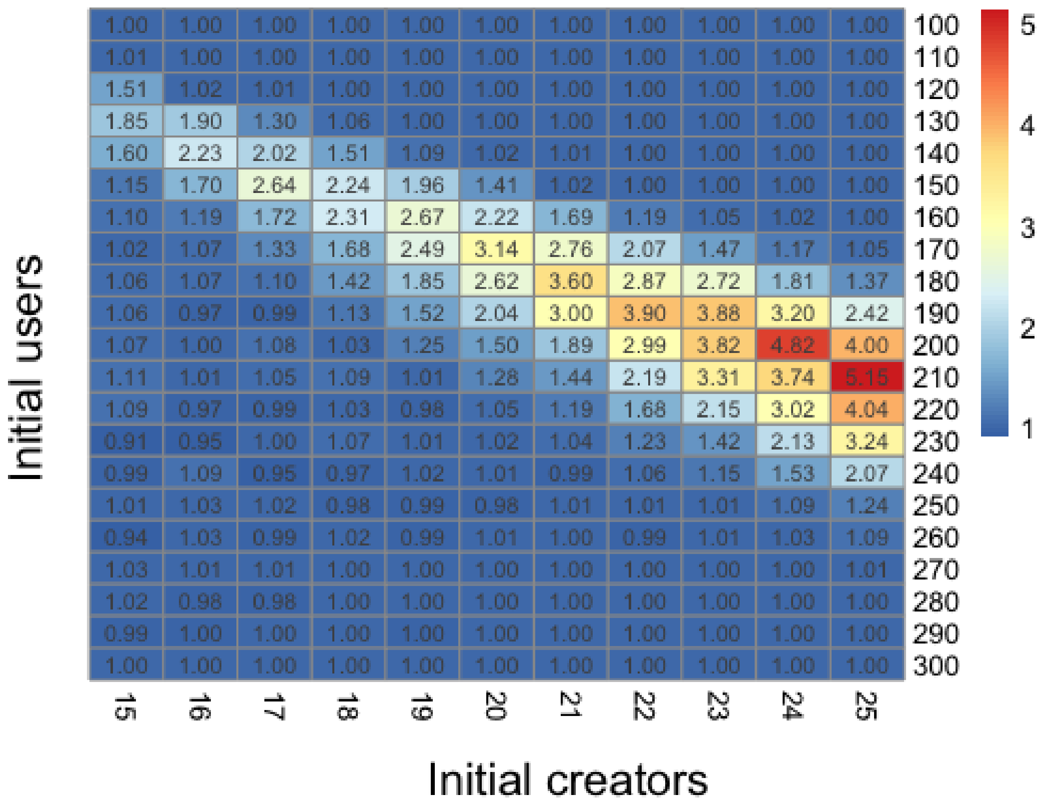


Table 7: Average final engagement under the user-centric algorithm.

U_{init} \ C_{init}	15	16	17	18	19	20	21	22	23	24	25
100	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
110	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
120	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
130	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
140	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
150	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
160	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
170	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
180	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
190	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
200	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
210	1.19	0.62	2.33	0.51	2.67	0.00	0.00	0.00	0.00	0.00	0.00
220	1.51	6.20	11.75	8.81	4.33	3.01	3.70	1.32	0.00	0.00	0.00
230	4.98	12.38	11.63	22.87	26.74	24.47	29.34	9.26	5.13	1.64	1.43
240	23.96	35.05	51.65	67.06	61.83	71.66	62.68	49.90	31.16	15.69	2.66
250	43.30	67.83	85.11	89.21	111.67	125.69	114.17	125.48	98.23	78.85	36.19
260	81.43	82.60	122.74	144.04	147.35	164.29	156.50	170.60	152.24	145.97	100.37
270	100.16	119.13	159.18	169.86	186.59	189.92	197.42	209.55	207.89	192.82	184.84
280	118.97	155.77	189.26	182.96	203.73	221.08	215.09	224.51	226.74	227.41	216.71
290	162.08	177.89	194.50	211.79	226.56	231.75	243.62	246.17	252.70	252.29	251.96
300	175.68	212.51	223.22	232.88	241.89	251.73	260.08	267.47	269.42	264.57	271.51

Table 8: Average final engagement under the filtered greedy algorithm.

		C_{init}										
		15	16	17	18	19	20	21	22	23	24	25
	100	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	110	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	120	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	130	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	140	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	150	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	160	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	170	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	180	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	190	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
U_{init}	200	0.00	0.34	0.00	0.00	0.00	0.37	0.51	0.00	0.00	0.00	0.00
	210	0.68	0.25	0.57	1.02	0.79	2.87	1.55	1.82	1.87	2.98	1.59
	220	7.40	10.81	9.58	10.51	7.47	4.07	10.86	14.87	10.85	10.76	11.09
	230	16.18	13.29	30.08	33.17	29.94	33.06	33.27	26.40	27.60	42.01	35.42
	240	25.27	35.06	46.17	57.23	71.11	73.29	65.53	74.00	61.37	64.35	82.35
	250	47.45	65.55	81.86	84.70	105.64	124.49	115.96	125.41	104.21	113.46	109.61
	260	68.78	92.71	109.38	128.77	152.14	163.29	163.25	175.60	166.05	157.43	154.50
	270	93.96	105.40	153.04	168.01	172.21	185.35	199.31	193.47	207.94	204.16	197.53
	280	135.55	157.61	173.71	189.88	210.91	213.16	219.28	226.64	224.04	233.29	237.07
	290	155.33	177.50	198.57	219.82	222.94	237.83	238.04	250.17	251.84	255.28	257.51
	300	173.97	201.43	221.12	231.18	251.14	254.79	259.67	260.95	265.51	272.00	267.00

Figure 9: Ratio of average final engagement, filtered greedy over user-centric. "NaN" represents cases where final engagement is 0 for both algorithms. "NA" represents cases where final engagement is 0 for user-centric but non-zero for filtered greedy.

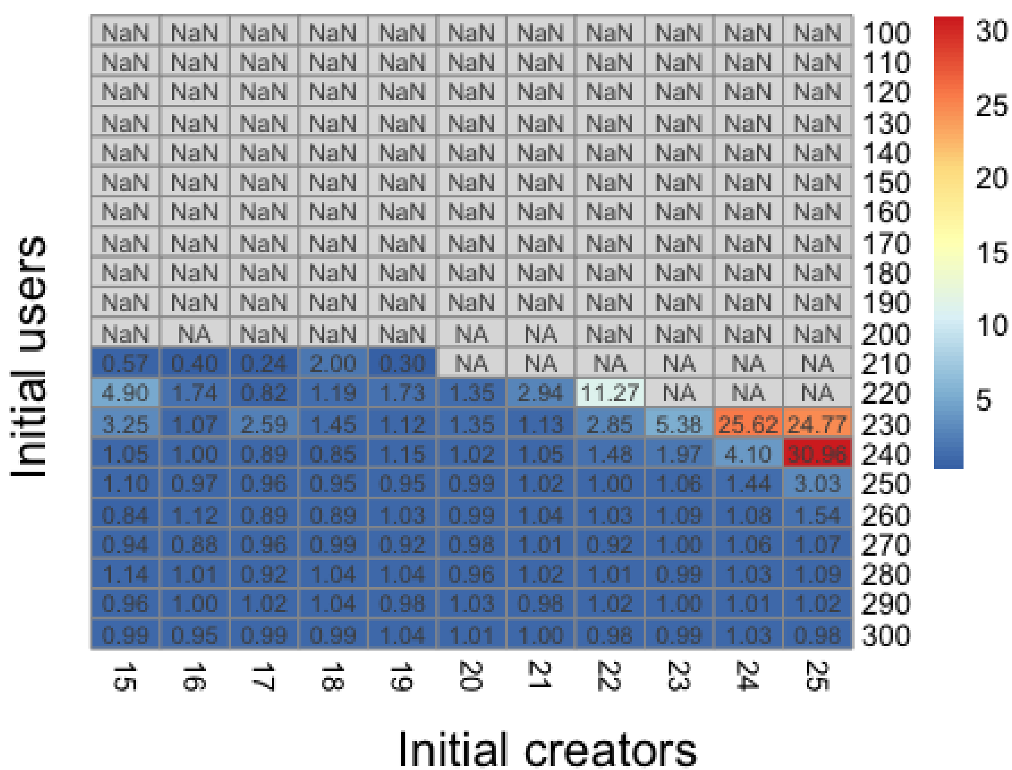


Table 10: Average turnover rate under the user-centric algorithm.

		C_{init}										
		15	16	17	18	19	20	21	22	23	24	25
	100	0.971	0.982	0.988	0.992	0.994	0.995	0.997	0.998	0.998	0.999	0.999
	110	0.938	0.962	0.975	0.984	0.989	0.991	0.994	0.996	0.997	0.997	0.998
	120	0.876	0.924	0.948	0.970	0.979	0.986	0.988	0.992	0.995	0.995	0.996
	130	0.782	0.856	0.909	0.938	0.960	0.973	0.979	0.986	0.990	0.992	0.994
	140	0.651	0.754	0.837	0.888	0.927	0.949	0.966	0.974	0.982	0.986	0.990
	150	0.498	0.639	0.746	0.827	0.880	0.913	0.936	0.956	0.968	0.977	0.982
	160	0.352	0.502	0.625	0.734	0.806	0.856	0.905	0.928	0.950	0.963	0.972
	170	0.226	0.366	0.501	0.621	0.718	0.791	0.844	0.892	0.917	0.937	0.954
	180	0.141	0.255	0.372	0.499	0.620	0.703	0.776	0.835	0.873	0.904	0.927
	190	0.082	0.157	0.267	0.388	0.503	0.604	0.693	0.761	0.823	0.862	0.900
U_{init}	200	0.046	0.099	0.182	0.286	0.391	0.501	0.601	0.681	0.754	0.806	0.850
	210	0.026	0.064	0.117	0.202	0.302	0.405	0.500	0.589	0.675	0.740	0.793
	220	0.013	0.032	0.073	0.139	0.213	0.314	0.406	0.504	0.590	0.661	0.730
	230	0.005	0.017	0.044	0.087	0.152	0.222	0.325	0.409	0.501	0.586	0.660
	240	0.004	0.011	0.024	0.059	0.104	0.174	0.244	0.324	0.413	0.504	0.583
	250	0.001	0.005	0.015	0.034	0.076	0.115	0.180	0.256	0.339	0.422	0.500
	260	0.001	0.003	0.009	0.020	0.047	0.077	0.135	0.194	0.268	0.342	0.427
	270	0.000	0.001	0.006	0.012	0.030	0.052	0.090	0.151	0.209	0.284	0.353
	280	0.000	0.001	0.003	0.007	0.018	0.039	0.071	0.111	0.162	0.225	0.288
	290	0.000	0.000	0.001	0.005	0.011	0.023	0.043	0.078	0.127	0.171	0.237
	300	0.000	0.000	0.001	0.002	0.006	0.016	0.028	0.058	0.088	0.130	0.182

Table 11: Average turnover rate under the filtered greedy algorithm.

		C_{init}										
		15	16	17	18	19	20	21	22	23	24	25
	100	0.971	0.982	0.988	0.992	0.995	0.995	0.997	0.998	0.998	0.998	0.998
	110	0.939	0.963	0.976	0.983	0.989	0.992	0.994	0.996	0.996	0.998	0.998
	120	0.687	0.913	0.949	0.970	0.975	0.986	0.989	0.992	0.994	0.995	0.996
	130	0.404	0.565	0.837	0.936	0.959	0.970	0.981	0.986	0.989	0.992	0.993
	140	0.433	0.394	0.513	0.748	0.901	0.943	0.962	0.975	0.982	0.987	0.990
	150	0.434	0.409	0.391	0.476	0.646	0.822	0.925	0.953	0.969	0.976	0.982
	160	0.334	0.414	0.413	0.378	0.451	0.598	0.745	0.890	0.938	0.961	0.970
	170	0.228	0.343	0.393	0.406	0.373	0.433	0.545	0.694	0.821	0.903	0.947
	180	0.144	0.248	0.334	0.376	0.388	0.373	0.416	0.528	0.650	0.781	0.871
	190	0.079	0.163	0.254	0.353	0.371	0.392	0.361	0.406	0.493	0.625	0.713
U_{init}	200	0.045	0.099	0.177	0.271	0.324	0.389	0.376	0.365	0.396	0.483	0.574
	210	0.023	0.057	0.117	0.194	0.279	0.333	0.352	0.361	0.362	0.386	0.473
	220	0.012	0.034	0.073	0.132	0.211	0.280	0.337	0.370	0.373	0.359	0.390
	230	0.007	0.015	0.044	0.086	0.147	0.214	0.288	0.343	0.362	0.372	0.353
	240	0.002	0.009	0.024	0.055	0.104	0.164	0.227	0.294	0.331	0.344	0.357
	250	0.001	0.006	0.015	0.034	0.070	0.118	0.181	0.239	0.296	0.336	0.353
	260	0.001	0.002	0.009	0.020	0.042	0.083	0.135	0.181	0.242	0.300	0.327
	270	0.000	0.001	0.005	0.014	0.027	0.057	0.089	0.134	0.194	0.252	0.291
	280	0.000	0.001	0.003	0.008	0.018	0.040	0.065	0.106	0.155	0.207	0.255
	290	0.000	0.001	0.001	0.004	0.012	0.024	0.043	0.074	0.122	0.164	0.209
	300	0.000	0.000	0.001	0.002	0.006	0.016	0.032	0.062	0.084	0.126	0.175

Figure 12: Ratio of turnover rate, filtered greedy over user-centric.

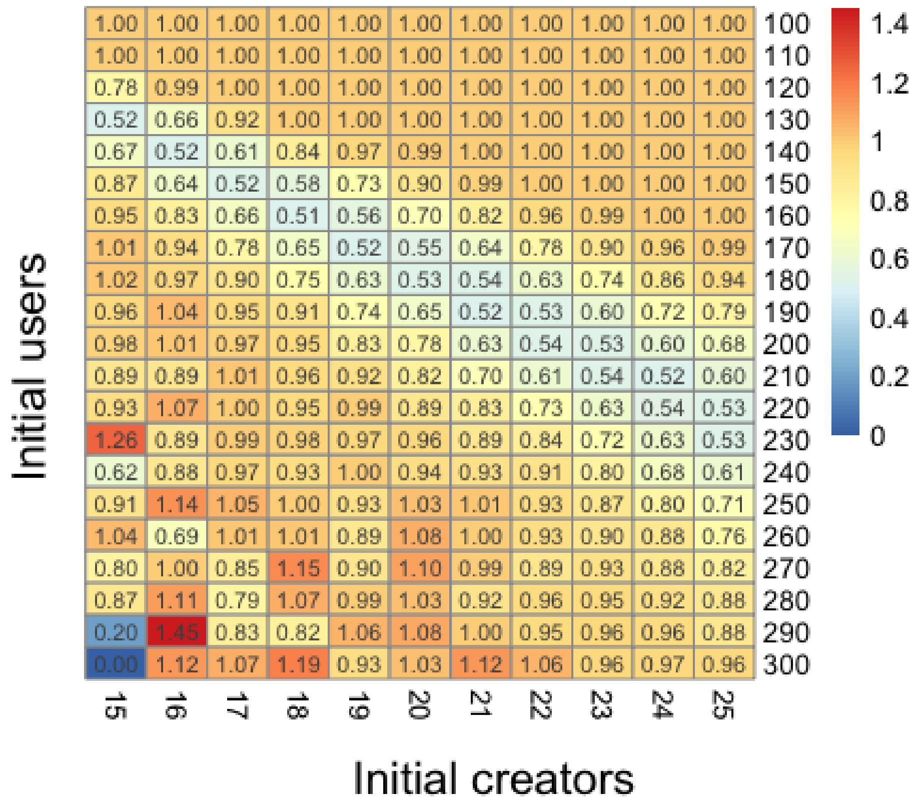


Table 13: Average total engagement under the local clustering algorithm.

						C_{init}					
		15	16	17	18	19	20	21	22	23	24	25
	100	88	91	95	95	97	98	98	99	99	99	100
	110	98	100	104	105	106	107	108	108	109	109	109
	120	107	110	114	114	116	116	118	119	119	119	119
	130	118	121	123	124	126	127	128	128	129	129	129
	140	145	133	132	134	136	136	137	138	138	139	139
	150	225	181	151	147	145	147	147	148	149	149	149
	160	320	259	196	171	159	158	158	158	159	159	159
	170	455	389	332	255	201	176	170	169	168	169	170
	180	642	622	515	409	288	223	201	187	181	181	179
	190	784	823	842	652	468	382	275	227	201	193	190
U_{init}	200	1053	1110	1155	1000	828	642	445	304	272	230	214
	210	1209	1300	1448	1524	1339	1052	830	619	403	312	273
	220	1479	1724	1884	1929	1983	1856	1443	1158	678	523	344
	230	1909	2110	2340	2441	2541	2478	2160	1729	1315	874	689
	240	2313	2760	3018	3074	3160	3237	3039	3108	2594	1748	1206
	250	2611	2945	3337	3567	3668	3903	3747	3628	3448	2844	2130
	260	3077	3475	3794	3976	4110	4130	4293	4355	4277	3919	3312
	270	3686	3852	4111	4301	4584	4565	4735	4716	4729	4677	4270
	280	3694	4184	4370	4722	4822	4898	4932	5081	5141	5123	5026
	290	4222	4404	4779	4914	4969	5154	5273	5252	5418	5397	5375
	300	4383	4762	4970	5178	5255	5433	5518	5546	5650	5633	5682

Table 14: Average total engagement under the creator ranking algorithm.

		C_{init}										
		15	16	17	18	19	20	21	22	23	24	25
U_{init}	100	89	91	93	96	97	98	98	99	99	99	100
	110	99	101	103	105	106	108	108	108	109	109	109
	120	106	111	113	114	116	117	118	118	119	119	120
	130	120	121	123	124	125	127	128	128	129	129	129
	140	148	136	132	133	135	137	137	138	138	139	139
	150	213	175	156	147	145	146	148	148	149	149	149
	160	329	276	205	174	165	157	157	158	159	158	159
	170	449	412	308	245	199	177	170	170	168	170	169
	180	615	570	496	369	288	225	206	189	180	184	180
	190	807	824	743	654	478	329	296	250	231	220	206
	200	959	1040	1063	928	816	604	471	422	358	352	314
	210	1204	1298	1401	1434	1188	1019	835	728	706	641	558
	220	1677	1666	1814	1838	1646	1614	1451	1339	1278	1147	1041
	230	1926	2122	2280	2485	2419	2386	2185	2362	2090	2047	1881
	240	2153	2473	2733	2842	3110	3103	2990	3031	2890	2902	2781
	250	2633	2942	3211	3462	3539	3606	3767	3732	3780	3647	3496
	260	3027	3327	3776	3837	3846	4194	4264	4303	4305	4104	4153
	270	3421	3732	4052	4267	4401	4419	4539	4586	4813	4705	4732
	280	3699	4051	4580	4722	4698	4952	4939	5003	5054	4962	4988
	290	3992	4457	4621	4952	5050	5148	5207	5196	5283	5381	5351
	300	4390	4673	4943	5212	5334	5503	5543	5520	5627	5527	5593

References

[1] Irina Beregovskaya and Mikhail Koroteev. *Review of Clustering-Based Recommender Systems*. 2021. arXiv: 2109.12839 [cs.IR].

[2] Talha Burki. "Vaccine misinformation and social media". In: *The Lancet Digital Health* 1.6 (2019), e258–e259. ISSN: 2589-7500. DOI: https://doi.org/10.1016/S2589-7500(19)30136-0. URL: https://www.sciencedirect.com/science/article/pii/S2589750019301360.

[3] Daniel Huttenlocher et al. *Matching of Users and Creators in Two-Sided Markets with Departures*. 2024. arXiv: 2401.00313 [cs.GT].

[4] Deddy Jobson. "Using Recommendations to balance demand and supply in two-sided marketplaces". In: *Workshop on Recommendation Ecosystems: Modeling, Optimization and Incentive Design*. 2024. URL: https://openreview.net/forum?id=ljtKZdRGu3.

[5] Gourab K Patro et al. "FairRec: Two-Sided Fairness for Personalized Recommendations in Two-Sided Platforms". In: *Proceedings of The Web Conference 2020*. WWW '20. ACM, Apr. 2020. DOI: 10.1145/3366423.3380196. URL: http://dx.doi.org/10.1145/3366423.3380196.

[6] Nasim Sonboli et al. "The Multisided Complexity of Fairness in Recommender Systems". In: *AI Magazine* 43.2 (June 2022), pp. 164–176. DOI: 10.1002/aaai.12054. URL: https://ojs.aaai.org/aimagazine/index.php/aimagazine/article/view/21742.

[7] Cai-Nicolas Ziegler et al. "Improving recommendation lists through topic diversification". In: *Proceedings of the 14th International Conference on World Wide Web*. WWW '05. Chiba, Japan: Association for Computing Machinery, 2005, pp. 22–32. ISBN: 1595930469. DOI: 10.1145/1060745.1060754. URL: https://doi.org/10.1145/1060745.1060754.