Amplification Pipelines

The Role of Feedback Loops in Recommender System Bias

Caio Truzzi Lente

Thesis presented to the
Institute of Mathematics and Statistics
of the University of São Paulo
in partial fulfillment
of the requirements
for the degree of
Master of Science

Program: Computer Science
Advisor: Prof. Dr. Roberto Hirata Jr.

São Paulo
December 30, 2022

Amplification Pipelines

The Role of Feedback Loops in Recommender System Bias

Caio Truzzi Lente

This is the original version of the thesis
prepared by the candidate Caio Truzzi Lente,
as submitted to the Examining Committee.

To Vicente Truzzi (in memoriam)

Resumo

Caio Truzzi Lente. **Canais de Amplificação: *O Papel da Retroalimentação no Viés de Sistemas de Recomendação***. Dissertação (Mestrado). Instituto de Matemática e Estatística, Universidade de São Paulo, São Paulo, 2022.

Algoritmos de recomendação tornaram-se essenciais para o funcionamento de diversos sistemas que usamos no dia a dia, desde quais filmes assistir até quais produtos comprar. Entretanto, com a proliferação destes modelos nas redes sociais, surgiram também novas preocupações. Evidências anedóticas e um corpo cada vez mais robusto de pesquisa têm indicado que os algoritmos das redes sociais, por valorizarem engajamento, podem estar radicalizando usuários através da criação das chamadas câmaras de eco. Este trabalho pretende estudar algoritmos de recomendação como sistemas dinâmicos de modo a identificar ciclos de retroalimentação degenerados.

Palavras-chave: sistemas de recomendação. viés algorítmico. aprendizagem de máquina.

Abstract

Caio Truzzi Lente. **Amplification Pipelines: *The Role of Feedback Loops in Recommender System Bias***. Thesis (Masters). Institute of Mathematics and Statistics, University of São Paulo, São Paulo, 2022.

Recommendation algorithms have become essential to various day to day systems we use, from what movies to watch to what products to buy. However, with the proliferation of these models on social networks, new concerns have come to light. Anecdotal evidence and an ever growing body of research indicate that social network algorithms that promote engaging content might be radicalizing users, creating what has become known as echo chambers. The present study aims to study recommendation algorithms as dynamical systems as a means to identify degenerate feedback loops.

Keywords: recommender system. algorithmic bias. machine learning.

List of Figures

List of Tables

Contents

Chapter 1

Introduction

Social networks have all but taken over contemporary daily life. From the eponymous socializing, to reading news, to expressing ourselves, social media has creeped into every corner of society. Most of its side-effects, it could be argued, are positive (shortening distances, political accountability, social organizing), but they are not perfect institutions.

Social media companies already face significant backlash for their questionable business model and ethics. Cambridge Analytica's election meddling (), Facebook's subliminal experiments (), YouTube's problem with disturbing content marketed at kids (), and Twitter's bot infestation () are just a few recent scandals that have put the societal role of social media into question.

One particular controversy that has taken over public discourse around social networks is the role that their algorithms might have in radicalizing users, specially younger ones. The aforementioned experiments conducted by Facebook to influence people's emotions and the proliferation of more than questionable videos aimed at children on YouTube are instances that seem to corroborate the notion that there is something fundamentally wrong with these companies' algorithms.

News organizations, in general, have been skeptical of social networks. Journalists and specialists alike argue that social media's algorithms (specially recommender algorithms) are tuned to peddle conspiracy theories, extremist views, and false information (). This would be the source cause for a plethora of what they consider contemporary evils: religious extremism, anti-democratic leaders, widespread depression among teenagers, anti-science movements, etc.

This narrative, of course, has been questioned for a variety of reasons. Some say that it is self serving: traditional news organizations are being displaced by social media and it would be convenient for them to mine the public's trust in them (Munger and Phillips, 2020). Others claim that these recommender algorithms are not to blame for political polarization and that social networks even have a tendency to favor more left-wing viewpoints (Ledwich and Zaitsev, 2019).

The debate around the role of recommender systems in social media radicalization

is still, unfortunately, too recent and based in anecdotes. Since its impacts are all but universal, more quality research is vital to inform both the public and opinion makers about if and how much recommendation algorithms influence social media users.

This dissertation aims to further such research.

1.1 Social Networks

Social networking services, also referred to as social networks and social media, are notoriously difficult to define. Some definitions might be too narrow (excluding instant messaging services), while some might be too broad (including technologies such as telephone networks). Most definitions () include some common features:

- Internet-based
- Focus on user-generated content
- Users have profiles
- Users can connect

While social-networking-like applications already existed in Usenet, Geocities, launched in 1994, is usually regarded as the first major social network. Friendster and Myspace followed in 2003, with Orkut and Facebook slightly lagging behind in 2004. Each hit their peak at different moments and different countries, but Facebook overtook all of them in 2009 when it became the most popular social networking service in the world, still maintaining the title over 13 years latter at the moment of writing ().

Even though all aforementioned social networks are multimedia, that is, users can post text, photos and videos, some of the most popular services focus on a specific type of media. For instance, YouTube (2009) centers around videos, WhatsApp (2009) and WeChat (2011) were originally designed for text-based communication, and Instagram's (2010) main focus is photos.

Parallel to all other features and idiosyncrasies, there lay the recommendation algorithms. While a few social networking services (e.g. WhatsApp) do not recommend any content or profiles to the user, most do and, according to recent studies, these recommendations have become the main drivers of interactions ().

1.2 Recommender Systems

Recommender systems (sometimes called recommendation systems or recommender algorithms) first appeared in 1992 under the name "collaborative filtering", even though that term nowadays refers to a subclass of recommender systems (Goldberg *et al.*, 1992). The aim of such an algorithm is providing users with personalized product or service recommendations, an essential task when considering the ever increasing number of possible videos to watch, music to listen, products to buy.

The input of a recommender system is usually information about the preferences (ratings, likes/dislikes, watch time, etc.) of consumers for a set of items. Preference information can be gathered from explicit behaviors (e.g. rating a product in a scale ranging from 0 to 5 stars) or from implicit behaviors (e.g. how much time the user lingers on a product's page). These data can be combined with information about the user (age, political leaning, etc.) in order to create the best possible representation of the user's preferences.

The output of these systems can come in the form of a prediction or a list of recommended items. In the first case, the goal of the algorithm is approximating the rating a user would attribute to a yet unrated item, while the second type of output involves gathering the items that most likely would interest the user. Simple recommender systems that suggest items similar to the one being queried do not necessarily involve rating predictions, but it is common to have the list of recommended items based on the ratings the algorithms estimated the user would give to those items.

Most recommender systems fall into one of four categories according to the filtering algorithm they use, that it, the strategy for generating predictions or selecting the top-N items: content-based filtering, demographic filtering, collaborative filtering, and hybrid filtering (Bobadilla *et al.*, 2013).

Content-based filtering leverages characteristics of the content in order to generate the recommendations (Ricci *et al.*, 2011). One such algorithm might use the genres of watched movies in order to recommend new ones, while another might analyze the sound signature of a song to recommend similar ones, but, either way, all content-based systems establish a similarity between items as a basis for recommendations. Analogously, demographic filtering uses demographic data to establish a similarity between users and recommend items positively rated by similar people.

Collaborative filtering algorithms also recommend items that similar users liked, but, in this case, the similarity between users is based on past ratings and not demographic information (Ricci *et al.*, 2011). Hybrid filtering usually mix collaborative methods with either content-based or demographic filtering (Ricci *et al.*, 2011).

As with other knowledge-based systems, recommendation algorithms have quickly incorporated neural networks and other machine learning techniques over the past few years. Even though the implementation of YouTube's recommendation algorithm is a trade secret, it is known to gather enormous amounts of data about the user's interaction with the website and to require Google's own TPUs in order to be trained. It also involves two distinct steps: candidate generation (when the billions of videos available on the platform are quickly narrowed down to a few hundreds that might be relevant) and ranking (when the algorithm actually attempts to predict the score a user would implicitly give to the candidate videos) ().

Another relevant aspect of recommender systems that is well-exemplified by YouTube is the use of balancing factors such as novelty, dispersity, and stability (Zhao *et al.*, 2019). In the case of Google's video giant, there is a baked-in bias for recency, strongly favoring newer videos in detriment of older content (Zhao *et al.*, 2019).

From this kind of bias stems much debate: as recommender systems explode in popularity, so does research regarding its shortcommings. User radicalization and algorithmic

bias (explicitly programmed or not) are hotly debated subjects in the literature.

1.3 Radicalization and Bias

Opinion polarization is far from a recent phenomenon, and social media is only the most recent communication medium where it can be detected and studied. An important question is whether it facilitates or attenuates polarization: anecdotal evidence might suggest that social network structures incentivize users to gather into antagonistic communities, but this could be a result of people simply being more likely to express their preferences online, not of some intrinsic property of social media.

One possible byproduct of polarization is radicalization. Despite not being entirely different phenomena, these concepts deserve distinct levels of attention. While polarization can be considered a natural part of democratic discourse, radicalization only happens when certain conditions are met. UNESCO defines radicalization as (Séraphin *et al.*, 2017):

- The individual person's search for fundamental meaning, origin and return to a root ideology;
- The individual as part of a group's adoption of a violent form of expansion of root ideologies and related oppositionist objectives;
- The polarization of the social space and the collective construction of a threatened ideal 'us' against 'them,' where the others are dehumanized by a process of scapegoating.

The third point is of special importance to the distinction between polarization and radicalization. The first might be a simple consequence of democratic disagreements between opposing parties, but the latter involves a dehumanization of the opposition, which can lead to extremism: radicalism so intense that the only effective strategy is physically exterminating the opposition.

Understanding how polarization might lead to radicalization (and, ultimately, to extremism) is, therefore, of paramount significance to cultivate healthy democracies, specially in the digital age. Since most social networks, as of this writing, are still poorly moderated, they allow users to be exposed to a plethora of viewpoints, from benign to insidious, possibly configuring a "pipeline of radicalization" through which regular users end up radicalized by coming into contact with extreme content (M. H. Ribeiro *et al.*, 2020).

Of course this argument is still very much open for debate. As will be shown in Chapter 2, researchers have found evidences both for and against the pipeline hypothesis and even proposed other means though which social media might help radicalize users (e.g. the supply and demand hypothesis (Munger and Phillips, 2020)). Despite all disagreements, one common point addressed by most research is the role of recommendation algorithms in serving users with radicalizing content.

Proponents of the pipeline hypothesis, for instance, argue that recommendation systems, aiming to maximize content consumption, suggest items that reinforce preconceived notions of the user and that play on fear and paranoia (M. H. Ribeiro *et al.*, 2020). Content that appears urgent and leaves the user fearful (for their live, their community, or their

identity) could be more engaging and, therefore, might be more susceptible to being considered as relevant by the algorithm.

Even if the pipeline hypothesis is correct, specifics of how much algorithms are to blame for radicalization are still unknown and hard to pin down. Most research about the subject focuses on specific platforms (like Twitter and YouTube) and have severe limitations with regards to how much data those companies make available, not to mention the constant changes made to the algorithms over the years that might alter their radicalization properties. Definitive evidence for one theory or another must, therefore, apply to recommender systems in general and be predictive of how they work both in controlled and real life scenarios.

YouTube, for example, currently has over 2 billion monthly logged-in users (), but it makes no significant effort to clarify changes made to the algorithm or even whether they fulfill their promises of reducing user exposure to radicalizing content. With more than 500 hours of content being uploaded every minute (), if 1% of all videos can be considered radicalizing and the algorithm can detect 99% of them, that still leaves over 25.000 hours of brand new extremist content free to spread on the platform every year. This goes to show that, in the scale that these companies operate, even a small fraction of content might still be enough to influence the overall recommendations made by the algorithm. It is also worth noting that most of these platforms' efforts are concentrated in their parent countries (usually the United States), so, even if they actually try and remove the offending content, most of the non-English-speaking world would still not be impacted by their policy changes.

Closely related to user radicalization is the subject of algorithmic bias. YouTube, for example, has an explicit bias towards recency (Covington *et al.*, 2016), meaning that more recent videos get "boosted" by their recommendation algorithm. This is explicitly coded into the system, but there are also implicit biases, learned by watching user behavior.

Stoica (Stoica, Riederer, *et al.*, 2018) studied Instagram profiles before and after the implementation of their recommendation engine. They discovered that male users had a slight predilection for following other men, while women displayed no such preference and, as soon as the recommender system was deployed, engagement with profiles of male users skyrocketed even though they were the minority on the platform. The algorithm recognized and leveraged this so called differentiated homophily effectively, but we might question whether or not this should be the desired outcome of a good recommender system.

In social networks where recommendations are front and center, such as YouTube, the algorithm could go from being a mere reflection of user preferences to actively shaping user behavior. Hypothetically, a minoritary group of highly engaged users with strong self-reinforcing consumption habits could tip the scales of the algorithm and cause fringe content to be amplified; this is only one way through which a radicalization pipeline could spontaneously form in a social network.

1.4 Research Goals

As explained in the previous sections, social networks' recommendation algorithms might play a significant role in radicalizing users. This could be, at least in part, be a result of implicit and explicit biases in recommender systems.

This dissertation aims to explore the radicalization pipeline hypothesis and, more specifically, understand the mechanisms through which recommender systems can end up learning or developing biases. The research developed here revolves around the dynamical properties of recommender systems (i.e., the sequence of items suggested to an arbitrary user over time) and how feedback loops can create "amplification pipelines" inside these engines.

In short, the main motivator of this research is to test the pipeline hypothesis in a setting where recommendation algorithms learn dynamically.

1.5 Outline

...

Chapter 2

Literature review

There are three types of work that are relevant to the current topic: general literature about recommender systems, evidences of algorithmic bias, and methods of creating fairer recommendations. Since this area of study is still mostly unexplored, there is no consensus on whether social media recommender systems favor extremist content (or even whether they are actually deradicalisation agents), which means that many references used in this work might disagree amongst themselves.

2.1 Scientific literature

General literature about recommendation algorithms is abound. One of the most cited surveys was elaborated by Bobadilla *et al.* (2013), but works by He *et al.* (2016) (about interactive recommender systems), and by Kunaver and Požrl (2017) (about diversity in recommender systems) were also used in order to draw a complete panorama of the field.

Another relevant article, by Guy *et al.* (2010), is the landmark paper that inaugurates the usage of user data alongside labels to create a recommendation algorithm that is highly accurate and a staple of modern social networks. This essentially starts the usage of recommenders systems in social media.

When talking specifically about YouTube's recommendation algorithms, two papers deserve special attention. The first one, by Covington *et al.* (2016), marks YouTube's move towards the usage of deep neural networks to generate video recommendations. The authors describe a two-stage model that first generates a list of candidates and then ranks them, also reporting dramatic performance improvements. The second one, by Zhao *et al.* (2019), describing a more recent version of YouTube's recommendation algorithm, explores the Multi-gate Mixture-of-Experts technique to optimize recommendations for more than one ranking objective and the Wide & Deep framework to mitigate selection biases. The authors also make it clear that YouTube's recommender system has a strong bias towards more recent content instead of more traditional metrics.

Many authors have also explored how biases in recommendation engines might lead to user radicalization. Agarwal and Sureka (2015) developed an early example of a

technique to try and find extremist content on YouTube. Using advanced machine learning methods, the authors create a YouTube crawler that starts from a seed video and iteratively classifies featured channels and videos according to their potential extremism. A more recent example of this can be found in Tangherlini *et al.* (2020), where the authors propose a novel approach for identifying conspiracy theories online. By analyzing the narrative structure of a conspiracy theory (Pizzagate) and comparing it to an actual conspiracy (Bridgegate), they create a model that can guess whether a conspiratorial narrative is or not fabricated. According to their findings, a multi-domain nature and the presence of keystone nodes are signs that strongly indicate a conspiracy theory.

Besides just finding and identifying radicalizing content on YouTube, many authors have been concerned with studying the radicalization dynamics directly. Alfano *et al.* (2020), for example, claim to be "the first systematic, pre-registered attempt to establish whether and to what extent the recommender system tends to promote such [extremist] content." Cho *et al.* (2020) also attempt to understand how users can be radicalized by the algorithm. By experimentally manipulating user search/watch history, the authors concluded that algorithmically recommended content can reinforce a participant's political opinions.

In the same vein, Faddoul *et al.* (2020), after some high-profile cases of users being radicalized through YouTube videos, studied the efforts announced by the platform to curb the spread of conspiracy theories on the website. The paper aimed to verify this claim by developing both an emulation of YouTube's recommendation algorithm and a classifier that labeled whether a video is conspiratorial or not. The authors describe an overall decrease in the number of conspiracy recommendations, though not when weighing these recommendations by views.

Three papers that deserve a closer look are those that investigate how regular recommendation algorithms can learn covert biases in the users of a social network and amplify them to previously unimaginable rates. Stoica, Riederer, *et al.* (2018) explore the existence of an "algorithmic glass ceiling" and introduces the concept of differentiated homophily. The authors experiment on a Instagram dataset before and after the introduction of algorithmic recommendations and discover that, even though most of that network's users were female, the most followed profiles were male. They explain this phenomenon by postulating that the algorithm learns biases in the population, that is, male preference for male profiles (which doesn't happens for females and thus characterizes an asymmetric—differentiated—homophily), and ends up enhancing this effect. Stoica and Chaintreau (2019), building on top of their previous work, create a proposal for new recommender systems that take differentiated homophily into account in order to reduce the "glass ceiling" effect observed in non-corrected recommendation algorithms. The work focuses on the theoretical description of the algorithm, but also attempts to validate its hypothesis in real world data. Stoica (2020), in their most recent paper, show that the most commonly used metrics in recommender systems "exacerbate disparity between different communities" because they reinforce homophilic behavior of the network. This has profound implications, since these algorithms might further suppress already minoritary viewpoints without being explicitly programmed to do so.

Like the aforementioned articles, Matakos *et al.* (2020) also propose a novel recommen-

dation algorithm that tries to strike a balance between information spread and ensuring that the users are exposed to diverse viewpoints. The authors show that this goal is important if we want to foster healthy online debate, and that the algorithm is efficient and scalable with a minor approximation. One possible inspiration for these papers might be one by Su *et al.* (2016) that studied the network structure of Twitter before and after the introduction of algorithmic recommendations ("Who to Follow"). The authors of the paper discovered that all users benefitted recommendations, but that users with already popular profiles benefitted even more, effectively changing the network structure and dynamics. Caton and Haas (2020) have recently compiled other valuable information on fairness in machine learning into a survey.

Because of data limitations, there still are few studies that investigate how recommendation algorithms work dynamically, over time. Burke (2010) point out that most methods for evaluating recommender systems are static, that is, involve static snapshots of user and item data. The authors propose a novel evaluation technique that helps provide insight into the evolution of recommendation behavior: the "temporal leave-one-out" approach. A more recent example of this approach was developed by Roth *et al.* (2020). Their paper delves into the confinement dynamics possibly fostered by YouTube's recommendation algorithm. The authors create, from a diverse set of seed videos, a graph of the videos iteratively recommended by YouTube and, from this, study whether there were created "filter bubbles". They find that indeed YouTubes recommendations are prone to confinement dynamics be it topological, topical or temporal.

Even more recently, Yao *et al.* (2021) propose an approach for measuring recommender system bias based on simulated users. Even though this work focuses only on bias towards popular content, it is of particular importance because it was written by researchers from Google itself. Some years before, Dash *et al.* (2019) also proposed a framework for auditing recommender systems based on its network of users. Another contribution of their work is a novel quantifications of diversity.

A different approach to understanding biases in recommendation algorithms range from analyzing similarity metrics to developing theoretical bounded confidence models. Giller (2012) goes with the first strategy, and identifies certain aspects of cosine similarity that are often overlooked. Starting from simple theorems regarding the density of n-dimensional spheres, the author concludes that the expected cosine similarity between random bitstreams might be significantly different from the average. This is noteworthy because many recommendation algorithms use cosine similarity in order to determine the similarity between two items to recommend. Sîrbu *et al.* (2019) go with the latter, providing an interesting theoretical model of how inherent biases in algorithmic recommendations might heighten opinion polarization. Using a bounded confidence model, the authors propose the addition of a γ term that represents the odds of an algorithm recommending content that differs from that of a user.

Some recent papers also try to understand how YouTube might be favoring right-wing and fascist content in specific, as opposed to trying to prove a more general (and possibly less tractable) claim. Hosseinmardi *et al.* (2020) find evidence via a longitudinal study that there exists "a small but growing echo chamber of far-right content consumption" on YouTube. According to their research, these users are more engaged than other, with

YouTube generally accounting for a larger share of their online news diet than the average. The authors, however, find no evidence of this phenomenon being due to recommendations. A popular article in the field, by M. H. Ribeiro *et al.* (2020), explored the radicalization pipeline hypothesis of algorithmic enabled radicalization. The authors collect huge amounts of YouTube comment data over time, and determine a significant migration of users from "lighter" content towards more extreme videos. This does not prove that the pipeline exists, but is a strong argument for its existence.

Twitter was also found to consistently favor right-wing content. Huszár *et al.* (2021) conducted a "long-running, massive-scale randomized experiment" across 7 countries in order investigate the effects of algorithmic personalization on users' feeds and, according to their results, "mainstream political right enjoys higher algorithmic amplification than the mainstream political left".

Finally, feedback loops are of special interest to this discussion. Caused by the inevitable fact that recommender systems must learn from users' reactions to its own recommendations, they are a widely believed to be a powerful engine of bias amplification and are discussed at length in the literature. Already in the last decade, Sinha *et al.* (2017) investigated the viability of identifying items affected by these feedback loops and attempted to created a method of deconvolving them. More recently, Jiang *et al.* (2019) explored what they called "degenerate feedback loops" and their capability of creating echo chambers, going as far as proposing a novel approach of slowing down this tendency towards degeneracy. In a related study, Mansoury *et al.* (2020) explored how recommender systems amplify already popular content, but, more importantly, how this tendency might reduce content diversity and cause users' tastes to shift over time. Depending on what a systems values (recency, virality, controversy, engagement), this type of feedback loop could possibly amplify not "popular" content, but divisive and extremist content.

A minority of papers tries to disprove the hypothesis that social networks in general, and YouTube in specific, have a radicalizing tendency. Munger and Phillips (2020) published a controversial article that postulates a new model for YouTube radicalization. According to the authors, YouTube's algorithm is not to blame, the users themselves are looking for extreme content and the recommender system only supplies them. Its methods were highly questioned by the community and is currently the only paper that spouses the supply and demand hypothesis. Ledwich and Zaitsev (2019) also wrote a highly controversial paper where its authors claim to have found evidence to support the hypothesis that YouTube's recommendation algorithm favors mainstream and left-leaning channels instead of right-wing ones. They categorize almost 800 channels into groups of similar political leaning and analyze recommendations between each group, finding that YouTube might actually discourage users from viewing radicalizing content. Most researchers though do not support the methods employed by these two articles. In an even earlier study on news recommendations of a major Dutch newspaper, Möller *et al.* (2018) claim that recommenders systems had no significant impact on content diversity.

Even with a quickly growing body of research, further studies are needed in order to shed more light into the inner workings of how recommendation algorithms are used by social networks. Articles like the ones described in this chapter are of utter importance to this task, but generalist studies that are able to capture dynamics common to all or most

recommender systems are still nonexistent.

2.2 Journalistic efforts

Since this field of study is still in its infancy, many relevant sources are not scientific in nature. Journalism, specially when investigative in nature, is a valuable ally when trying to understand what is happening behind the curtains of social platforms.

Some examples of journalistic endeavors that inform and guide scientific research include, but are not limited to, a series by Lecher and Yin (2022) on how different are Americans' Facebook feeds, a report (in Portuguese) by P. V. Ribeiro (2021) on how the far-right is still able to cheat YouTube's attempts at curbing extremist content, and a whistleblower's account to Wong (2021) of how Facebook's executives resist on restricting fake engagement that is able to distort global politics.

Chapter 3

Static Analysis

Understanding how social networks recommend content to users is central to the debate around political polarization and radicalization. There are many ways of exploring recommender systems without examining their code, from simulating their behavior after careful observation (Yao *et al.*, 2021) to directly collecting recommendation data (M. H. Ribeiro *et al.*, 2020), but most of them allow us to examine only one perspective of the algorithm at a time. This means that studying a social network's recommendation technique has inherent limitations.

Most of the algorithms currently employed by social media companies are trade secrets. They are also subject to constant experimentation and tuning (), which might render worthless any research performed before an update to the algorithm, no matter how careful the design of the study was.

With our experiments we aim to make a tangible contribution to the field of recommender systems, specifically how their design might (or might not) foster confinement dynamics and create degenerate "feedback loops". If the main hypothesis is confirmed, this could mean that even a relatively small fraction of the content can tip the algorithm in its favor, amplifying their message, creating filter bubbles, and possibly sending users on a radicalization spiral if that topic is related to politics or other contentious subjects.

In this chapter we will analyze a recommendation system statically, that is, without taking into account its evolution after multiple rounds of training and learning from new data. This step is essential insofar as it generates valuable information to better orient our dynamic analyses.

3.1 Datasets

Before discussing any experiment, it is necessary to introduce the datasets used in the models. The main dataset explored in this dissertation is MovieLens (Harper and Konstan, 2015), a well-known set of movie reviews that has been featured in many recommender system tutorials and papers for the past few years. The full dataset, with 27,000,000 ratings applied to 58,000 movies, was enriched by Banik (2017) with information about the movies'

credits, metadata, keywords, and links. The first five rows of the dataset are reproduced in Table 3.1.

user_id	movie_id	rating	timestamp
1	1193	5	978300760
1	661	3	978302109
1	914	3	978301968
1	3408	4	978300275
1	2355	5	978824291
1	1197	3	978302268

Table 3.1: *First five rows of the MovieLens dataset.*

In the end, because of technical limitations, the dataset used in this chapter was sampled until 30,689 movies were left; this was the largest dataset that could be processed in less than one hour by the hardware we had access to.

The second dataset, used for validation purposes only, was the Book-Crossing Dataset (Ziegler, 2004). Just like the enriched MovieLens, this dataset contained entries for ratings (1,149,780) applied by users (278,858) to items (271,379 books), and information about these items like title, author, publisher, etc.

3.2 Experiments

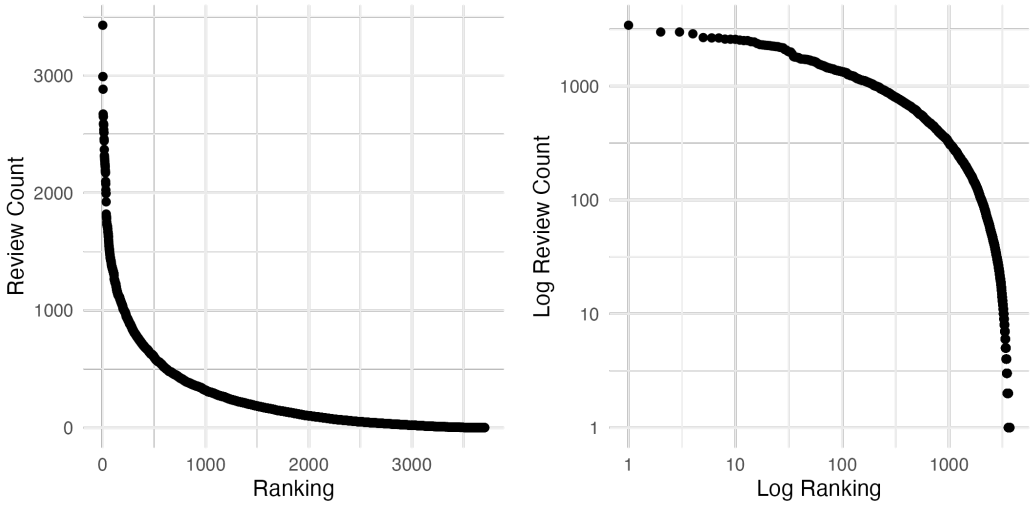
The static analysis involved crafting many different models and analyzing their recommendations as a whole. The goal of these experiments was to test the hypothesis that even a simple recommendation algorithm could demonstrate some sort of bias towards a subset of the items being recommended. More specifically, given an algorithm that is user agnostic, i.e., that cannot be influenced by users' personal preferences, would the resulting recommender system favor some item? Excluding user information is important because, as demonstrated by Stoica, Riederer, *et al.* (2018), users might have their own biases and these could get transferred on to the model; the objective here is understanding the algorithm by itself without external influences.

To evaluate the recommendation models, at least qualitatively, we can compare their "recommendation profiles": a summary of how many times every item is recommended overall. To create this profile, the model is asked to return the top-n most similar items to the input according to its internal similarity metric, and this process is repeated for every item in the dataset. The recommendation profile of the model is the number of times each item showed up in the top-n most similar items of the whole dataset.

For example, a recommendation algorithm applied the present version of the MovieLens dataset would generate a list of n movies for each input, creating a list of $30,689 \times n$ movies. This operation is very costly, approaching $O(MN)$ complexity, where M is the number of users in the dataset and N is the total number of available numbers. After this computationally intensive calculation, the occurrence of each ID would be counted and ranked accordingly to facilitate interpretation of results, that is, the movie ranked number 1 would be the movie featured the most times in the set of all recommendations, and so

forth for every other rank. From now on n will be fixed to 10, which is, on average, roughly how many videos YouTube shows on its recommendation sidebar.

The baseline for these visualizations is MovieLens' original review distribution. Figure 3.1a ranks every movie in the original dataset by their review count, i.e., the number of users that reviewed each movie. It is important to note that we're not taking user ratings into account for these profiles and this is deliberate: social networks like YouTube seem to mostly ignore explicit user feedback when making recommendations. Our models will, therefore, use ratings internally, but we will evaluate them based on impressions alone.



(a) *Review profile.* **(b)** *Log-log plot.*

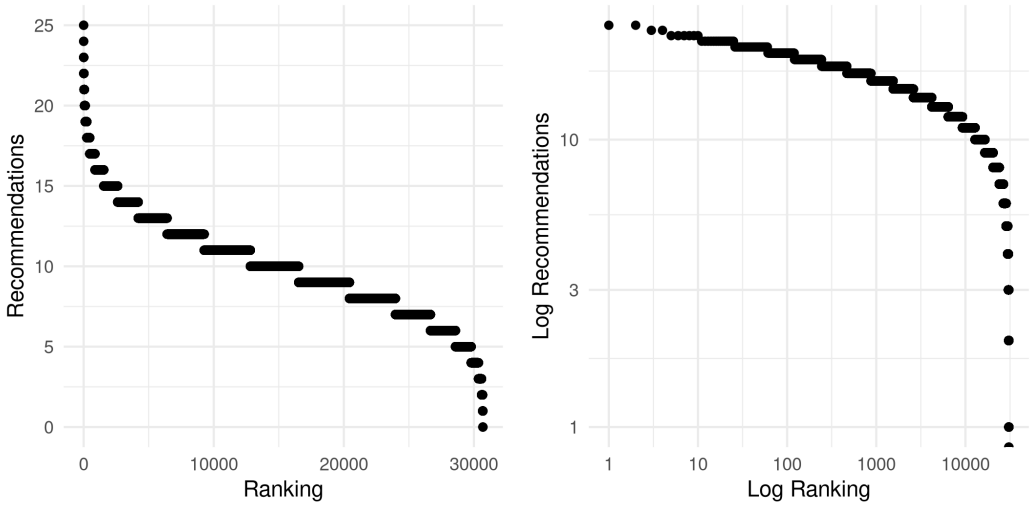
Figure 3.1: *Review profile for the full dataset (a) and log-log plot (b).*

3.2.1 Trivial Model

The first model examined is the "trivial model", which is a simple sampler that returns n movies at random when asked for a recommendation. Its recommendation profile can be seen in Figure 3.2 and, since it is the sum of many uniform samples, the number of times each movie is recommended approaches a normal distribution and, therefore, the recommendation profile also approaches the cumulative distribution function (CDF) of said distribution. The most recommended movie appeared 25 times in the final list, while the least recommended movie did not appear at all. Figure 3.2b shows the log-log plot of the recommendation profile.

3.2.2 Vanilla Model

Besides the trivial model, the simplest model that excludes user information is the content-based recommender. In the real world this is an algorithm that is able to identify similar items based on their metadata (description, tags, etc.) and suggest the closest items to the one being purchased or viewed. A straightforward way of building such an algorithm is creating a vector representation of each item and then using a similarity



(a) *Recommendation profile.* **(b)** *Log-log plot.*

Figure 3.2: *Recommendation profile for the trivial model (a) and log-log plot (b).*

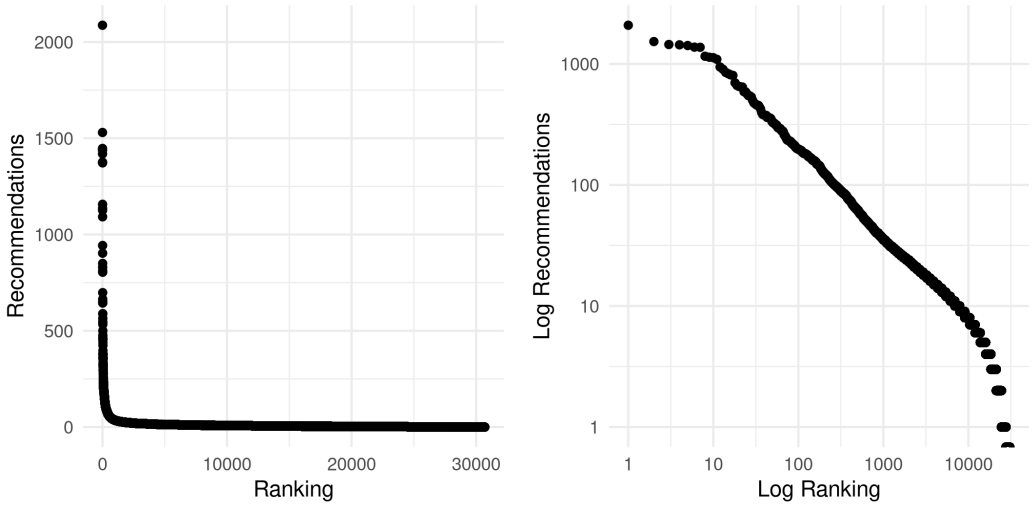
metric to recommend the items most similar to the one in question. The chosen similarity metric was cosine similarity because of its simplicity, robustness, and ubiquity (Sarwar *et al.*, 2001).

The main non-trivial model used in this study was the one that simply generated vector representations for the full MovieLens dataset, without any modifications (which is why it will henceforth be referred to as the "vanilla model"). The metadata for each movie was made up of its keywords, main cast, director, and genres. The vector transformation was very simple, with each position representing one of the words of the corpus, and each element indicating how many times that word appeared in the metadata for that movie. When the recommendation for a movie was requested, the algorithm measured the cosine similarity between it and every other movie, returning the IDs belonging to the top $n = 10$ most similar vectors.

The recommendation profile for the vanilla model can be seen in Figure 3.3a. Here, the movie ranked number 1 appeared more than 2000 times in the full list of recommendations, with an almost exponential decrease in the number of appearances from then on, as made evident by the log-log plot on Figure 3.3b. This recommendation profile is a big departure from the trivial model discussed above.

3.2.3 Cutoff Models

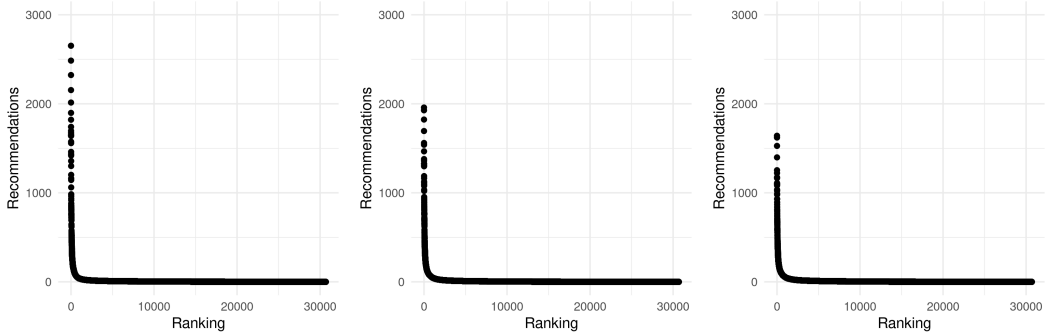
A potential explanation for the difference between trivial and vanilla could reside in the least used terms in the metadata: movies whose metadata share rare words might have been recommended less frequently than movies whose metadata are not so unique. To test this hypothesis, a cutoff point was created for words to be included in the vector representation of the movies. Three cutoff points were tested and only words with an absolute frequency larger than or equal to k, $k = \{2, 5, 8\}$, could be included in the vector representations. The results can be seen in Figure 3.4 and, aside from variations in the y-intercept, all plots are qualitatively very similar to Figure 3.3a, indicating that rare words



(a) *Recommendation profile.* **(b)** *Log-log plot.*

Figure 3.3: *Recommendation profile for the vanilla model (a) and log-log plot (b).*

probably are not to blame for the exponential-like decay.



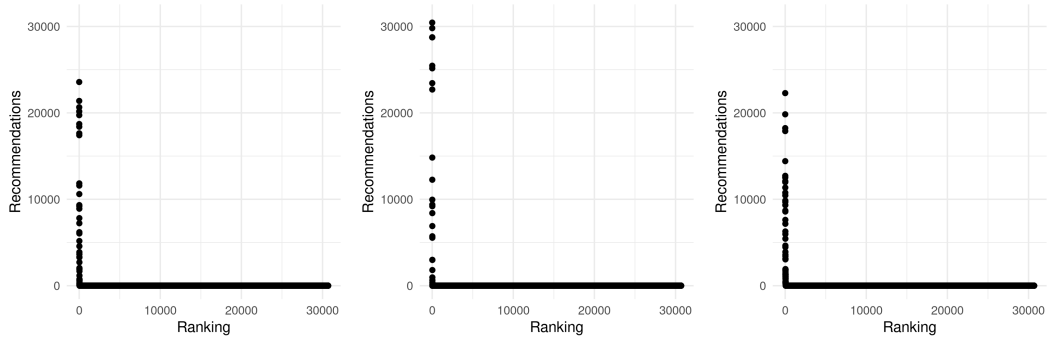
(a) *Cutoff* $k = 2$. **(b)** *Cutoff* $k = 5$. **(c)** *Cutoff* $k = 8$.

Figure 3.4: *Recommendation profile for cutoff* $k = 2$ *(a),* 5 *(b), and* 8 *(c).*

3.2.4 Similarity Models

The second validation experiment involved attempting to use other distance metrics instead of cosine similarity (Ricci *et al.*, 2011), since that could also be a source of the strange behavior of the recommendation profile. The goal here was to verify whether other metrics could do a better job at not creating a subset of movies that ended up exponentially more recommended than the rest.

Figure 3.5 showcases a comparison between cosine distance, euclidean distance and manhattan distance. It is clear that there are no meaningful differences between the recommendation profiles generated by each metric.



(a) *Cosine distance (vanilla).* **(b)** *Euclidean distance.* **(c)** *Manhattan distance.*

Figure 3.5: *Recommendation profile for cosine (a), euclidean (b), and manhattan (c) distances.*

3.2.5 Vanilla Model with Synthetic Metadata

At this point it is safe to say that the type of decay seen in recommendation frequencies up until now is not spurious and must have a clear cause. To better investigate whether word frequency had an impact on the recommendation profiles another hypothesis was taken into consideration: do sparser metadata cause the recommendation curves to display a steep left-hand side?

For the next experiment, each movie was assigned randomly generated metadata, i.e., each text was comprised of random words sampled uniformly from the full metadata corpus. The baseline probability of any given word w_i being added to the metadata of a movie was equal to $\overline{P(w)}$, the average probability over every word in the original corpus.

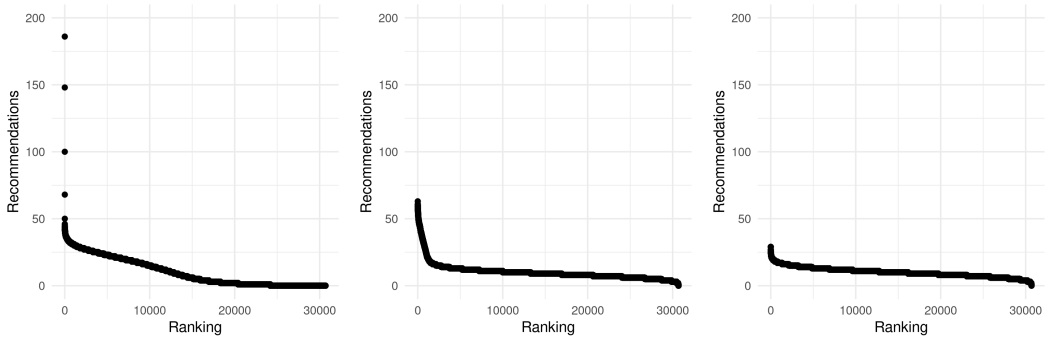
In addition to this baseline probability, two extra sets of metadata were constructed: one where each word was 10 times more likely to be selected than $\overline{P(w)}$, and another where each word was 100 times more likely. All three constructed datasets can be concisely described through the expression $P(w_i) = C \times \overline{P(w)}, C = \{1, 10, 100\}$.

Figure 3.6 displays the recommendation profiles for the vanilla model applied to datasets with each distinct value of C. Concretely, the figures are equivalent to creating random metadata for the movies where the probability of any single word being selected was approximately 1.54×10^{-4}, 1.54×10^{-3}, and 1.54×10^{-2} respectively. The results do support the aforementioned hypothesis since less sparse vectors indeed generated less exponential decays.

3.2.6 Sanity Checks

After the previous experiments, sanity checks were needed in order to guarantee that the previous hypothesis was generalizable. The first check should verify whether an artificial movie created as a combination of the metadata from other movies favored by the recommendation algorithm would also be favored, while the second should check whether shorter vectors would change the decay already observed despite being as sparse as their longer counterparts.

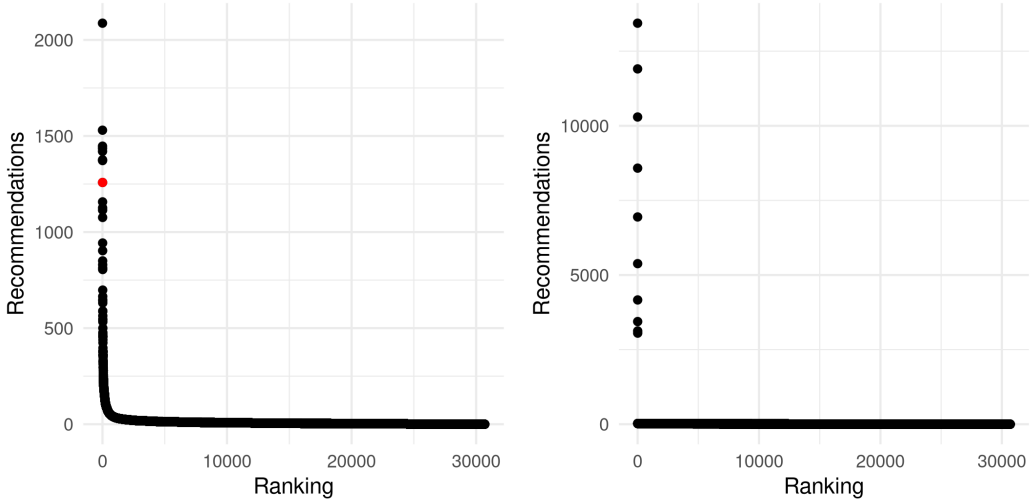
Figure 3.7 showcases the two sanity checks. Figure 3.7a was a model applied to the vanilla dataset with the addition of the movie highlighted in red. As expected, this movie



(a) $P(w_i) = 1 \times \overline{P(w)}$. **(b)** $P(w_i) = 10 \times \overline{P(w)}$. **(c)** $P(w_i) = 100 \times \overline{P(w)}$.

Figure 3.6: *Recommendation profile of samples with* $P(w_i) = C \times \overline{P(w)}$, $C = 1$ *(a),* 10 *(b), and* 100 *(c).*

also showed up in the top-recommended subset. Figure 3.7b comes from a model applied to randomly generated vector representations in a similar fashion to the ones in Figure 3.6, except each vector could only have 15,000 elements instead of 55,681 (as with the vanilla model). The pattern observed before persisted, meaning that the hypothesis still stood.

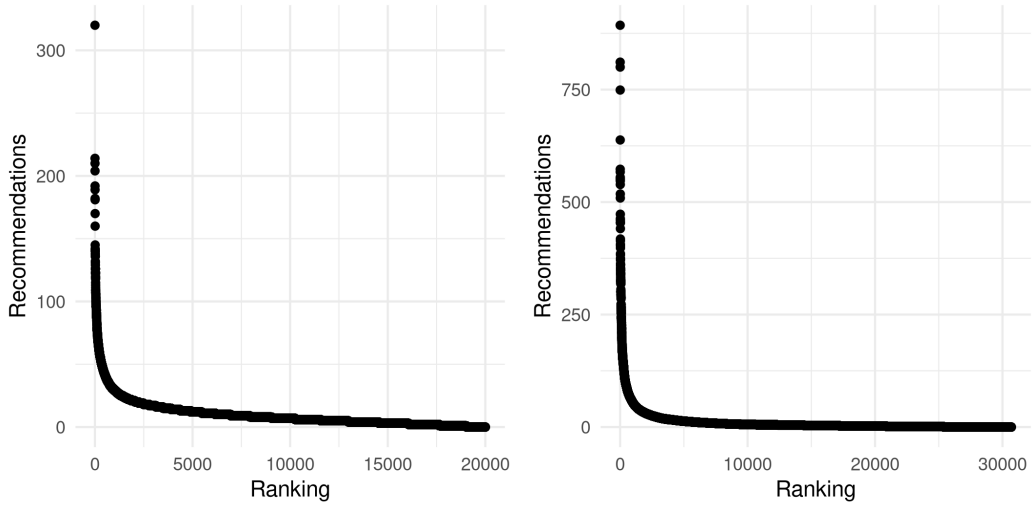


(a) *Vanilla model with artificial movie in red.* **(b)** *Model with short vector representations.*

Figure 3.7: *Recommendation profile for artificial movie (a) and short vector representations (b).*

The last two models were considered the confirmations of the hypothesis that (at least for this kind of recommendation systems) a subset of items was always much more recommended than the rest as long as the data was sparse. Figure 3.8a represents the same recommendation algorithm applied to another dataset, the Book-Crossing dataset. Figure 3.8b contains the results of the model applied to another set of random vector representations, this time with the probability of each element being non-zero respecting the marginal distributions of the vanilla dataset. Again, the exponential decay pattern persisted, only slightly less pronounced in the Book-Crossing case.

The analysis up until now has been static, that is, the recommendation model does not learn from the users' responses to its suggestions. There is no interaction with users



(a) *Recommendation profile for book dataset.*

(b) *Random simularion of vanilla.*

Figure 3.8: *Recommendation profile for book dataset (a) and random simulation of vanilla (b).*

and no opportunity to evolve over time. The next chapter addresses this point by using Google's newly released TensorFlow Recommenders library (*TensorFlow Recommenders* 2021) to gather data about what happens to a system's recommendations as users follow its suggestions. Employing a deep learning model that is able to improve over time is a significant departure from the content-based models presented here and, if a similar recommendation profile can also be detected for multi-criteria recommender systems on dynamic scenarios, then the hypothesis ventilated in the section above would become even more plausible.

Chapter 4

Dynamic Analysis

Following the static analysis, we moved on to the dynamic analysis. Understanding how the recommendation model responds to users reinforcing its internal biases, like the ones already detected, could potentially lead to a better understanding of how these systems favor certain kinds of content. We also were not able to find any published dataset that fit our experimental design.

In order for this analysis to be more true to reality, we implemented a simple recommendation algorithm using TensorFlow Recommenders (*TensorFlow Recommenders* 2021), a library for machine learning developed by Google for use with its TensorFlow () framework. This means that, even though our model is deliberately bare-bones, it conforms to industry-standard technology and practices.

The choice to use a simple recommendation algorithm instead of a more complex one was twofold: first, we did not want to use a model that could introduce many confounding parameters to the analysis (e.g. hyperparameters, hardware requirements, etc.), and second, we wanted to study a baseline that could, in the future, be used as a comparison point for more complex algorithms. Our algorithm is, therefore, a straightforward neural network made up of linear stacks of layers.

The goal of this step is to gather data on how the recommendation system behaves over time. As will be explained in the next sections, we aim to understand what happens to the recommendation profile of the algorithm as it interacts with itself via users that follow the generated suggestions.

The hypothesis is that the recommendation profile will grow ever more steep, which is a reasonable idea; if the users reinforce the beliefs of the algorithm, then is stands to reason that it will recommend popular movies with more and more frequently, to more and more users. How much more frequently, however, is the true question.

For the sake of clarity, let us imagine two users with very distinct preferences: Alice, who enjoys adventure movies, and Bob, who enjoys horror movies. In principle, the algorithm should have very different recommendations for both of them and, were they to follow them, their custom suggestions should grow increasingly different. At the end of this experiment, users like Alice would all be recommended the same movies, and users like Bob

would have their own set of very popular films; we should expect, therefore, a multimodal distribution of the recommendation frequencies, with "typical" adventure movies and "typical" horror movies being much more popular than comedy, for example.

However, if the final recommendation profile looked like what was showcased in the previous chapter, i.e. a very small subset of movies being recommended to most users, then we could infer that the system devolved into a degenerate feedback loop, ignoring personal preferences and distinctions between films.

4.1 Datasets

For the dynamic experiments, we kept using the Movielens dataset (Harper and Konstan, 2015). This time, however, we used the full "1M" dataset instead of sampling movies from the larger "25M" version. Given that we wanted our dynamic analysis to be conducted in a realistic scenario, we decided that it would be better not to change the data. This whole experiment will, therefore, use a version of the dataset commonly used for machine learning benchmarks with no alteration whatsoever.

The 1M dataset contains 1000209 ratings of almost 4000 movies made by over 6000 anonymous MovieLens users who joined the platform in 2000. In this particular version, each user has made at least 20 ratings. There are 4 columns available:

- `UserID`: Unique user identifier, ranging from 1 to 6040.
- `MovieID`: Unique movie identifier, ranging from 1 to 3952.
- `Rating`: Movie rating according to user, from 0 to 5 stars.
- `Timestamp`: When the user made the rating, in seconds since the epoch.

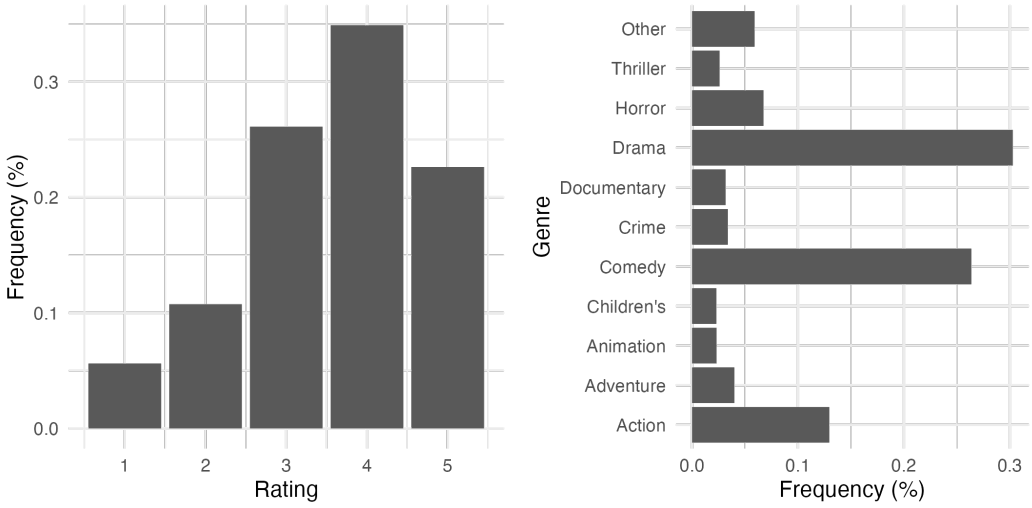
A second, auxiliary, dataset was also used to enrich the main one. "Movies" contains extra information about the movies in 1M, which allowed us to add more variables to the recommendation system. This new dataset has 3 columns:

- `MovieID`: Unique user identifier, ranging from 1 to 6040.
- `Title`: Title of the movie, as provided by IMDB.
- `Genres`: Pipe-separated string with all applicable genres.

Figure 4.1a showcases the distribution of ratings from the "1M" dataset and Figure 4.1b showcases the distribution of genres from the "Movies" dataset. Note that, for both this visualization and the rest of the chapter, the genre of each movie is taken to be the first one listed on the `Genres` column.

4.2 Experiments

The dynamic experiment starts in a manner much similar to the static experiment. The full MovieLens dataset is fed as training data to a recommendation system in order to get it ready for giving suggestions to users. As explained in the opening section of this chapter,



(a) *Rating distribution.*

(b) *Genre distribution.*

Figure 4.1: *Distributions of ratings (a) and genres (b).*

we chose a simple algorithm, (i.e., without rich features or complex layer arrangements) in order to reduce the number of possible interferences architecture could have on our analysis.

The chosen recommendation algorithm was a basic ranking model described in using TensorFlow Recommenders (*TensorFlow Recommenders* 2021). It is composed of multiple stacked dense layers and uses regularized mean squared error as its loss function in order to avoid overfitting. The main class in the model is reproduced below, and the full algorithm is listed in Appendix ??.

```
class MovielensModel(tfrs.models.Model):

  def __init__(self):
    super().__init__()
    self.ranking_model: tf.keras.Model = RankingModel()
    self.task: tf.keras.layers.Layer = tfrs.tasks.Ranking(
      loss = tf.keras.losses.MeanSquaredError(),
      metrics=[tf.keras.metrics.RootMeanSquaredError()]
    )

  def call(self, features: Dict[str, tf.Tensor]) -> tf.Tensor:
    return self.ranking_model(
        (features["user_id"], features["movie_title"]))

  def compute_loss(self, features: Dict[Text, tf.Tensor],
                   training=False) -> tf.Tensor:
    labels = features.pop("user_rating")

    rating_predictions = self(features)
```

```
    # The task computes the loss and the metrics.
    return self.task(labels=labels, predictions=rating_predictions)
```

In the first step of the experiment, we trained the recommendation model using Movielens' 1M ratings dataset, which we will refer to as `ratings0` from now on. Evaluating our model, called `model0`, on test data yielded a root mean squared error (RMSE) of 0.92; this result is similar to TFRS' deep & cross network () results when trained on the same data. Once `model0` was ready for making recommendations, we applied it to every possible user-movie pairing, generating a complete matrix of predicted ratings called `predictions0`. This process corresponds to the top half of Figure 4.2.

In an environment like YouTube's recommendations sidebar, the user is presented with a few items that the algorithm thinks they would like, and then they can either ignore the sidebar or select one of the options to watch. In this work, we assume that the user never ignores their recommendations because it would increase the complexity of the simulation exponentially; we, therefore, always picked one movie at random from each user's 10 best-ranked entries.

This set of well-ranked movies was our way of simulating thousands of users simultaneously approving of the algorithms recommendations and selecting one option from their sidebars. The last step involved removing the oldest rating of each each user from `ratings0` and appending these these selections to the dataset in order to create `ratings1`. The full data flow is illustrated in Figure 4.2.

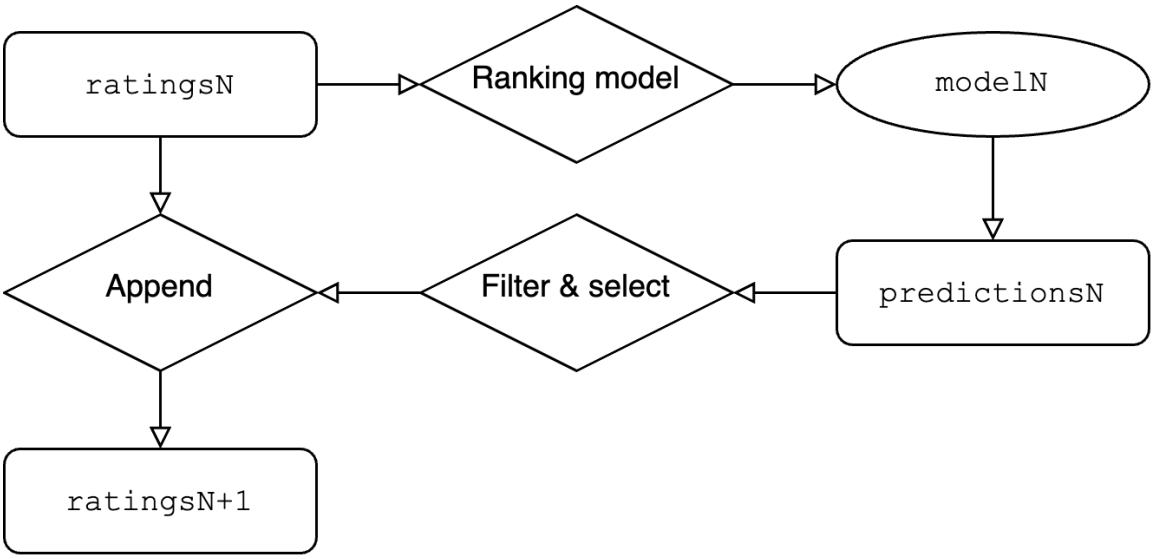


Figure 4.2: *Data flow diagram.*

A feedback loop, however, can't be created from a single iteration. For this reason, the process just described was classified as the "zeroth" iteration in the process (as it involved `ratings0`). The "Nth" iteration began with `ratingsN`, trained `modelN`, generated `predictionsN`, and ended with `ratingsN+1`. We repeated this process until we got to `ratings4`, totaling 5 ratings datasets (one original and 4 derived through ranking models). The full code for this data flow is listed in Appendix **??**.

With these new datasets we were able to analyze the differences between distinct

generations of models and understand exactly how the positive feedback loop influenced the last iteration.

Using these datasets as sources, we found that, with each iteration, the recommendation profile got steeper and steeper, that is, a few popular items got more recommended while the rest fell into disfavor; this was predictably more noticeable in movies with higher average ratings. In fact, a small set of around 20 movies were the only ones that consistently rose in popularity with new iterations.

Figure 4.3 has five subplots which represent the recommendation profile of each iteration of the recommendation system. For `ratings0`, it's possible to see that a number of well-rated movies were more popular, i.e., were rated by more people. Once we generate the first batch of recommendations, we add up the number of users each movie was recommended to; this is seen in the second plot, `ratings1`. The outlier points to the left clearly indicate that a small subset of movies strayed from the pack and were recommended to more people than had watched them previously.

This process repeats itself until, in `ratings4`, the most popular movie is recommended more than 2000 times, while the most popular movie in `ratings0` was watched a little over 500 times.

As explained before, we expected that movies which where already popular would be recommended more times, but these plots indicate a powerful feedback loop. Examining the data, we saw that the algorithm was consistently recommending movies which the users had already watched and this process only became more accentuated with each subsequent iteration. This is in line with recommendations from large social networks like YouTube, which do recommend videos that were already watched by the user.

Table 4.1 contains the most popular movies on each generation of the experiment. Just as in Chapter 3, the popularity of the movies in generation 0 (i.e. the original dataset) is taken to be the number of reviews each movie received. It is remarkable that the most recommended movies in generation 4 were neither the most popular nor the highest rated.

Original data	
Title	Popularity
Sixth Sense, The (1999)	256
Who Framed Roger Rabbit? (1988)	244
X-Men (2000)	210
Gladiator (2000)	208
Shakespeare in Love (1998)	202
American Beauty (1999)	198
Talented Mr. Ripley, The (1999)	197
Fargo (1996)	175
Men in Black (1997)	174
Star Wars: Episode I - The Phantom Menace (1999)	165

Generation 1

Title	Popularity
Usual Suspects, The (1995)	715
Shawshank Redemption, The (1994)	705
Godfather, The (1972)	669
Close Shave, A (1995)	664
Schindler's List (1993)	663
Seven Samurai (The Magnificent Seven) (Shichinin no samurai) (1954)	652
Rear Window (1954)	644
Wrong Trousers, The (1993)	629
Sunset Blvd. (a.k.a. Sunset Boulevard) (1950)	601
Raiders of the Lost Ark (1981)	520

Generation 2

Title	Popularity
Shawshank Redemption, The (1994)	1296
Usual Suspects, The (1995)	1292
Seven Samurai (The Magnificent Seven) (Shichinin no samurai) (1954)	1276
Godfather, The (1972)	1261
Close Shave, A (1995)	1252
Rear Window (1954)	1252
Wrong Trousers, The (1993)	1217
Sunset Blvd. (a.k.a. Sunset Boulevard) (1950)	1191
Star Wars: Episode IV - A New Hope (1977)	772
Schindler's List (1993)	657

Generation 3

Title	Popularity
Shawshank Redemption, The (1994)	1908
Usual Suspects, The (1995)	1898
Seven Samurai (The Magnificent Seven) (Shichinin no samurai) (1954)	1881
Godfather, The (1972)	1861
Close Shave, A (1995)	1810
Wrong Trousers, The (1993)	1808
Sunset Blvd. (a.k.a. Sunset Boulevard) (1950)	1786
Schindler's List (1993)	1256
Rear Window (1954)	1243
Third Man, The (1949)	1172

Generation 4

Title	Popularity
Seven Samurai (The Magnificent Seven) (Shichinin no samurai) (1954)	2496
Shawshank Redemption, The (1994)	2468

Close Shave, A (1995)	2436
Wrong Trousers, The (1993)	2432
Godfather, The (1972)	2422
Sunset Blvd. (a.k.a. Sunset Boulevard) (1950)	2385
Usual Suspects, The (1995)	1888
Schindler's List (1993)	1822
Rear Window (1954)	1814
Raiders of the Lost Ark (1981)	1718

Table 4.1: *Top 10 most popular movies at each generation of the algorithm.*

A good way to measure how diverse were the recommendations made by the algorithm is to calculate the entropy () of each set. This metric is defined for a discrete random variable X, which can assume values form the alphabet χ, through the formula:

$$H(X) = -\sum_{x \in \chi} p(x) \log p(x).$$

In the extreme, if the same movie is recommended to every user, than the entropy of the recommendations will tend to zero. In absolute terms, the entropy of `ratings0` was 7.42, and the entropy of `ratings4` was 6.08 (82% lower). A comparison in relative terms is displayed in Figure 4.4.

We can also observe this reduction of variety in an even more visual way. In Figure 4.5 we plotted movie popularity over time; each line represents a movie and each step in the x-axis is a new generation of the model. Figure 4.5a makes it very clear that only 13 movies rose in popularity over the four generations of the recommendation system, becoming orders of magnitude more popular than the rest. We also scaled the y-axis by taking the its log in Figure 4.5b, and we can see that, in fact, no other movie rose in popularity besides the ones that stick out after `model2`.

It is also of note that the few movies that rise in popularity are not the most popular ones from `ratings0`, even though genre was the only metadata we fed into the system. This uncovers a significant feature of the feedback loop we observed in Figure 4.3: the items that the algorithm amplifies don't necessarily have to be the most mainstream, or in other words, recommendation systems are able to boost content artificially.

4.3 Modeling

Deep learning models are extremely powerful and versatile; they are not, however, very explainable (). Recommendation algorithms are often described as black boxes because of the seemingly inscrutable impact that different inputs have on the output. The term "neural hallucination" () has even been coined to explain the process by which neural networks infer missing information form incomplete data.

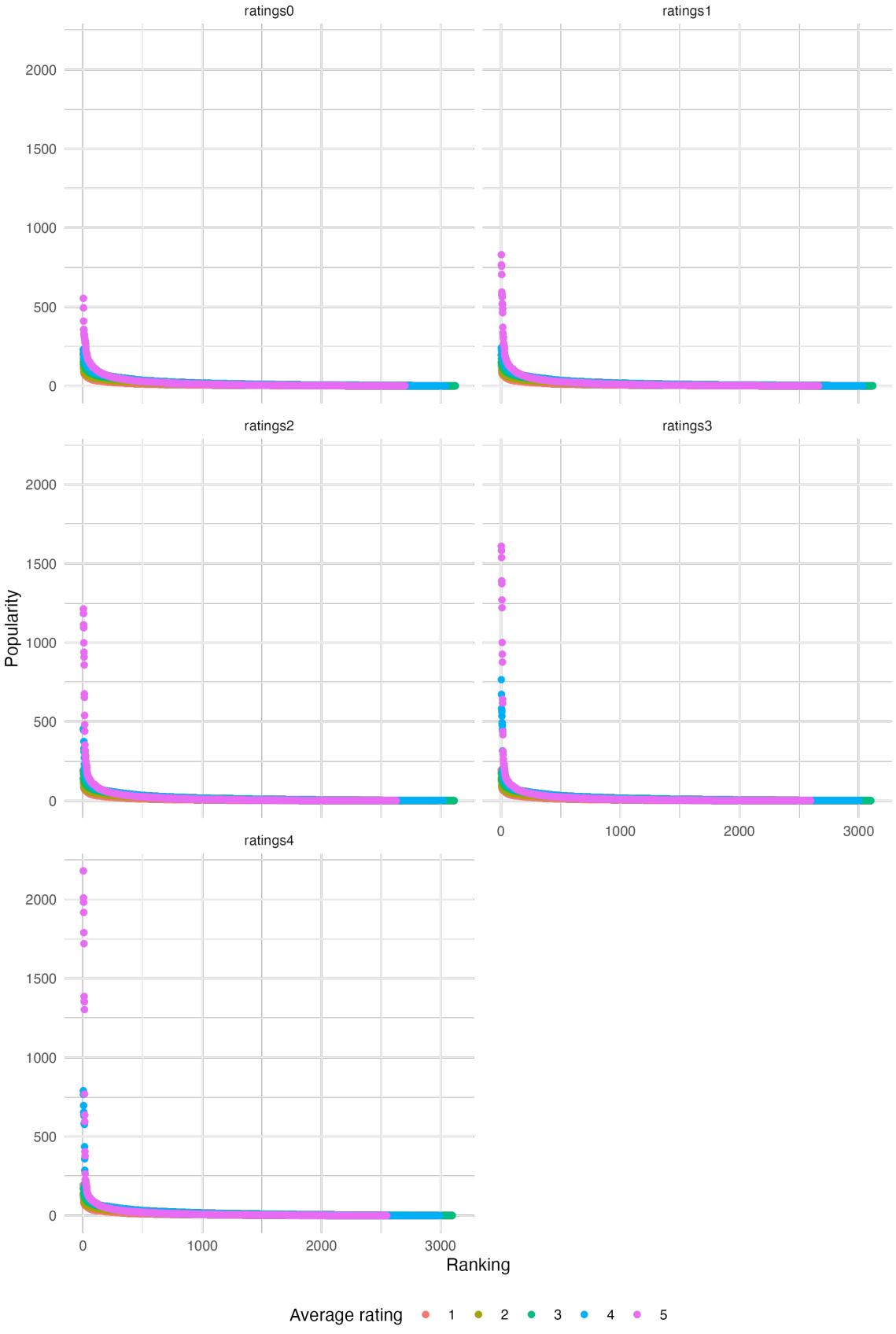


Figure 4.3: *Recommendation profile of every generation. Colors indicate average movie rating, which is a strong predictor of popularity over time.*

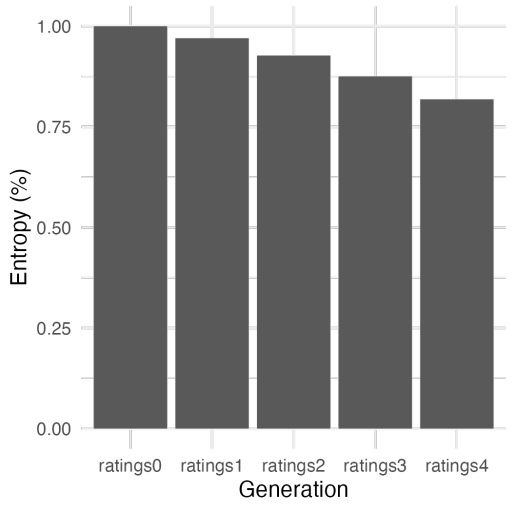
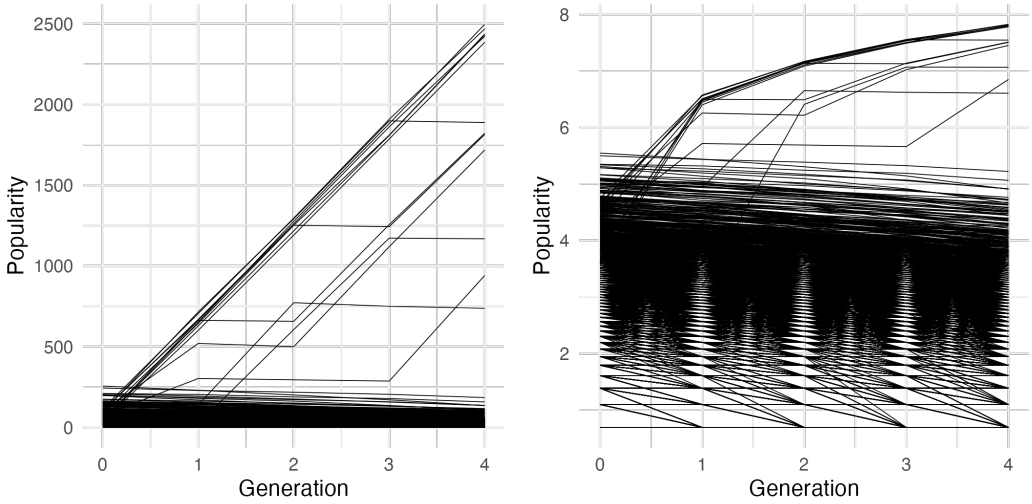


Figure 4.4: *Recommendation entropy as a percentage of ratings0s entropy.*



(a) *Movie popularity over time.*

(b) *Movie log-popularity over time.*

Figure 4.5: *Popularity of every movie from ratings0 to ratings4.*

Since we did identify a feedback loop in the recommendation system, our next goal was to fit a regression model on the generated data. This could help clarify how the algorithm had made it's recommendations; if we were able to create a regression model that replicated the behavior of the system, then we could make inferences about the algorithm based on the characteristics of said model.

As is common in computational inference (), our process involved multiple rounds of modeling. We started with a very simple model and, according to goodness-of-fit measurements, made it more and more complex in order to achieve a better representation of the behavior of the recommendation system. As we will see later, even after feeding all of the the available data (and some extra metadata that the recommender algorithm didn't have access to), our models were not able to capture such a degenerate distribution.

For every model described below, the response variable Y is the number of recommendations a movie received at a specific moment in time, while λ is the average recommendation rate (conditioned to a movie's genre and rating) at a specific moment in time.

4.3.1 Poisson Regression

From the quasi-exponential decay observed in Figure 4.3, we hypothesized that the logarithm of popularity was a function of the linear combination of genre, generation and rating. Our first attempt was, therefore, a simple Poisson regression (). For this kind of model, we fit a regression of the form

$$\log(\lambda_i) = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_k x_{ik},$$

where λ_i is the Poisson parameter, x_{ik} is the regressor variable, and β_i is the regression coefficient. To implement the regression, we used R's `glm()` () function with popularity being modeled by the factor crossing of generation (i.e., the iteration from 0 to 4), genre and average rating.

As the reader might be able to see, we used the genre variable in our regression model even though we didn't feed it to the recommendation system. As explained before, the algorithm only had access to the popularity and ratings of the movies, but it is possible the there exists a latent effect of genre on the other variables; since a machine learning system is much more flexible than a regression model, we opted to explicitly feed the genre into the Poisson model and all others that followed.

`glm()` and all other regression functions return the coefficients of the regression alongside their calculated significance. However, analyzing the coefficients by themselves is only one part of the full picture. proposed the usage of simulated envelopes, alongside coefficient significance, to assess global goodness-of-fit.

This method is such that, under the correct model, the plot for the observed data is likely to fall within the envelope. The objective is not to provide a region of acceptance, but some sort of guidance to what kind of shape to expect.

Obtaining the simulated envelope consists of fitting a model; extracting model diagnostics and calculating sorted absolute values; simulating the desired number of response

variables using the same model matrix, error distribution and fitted parameters; fitting the same model to each simulated response variable and obtaining the same model diagnostics, again sorted absolute values; and computing the desired percentiles (in this case, 2.5% and 97.5%) at each value of the expected order statistic to form the envelope. In our case, the expected order statistic was obtained through the following normal distribution:

$$\Phi^{-1}(\frac{i+3/8}{n+1/4})$$

For illustrative purposes, we will present the full output of the Poisson model below. It is notable how many coefficients are considered highly significant, meaning that, individually, they were able to capture the feedback loop generated by the recommendation system.

```
Call:
glm(formula = pop ~ t * genre * rating, family = "poisson", data = features)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-33.145   -3.088   -1.222    1.260   93.001

Coefficients:
                          Estimate Std. Error z value Pr(>|z|)
(Intercept)               9.210e-02  9.514e-02   0.968 0.333031
t                        -6.844e-02  4.200e-02  -1.629 0.103208
genreAnimation           -2.276e+00  1.475e-01 -15.430  < 2e-16 ***
genrechildren's           1.489e-02  1.754e-01   0.085 0.932347
genreComedy              -1.904e-01  1.041e-01  -1.828 0.067539 .
genreCrime               -4.898e+00  1.858e-01 -26.364  < 2e-16 ***
genreDocumentary         -1.801e+00  4.491e-01  -4.010 6.07e-05 ***
genreDrama               -1.035e+00  1.140e-01  -9.086  < 2e-16 ***
genreFilm-Noir           -1.631e+01  5.403e-01 -30.185  < 2e-16 ***
genreHorror               3.012e-01  1.178e-01   2.556 0.010582 *
genreMusical             -2.784e+00  4.813e-01  -5.784 7.31e-09 ***
genreMystery             -4.849e+00  2.464e-01 -19.681  < 2e-16 ***
genreRomance             -1.476e+00  5.299e-01  -2.785 0.005347 **
genreSci-Fi               1.583e+00  1.898e-01   8.341  < 2e-16 ***
genreThriller             3.854e-01  1.791e-01   2.152 0.031394 *
genreWestern             -4.253e+00  5.017e-01  -8.476  < 2e-16 ***
rating                    8.877e-01  2.740e-02  32.395  < 2e-16 ***
t:genreAnimation         -3.139e+00  6.223e-02 -50.435  < 2e-16 ***
t:genrechildren's        -5.212e-04  7.760e-02  -0.007 0.994641
t:genreComedy            -2.235e-02  4.602e-02  -0.486 0.627215
t:genreCrime             -3.561e+00  8.570e-02 -41.557  < 2e-16 ***
t:genreDocumentary        2.463e-01  1.933e-01   1.274 0.202583
t:genreDrama             -1.845e+00  5.015e-02 -36.799  < 2e-16 ***
t:genreFilm-Noir         -5.419e+00  1.944e-01 -27.872  < 2e-16 ***
```

```
t:genreHorror               7.776e-03  5.217e-02   0.149 0.881511
t:genreMusical              1.190e-01  2.140e-01   0.556 0.578236
t:genreMystery             -3.676e+00  1.032e-01 -35.632  < 2e-16 ***
t:genreRomance             -8.909e-02  2.334e-01  -0.382 0.702658
t:genreSci-Fi              -2.251e+00  8.076e-02 -27.869  < 2e-16 ***
t:genreThriller             6.362e-02  7.894e-02   0.806 0.420240
t:genreWestern              5.290e-02  2.211e-01   0.239 0.810874
t:rating                   -1.584e-02  1.212e-02  -1.307 0.191193
genreAnimation:rating       7.049e-01  3.938e-02  17.900  < 2e-16 ***
genrechildren's:rating     -4.814e-02  5.303e-02  -0.908 0.364006
genreComedy:rating          1.711e-02  2.989e-02   0.572 0.567055
genreCrime:rating           1.261e+00  4.818e-02  26.161  < 2e-16 ***
genreDocumentary:rating     5.638e-02  1.160e-01   0.486 0.626776
genreDrama:rating           1.163e-01  3.199e-02   3.635 0.000278 ***
genreFilm-Noir:rating       3.865e+00  1.239e-01  31.197  < 2e-16 ***
genreHorror:rating         -1.456e-01  3.541e-02  -4.111 3.95e-05 ***
genreMusical:rating         6.395e-01  1.255e-01   5.095 3.50e-07 ***
genreMystery:rating         1.286e+00  6.114e-02  21.030  < 2e-16 ***
genreRomance:rating         3.492e-02  1.506e-01   0.232 0.816639
genreSci-Fi:rating         -5.037e-01  5.457e-02  -9.230  < 2e-16 ***
genreThriller:rating       -1.952e-01  5.094e-02  -3.833 0.000127 ***
genreWestern:rating         9.773e-01  1.309e-01   7.467 8.19e-14 ***
t:genreAnimation:rating     8.263e-01  1.631e-02  50.671  < 2e-16 ***
t:genrechildren's:rating   -1.370e-03  2.350e-02  -0.058 0.953503
t:genreComedy:rating        6.194e-03  1.323e-02   0.468 0.639610
t:genreCrime:rating         9.129e-01  2.163e-02  42.204  < 2e-16 ***
t:genreDocumentary:rating  -5.843e-02  5.002e-02  -1.168 0.242813
t:genreDrama:rating         5.028e-01  1.399e-02  35.927  < 2e-16 ***
t:genreFilm-Noir:rating     1.279e+00  4.402e-02  29.061  < 2e-16 ***
t:genreHorror:rating       -6.526e-03  1.573e-02  -0.415 0.678237
t:genreMusical:rating      -3.530e-02  5.588e-02  -0.632 0.527603
t:genreMystery:rating       9.553e-01  2.495e-02  38.291  < 2e-16 ***
t:genreRomance:rating       3.023e-02  6.624e-02   0.456 0.648140
t:genreSci-Fi:rating        6.707e-01  2.195e-02  30.551  < 2e-16 ***
t:genreThriller:rating     -1.870e-02  2.252e-02  -0.830 0.406351
t:genreWestern:rating      -1.209e-02  5.768e-02  -0.210 0.834031
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 562028  on 13559  degrees of freedom
Residual deviance: 268737  on 13500  degrees of freedom
AIC: 318813

Number of Fisher Scoring iterations: 6
```

When we look at the simulated envelope of this regression, however, it is plain to see that the model does not conform to the delimited region and, therefore, isn't able to properly capture the variability of the data. The plot generated with the `hnp` () R package can be seen in Figure 4.6.

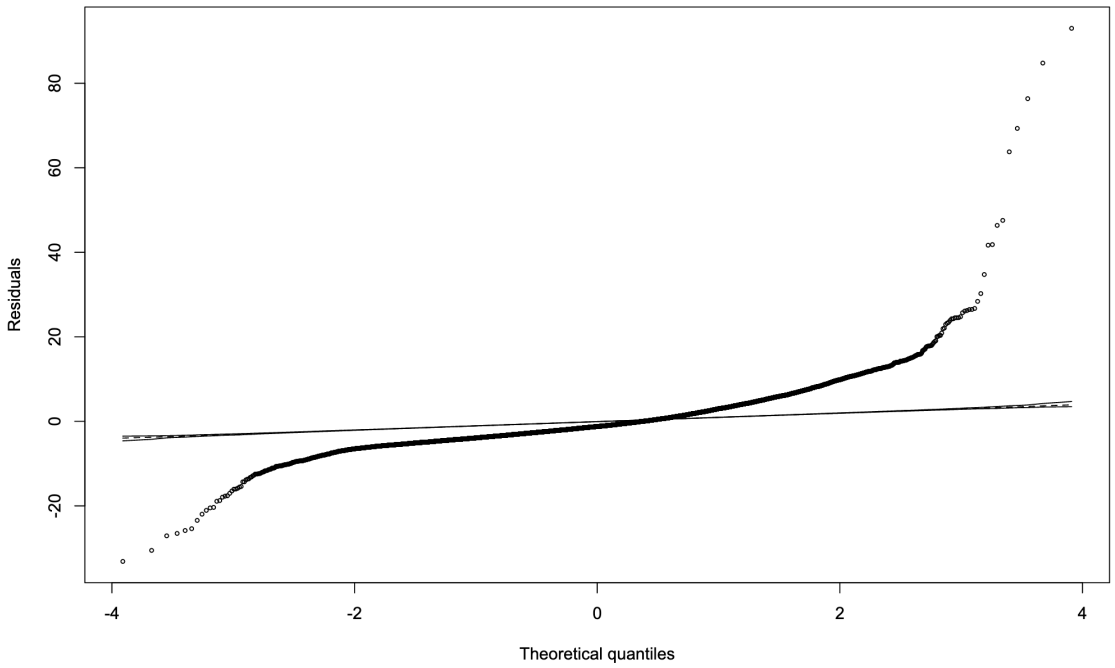


Figure 4.6: *Residual analysis via simulated envelope for the Poisson regression.*

4.3.2 Negative Binomial Regression

Our next attempt involved a generalization of the previous regression. In a Poisson model, both mean and variance are assumed to be the same; this could be limiting our previous attempt, since it cannot capture overdispersed data. The negative binomial distribution is a generalization of the Poisson distribution () that is able to better model data under these new assumptions. For this kind of model, we fit a regression of the form

$$\log(\lambda_i) = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_k x_{ik},$$

where λ_i depends on covariates, as does x_{ik}. More concretely, we used MASS's () `glm.nb()` function with the same formula as last time.

In this instance, the significancy of the coefficients of the regression (omitted) was even higher than of the Poisson model. However, upon calculating the simulated envelope, as seen in Figure 4.7, it became clear that this model, while better than the last, still was not adequately capturing the data's variability.

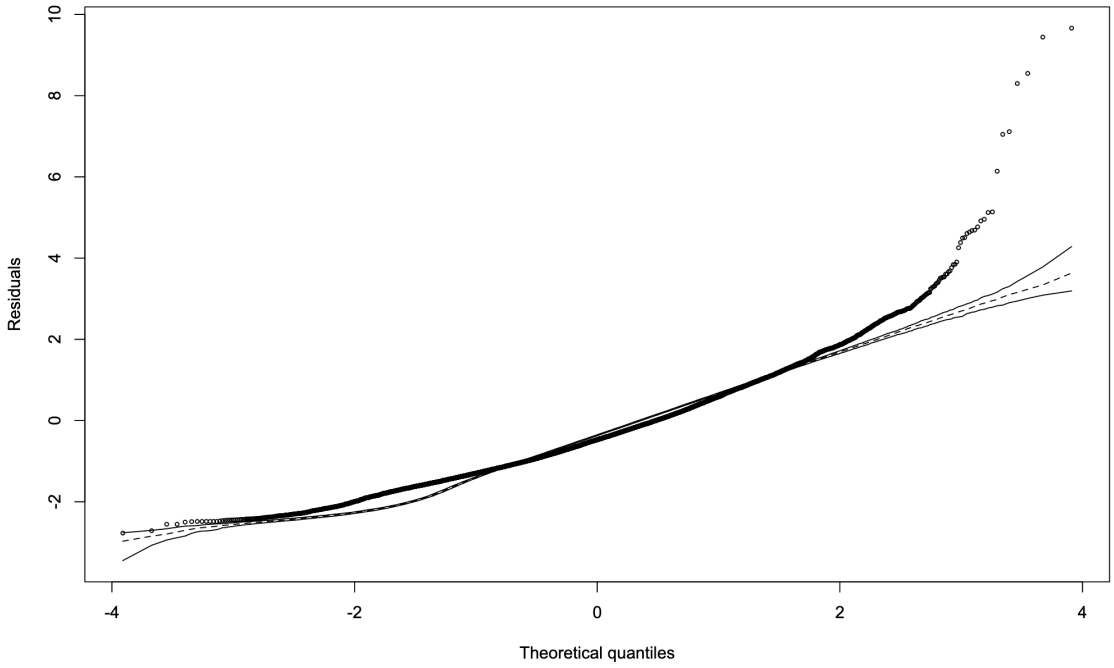


Figure 4.7: *Residual analysis via simulated envelope for the negative binomial regression.*

4.3.3 Mixed-effects Poisson Regression

Since both previous models failed to accurately capture the algorithm's recommendation profile, we moved on to mixed-effects models given their usefulness in longitudinal studies (). With this kind of model we would be able to vary the coefficients from movie to movie, allowing some movies to have steeper recommendation profiles than others.

The Poisson mixed-effects model can be generalized from the standard model by adding normally distributed random-effects to the usual fixed-effects already described:

$$\log(\lambda_i) = \delta + \beta_0 + \beta_1 x_{i1} + \cdots + \beta_k x_{ik},$$

where $\delta \sim \mathcal{N}(0, \psi)$ allows us to better model longitudinal observations within the same individual (which are not independent of each other) and serially correlated errors. In our regression, we added mixed-effects to the `movie_id` variable using the `glmmTMB` () R package.

As with our previous regressions, we evaluate the fit of the model not only through the significancy of the coefficients (which were on par with the other models), but also through the simulated envelope. Since we used a different R package to run the mixed-effects models, we also had to resort to another simulation package: `DHARMa` (); this means that the new plots in Figure 4.8 follow a different aesthetic convention and displays more information by default.

Note how the fit is much better than the fixed-effects models, but the outliers on the

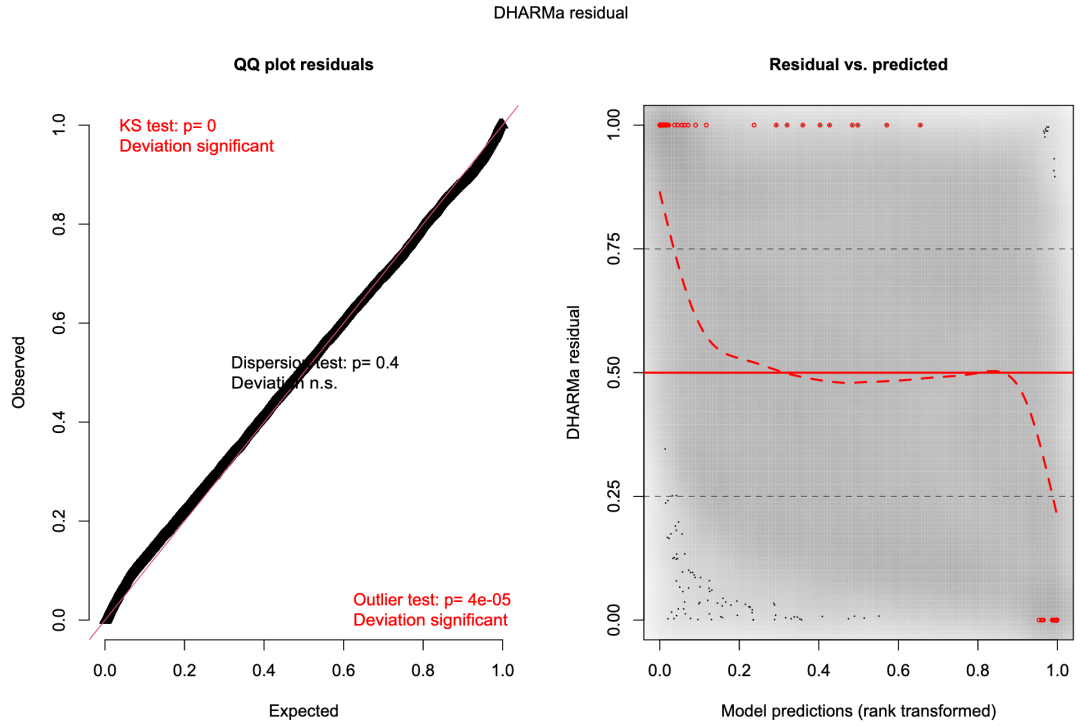


Figure 4.8: *Residual analysis via simulated envelope for the mixed-effects Poisson regression.*

left and right sides of the distribution are still present as attested by the outlier deviation test.

4.3.4 Mixed-effects Negative Binomial Regression

Our last attempt at capturing the recommendation profile of the recommender system involved using a negative binomial mixed-model regression in order to help with the overdispersion seen above. Similar to the Poisson mixed-effects model, this regression also extends the default formula of its fixed-effects counterpart:

$$\log(\lambda_i) = \delta + \beta_0 + \beta_1 x_{i1} + \cdots + \beta_k x_{ik}.$$

In the table below, the reader can see the full summary of the negative binomial mixed model. Figure 4.9 showcases the simulated envelope for the same model.

```
  Family: nbinom2  ( log )
Formula:          pop ~ t * genre * rating + (1 | movie_id)
Data: features

    AIC      BIC   logLik deviance df.resid
83863.8  84370.3 -41865.9  83731.8    15844

Random effects:
```

```
Conditional model:
Groups   Name        Variance Std.Dev.
movie_id (Intercept) 1.364    1.168
Number of obs: 15910, groups:  movie_id, 3182

Dispersion parameter for nbinom2 family ():   49

Conditional model:
                           Estimate Std. Error z value Pr(>|z|)
(Intercept)                -0.108714   0.272464  -0.399 0.689891
t                          -0.210272   0.021153  -9.941  < 2e-16 ***
genreAdventure              0.112129   0.574197   0.195 0.845174
genreAnimation             -2.286362   0.773873  -2.954 0.003132 **
genrechildren's             0.855196   0.755919   1.131 0.257915
genreComedy                -0.108159   0.352431  -0.307 0.758925
genreCrime                 -3.590470   0.850227  -4.223 2.41e-05 ***
genreDocumentary           -1.659325   1.508329  -1.100 0.271285
genreDrama                 -1.219969   0.409312  -2.981 0.002877 **
genreFilm-Noir            -16.633103   2.371128  -7.015 2.30e-12 ***
genreHorror                 0.512807   0.443025   1.158 0.247062
genreMusical               -4.259252   2.088483  -2.039 0.041410 *
genreMystery               -4.646920   1.461852  -3.179 0.001479 **
genreRomance               -2.118789   1.856289  -1.141 0.253699
genreSci-Fi                 0.363380   1.044431   0.348 0.727899
genreThriller               1.382908   0.855392   1.617 0.105944
genreWestern               -3.766758   2.108848  -1.786 0.074072 .
rating                      0.957533   0.085468  11.203  < 2e-16 ***
t:genreAdventure            0.161327   0.053403   3.021 0.002520 **
t:genreAnimation           -0.392012   0.067698  -5.791 7.01e-09 ***
t:genrechildren's           0.116028   0.066946   1.733 0.083068 .
t:genreComedy               0.119502   0.030504   3.918 8.94e-05 ***
t:genreCrime               -0.146821   0.085503  -1.717 0.085952 .
t:genreDocumentary          0.329357   0.195776   1.682 0.092507 .
t:genreDrama                0.005393   0.038739   0.139 0.889287
t:genreFilm-Noir           -0.730742   0.227275  -3.215 0.001303 **
t:genreHorror               0.148203   0.041813   3.544 0.000393 ***
t:genreMusical              0.265118   0.243060   1.091 0.275383
t:genreMystery             -1.150785   0.123034  -9.353  < 2e-16 ***
t:genreRomance              0.064921   0.227735   0.285 0.775590
t:genreSci-Fi              -0.311108   0.079779  -3.900 9.63e-05 ***
t:genreThriller             0.134758   0.077077   1.748 0.080403 .
t:genreWestern              0.184582   0.216581   0.852 0.394073
t:rating                    0.029594   0.006273   4.717 2.39e-06 ***
genreAdventure:rating      -0.209253   0.179840  -1.164 0.244606
genreAnimation:rating       0.594327   0.226301   2.626 0.008633 **
genrechildren's:rating     -0.473591   0.247951  -1.910 0.056131 .
```

```
genreComedy:rating          -0.194909   0.109223  -1.785 0.074341 .
genreCrime:rating            0.736375   0.243050   3.030 0.002448 **
genreDocumentary:rating     -0.113489   0.399299  -0.284 0.776242
genreDrama:rating           -0.031020   0.120957  -0.256 0.797601
genreFilm-Noir:rating        3.958891   0.595316   6.650 2.93e-11 ***
genreHorror:rating          -0.422452   0.151685  -2.785 0.005352 **
genreMusical:rating          0.944897   0.569196   1.660 0.096903 .
genreMystery:rating          1.092607   0.409047   2.671 0.007560 **
genreRomance:rating          0.013712   0.548422   0.025 0.980053
genreSci-Fi:rating          -0.423339   0.324094  -1.306 0.191477
genreThriller:rating        -0.729129   0.256016  -2.848 0.004400 **
genreWestern:rating          0.725673   0.578412   1.255 0.209625
t:genreAdventure:rating     -0.053250   0.015828  -3.364 0.000768 ***
t:genreAnimation:rating      0.110189   0.018603   5.923 3.16e-09 ***
t:genrechildren's:rating    -0.039499   0.020922  -1.888 0.059031 .
t:genreComedy:rating        -0.039790   0.008913  -4.464 8.04e-06 ***
t:genreCrime:rating          0.038460   0.022654   1.698 0.089557 .
t:genreDocumentary:rating   -0.088874   0.050610  -1.756 0.079076 .
t:genreDrama:rating         -0.006800   0.010754  -0.632 0.527202
t:genreFilm-Noir:rating      0.183567   0.054187   3.388 0.000705 ***
t:genreHorror:rating        -0.052874   0.013574  -3.895 9.81e-05 ***
t:genreMusical:rating       -0.082105   0.063538  -1.292 0.196285
t:genreMystery:rating        0.335177   0.032553  10.296  < 2e-16 ***
t:genreRomance:rating       -0.018634   0.064877  -0.287 0.773939
t:genreSci-Fi:rating         0.108210   0.023442   4.616 3.91e-06 ***
t:genreThriller:rating      -0.044310   0.022584  -1.962 0.049758 *
t:genreWestern:rating       -0.055461   0.056878  -0.975 0.329521
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

In comparison to the Poisson mixed model, the negative binomial mixed model is 2.5 times less dispersed and the outlier deviation is 4 times lower. Still, this was not enough to capture accurately the recommendation profile of our system for the more popular movies (left tail of the rightmost plot).

At this point, it is safe to assume that this distribution tends towards degeneracy, meaning it approaches a situation () where only one movie is recommended every single time and all others are never recommended.

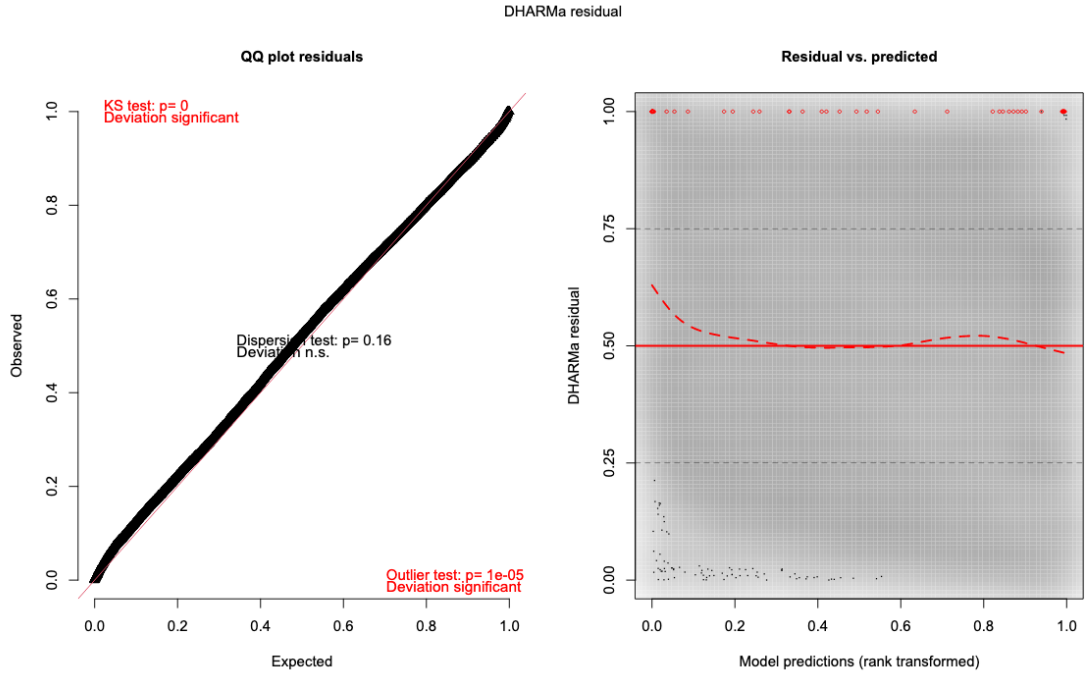


Figure 4.9: *Residual analysis via simulated envelope for the mixed-effects negative binomial regression.*

Chapter 5

Conclusion

In this work, we studied recommender system bias. As detailed in Chapter 1, these algorithms are pervasive in modern live and have significant impact on our information diet; however, a growing body of evidence indicates that these systems, specially when deployed in large social networks, can display significant biases in their recommendations. Researchers and activists alike worry that these biases might create echo chambers and radicalization pipelines by amplifying extremist content. In broad terms, our goal was to study one way through which these algorithms might be boosting fringe viewpoints and radicalizing users: degenerate feedback loops.

Degeneracy in recommender systems has been explored before by Nguyen *et al.* (2014) and Jiang *et al.* (2019). They concluded that these algorithms show a tendency to reduce the diversity of recommended content over time because of the feedback loop that occurs when the system must learn from the its own outputs. Our main research objective was to further characterize and model this behavior using both qualitative and quantitative methods.

In Chapter 2, we reviewed the scientific literature that deals with this topic and presented a few different viewpoints on the matter. Works by Hosseinmardi *et al.* (2020), Huszár *et al.* (2021), and others suggest that social networks tend to amplify far-right views, while Munger and Phillips (2020) and Ledwich and Zaitsev (2019) posit that this is not the case. M. H. Ribeiro *et al.* (2020) found evidence that, in general, YouTube users tend to migrate from "lighter" content towards more extreme videos, and Stoica, Riederer, *et al.* (2018) coined the term “algorithmic glass ceiling” to describe recommender system’s propensity to reinforce homophilic behavior.

We also drew attention to non-scientific endeavors that attempt to better understand filter bubbles, echo chambers, and their impacts on the public. While this issue is being debated in academia, journalists and whistleblowers like P. V. Ribeiro (2021) or Wong (2021) hold social media companies accountable by gathering anecdotal evidence and personal accounts about algorithmic bias.

Our main contributions come in Chapters 3 and 4, where we describe and conduct multiple experiments on recommender algorithms.

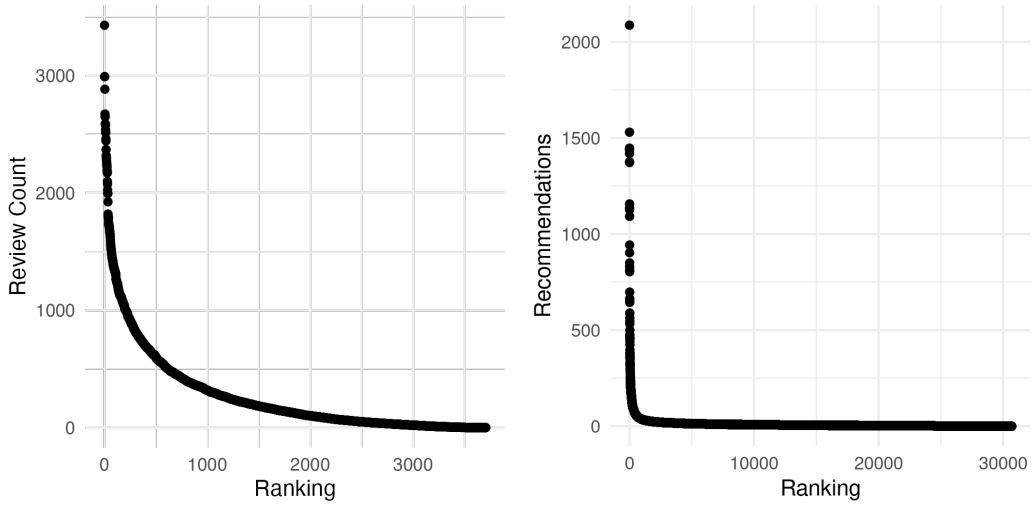
5.1 Static Analysis

Chapter 3 started with a question: can recommender systems send users in radicalization spirals? Given the many limitations of social media analysis, we argue that the best way to answer this question is to explore the simplest and most bare-bones algorithm possible. Our hypothesis is that, if such a simple algorithm displays a feedback loop dynamic, then there might be some intrinsic property of this type of system that pushes recommendations towards degeneracy.

We designed multiple experiments with the goal of analyzing the recommendation profile of our algorithm of choice: a simple, content-based recommender that used cosine similarity as its internal metric. Recommendation profile is a concept we developed as a shorthand for an algorithm's characteristic tendency to recommend more frequently a smaller (or larger) subset of the original data.

Our expectation was that a systems's recommendation profile would be approximately the same as the original dataset's item popularity distribution; users that enjoyed less popular movies would be recommended other unpopular items and users that enjoyed the more popular movies would likewise be recommended other prominent items. This, however, was not the case.

In the original MovieLens (Harper and Konstan, 2015) dataset, the popularity distribution was sub-exponential, meaning that there was not a sharp decrease in the number of reviews from the most popular movies to the least popular ones. This was in stark contrast to our vanilla model's exponential recommendation profile. Figure 5.1 displays the two profiles side-by-side for ease of reference.



(a) *Popularity distribution.* **(b)** *Recommendation profile.*

Figure 5.1: *Comparison between the original dataset's review profile (a) and the vanilla algorithm's recommendation profile.*

We experimented on many variations to the vanilla model and of the original dataset, but all of them displayed exponential or super-exponential recommendation profiles. This led us to posit that, indeed, recommender systems might be prone to artificially amplifying

some contents independently of the original data. However, this static analysis was only part of the story and, in order to confirm this hypothesis, we needed to take user dynamics into account.

5.2 Dynamic Analysis

We followed up the static experiments with a dynamic analysis. The main question that motivated Chapter 4 was whether the amplification pipeline detected in Chapter 3 would be even further emphasized by the interaction between user and recommendation system.

Even though we used a bare-bones algorithm again, this time we went with Google's own machine learning library: TensorFlow Recommenders (*TensorFlow Recommenders* 2021). Our goal was to capture the dynamics that arise when a machine learning system has to learn from its own data, i.e., the engagement of a user with items recommended by the algorithm itself. Since we needed to generalize our conclusions to the environment of a social network, the library Google uses to build its recommendation engines was the ideal choice.

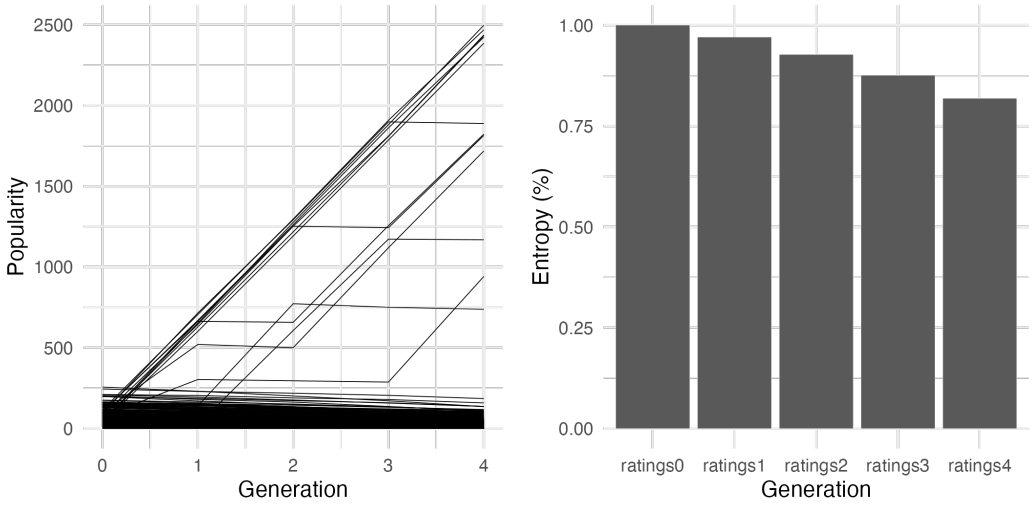
Using the same MovieLens (Harper and Konstan, 2015) dataset, we created a loop where the system would recommend 10 movies to a simulated user, this user would then chose one movie at random to "watch", and this interaction would be fed back into the model as a new input. This process was repeated 5 times for each user in the original dataset.

Just as with the static experiments, the recommendation profile of our algorithm got steeper and steeper over time, resulting in a similar super-exponential decay; a diminishing set of movies was being recommended to a growing number of users (as can be seen in Figure 5.2). Interestingly, the movies that grew in popularity with time were not the most popular ones in the original dataset.

In order to better understand the progression of the recommendation profile, we tried to model this behavior using well-understood statistical distributions. If we were able to accurately reproduce the results we obtained from the machine learning algorithm with a white-box model, we would gain a deeper insight into the internal mechanisms that caused the amplification pipeline we were detecting.

We attempted to model the pipeline with Poisson and negative binomial regressions, but had no success in both cases. We also tried adding mixed-effects to these models and were better able to capture the variance of the phenomenon, but a simulated envelope analysis revealed that our attempts were still falling short.

As with the numbers obtained by in a different scenario, the recommendation profiles of our model were tending quickly towards a degenerate distribution. We believe that this is yet another piece of evidence that suggests that recommender systems, if left unchecked, tent towards a confinement dynamic where users' feeds devolve into filter bubbles and where the points of view of highly-engaged minorities are amplified by a system that has to learn from itself.



(a) *Movie popularity over time.* **(b)** *Recommendation entropy over time.*

Figure 5.2: *Amplification pipeline detected on the dynamic experiment.*

5.3 Final Remarks

We began this work with the goal of developing a better understanding of the mechanisms through which recommendation algorithms could be radicalizing users, creating echo chambers and boosting fringe viewpoints. We believe we have found enough evidence to support the hypothesis that recommender systems are able to create amplification pipelines from degenerate feedback loops.

However, our findings are but one step in the direction of a solid theory. Further research is still needed in order to find an unambiguous causal between these processes and, more importantly, how to fix this behavior. A possible next step would involve applying the methods discussed in this work to more complex recommendation systems and to larger real-world datasets. Another possible improvement would be to simulate users that ignore their recommendations.

References

[Agarwal and Sureka 2015] Swati Agarwal and Ashish Sureka. "Topic-Specific YouTube Crawling to Detect Online Radicalization". In: *Databases in Networked Information Systems*. Ed. by Wanming Chu, Shinji Kikuchi, and Subhash Bhalla. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2015, pp. 133–151. ISBN: 978-3-319-16313-0. DOI: 10.1007/978-3-319-16313-0_10 (cit. on p. 7).

[Alfano *et al.* 2020] Mark Alfano, Amir Ebrahimi Fard, J. Adam Carter, Peter Clutton, and Colin Klein. "Technologically scaffolded atypical cognition: the case of YouTube's recommender system". In: *Synthese* (June 9, 2020). ISSN: 1573-0964. DOI: 10.1007/s11229-020-02724-x. URL: https://doi.org/10.1007/s11229-020-02724-x (visited on 12/02/2020) (cit. on p. 8).

[Banik 2017] Rounak Banik. *The Movies Dataset*. Nov. 10, 2017. URL: https://kaggle.com/rounakbanik/the-movies-dataset (visited on 03/01/2021) (cit. on p. 13).

[Bobadilla *et al.* 2013] J. Bobadilla, F. Ortega, A. Hernando, and A. Gutiérrez. "Recommender systems survey". In: *Knowledge-Based Systems* 46 (July 1, 2013), pp. 109–132. ISSN: 0950-7051. DOI: 10.1016/j.knosys.2013.03.012. URL: http://www.sciencedirect.com/science/article/pii/S0950705113001044 (visited on 10/29/2020) (cit. on pp. 3, 7).

[Burke 2010] Robin Burke. "Evaluating the dynamic properties of recommendation algorithms". In: *Proceedings of the fourth ACM conference on Recommender systems*. RecSys '10. New York, NY, USA: Association for Computing Machinery, Sept. 26, 2010, pp. 225–228. ISBN: 978-1-60558-906-0. DOI: 10.1145/1864708.1864753. URL: https://doi.org/10.1145/1864708.1864753 (visited on 10/29/2020) (cit. on p. 9).

[Caton and Haas 2020] Simon Caton and Christian Haas. "Fairness in Machine Learning: A Survey". In: *arXiv:2010.04053 [cs, stat]* (Oct. 4, 2020). arXiv: 2010.04053. URL: http://arxiv.org/abs/2010.04053 (visited on 06/20/2021) (cit. on p. 9).

[Cho *et al.* 2020] Jaeho Cho, Saifuddin Ahmed, Martin Hilbert, Billy Liu, and Jonathan Luu. "Do Search Algorithms Endanger Democracy? An Experimental Investigation of Algorithm Effects on Political Polarization". In: *Journal of Broadcasting & Electronic Media* 64.2 (May 1, 2020). Publisher: Routledge _eprint: https://doi.org/10.1080/08838151.2020.1757365, pp. 150–172. ISSN: 0883-8151. DOI: 10.1080/08838151.2020.1757365. URL: https://doi.org/10.1080/08838151.2020.1757365 (visited on 12/02/2020) (cit. on p. 8).

[Covington *et al.* 2016] Paul Covington, Jay Adams, and Emre Sargin. "Deep Neural Networks for YouTube Recommendations". In: *Proceedings of the 10th ACM Conference on Recommender Systems*. RecSys '16. New York, NY, USA: Association for Computing Machinery, Sept. 7, 2016, pp. 191–198. ISBN: 978-1-4503-4035-9. DOI: 10.1145/2959100.2959190. URL: https://doi.org/10.1145/2959100.2959190 (visited on 11/08/2020) (cit. on pp. 5, 7).

[Dash *et al.* 2019] Abhisek Dash, Animesh Mukherjee, and Saptarshi Ghosh. "A Network-centric Framework for Auditing Recommendation Systems". In: *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*. IEEE INFOCOM 2019 - IEEE Conference on Computer Communications. ISSN: 2641-9874. Apr. 2019, pp. 1990–1998. DOI: 10.1109/INFOCOM.2019.8737486 (cit. on p. 9).

[Faddoul *et al.* 2020] Marc Faddoul, Guillaume Chaslot, and Hany Farid. "A Longitudinal Analysis of YouTube's Promotion of Conspiracy Videos". In: *arXiv:2003.03318 [cs]* (Mar. 6, 2020). arXiv: 2003.03318. URL: http://arxiv.org/abs/2003.03318 (visited on 12/02/2020) (cit. on p. 8).

[Giller 2012] Graham L. Giller. *The Statistical Properties of Random Bitstreams and the Sampling Distribution of Cosine Similarity*. SSRN Scholarly Paper ID 2167044. Rochester, NY: Social Science Research Network, Oct. 25, 2012. DOI: 10.2139/ssrn.2167044. URL: https://papers.ssrn.com/abstract=2167044 (visited on 10/29/2020) (cit. on p. 9).

[Goldberg *et al.* 1992] David Goldberg, David Nichols, Brian M. Oki, and Douglas Terry. "Using collaborative filtering to weave an information tapestry". In: *Communications of the ACM* 35.12 (Dec. 1, 1992), pp. 61–70. ISSN: 0001-0782. DOI: 10.1145/138859.138867. URL: https://doi.org/10.1145/138859.138867 (visited on 02/28/2021) (cit. on p. 2).

[Guy *et al.* 2010] Ido Guy, Naama Zwerdling, Inbal Ronen, David Carmel, and Erel Uziel. "Social media recommendation based on people and tags". In: *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*. SIGIR '10. New York, NY, USA: Association for Computing Machinery, July 19, 2010, pp. 194–201. ISBN: 978-1-4503-0153-4. DOI: 10.1145/1835449.1835484. URL: https://doi.org/10.1145/1835449.1835484 (visited on 10/29/2020) (cit. on p. 7).

[Harper and Konstan 2015] F. Maxwell Harper and Joseph A. Konstan. "The MovieLens Datasets: History and Context". In: *ACM Transactions on Interactive Intelligent Systems* 5.4 (Dec. 22, 2015), 19:1–19:19. ISSN: 2160-6455. DOI: 10.1145/2827872. URL: https://doi.org/10.1145/2827872 (visited on 02/19/2021) (cit. on pp. 13, 22, 40, 41).

[He *et al.* 2016] Chen He, Denis Parra, and Katrien Verbert. "Interactive recommender systems: A survey of the state of the art and future research challenges and opportunities". In: *Expert Systems with Applications* 56 (Sept. 1, 2016), pp. 9–27. ISSN: 0957-4174. DOI: 10.1016/j.eswa.2016.02.013. URL: http://www.sciencedirect.com/science/article/pii/S0957417416300367 (visited on 10/29/2020) (cit. on p. 7).

[Hosseinmardi *et al.* 2020] Homa Hosseinmardi *et al.* "Evaluating the scale, growth, and origins of right-wing echo chambers on YouTube". In: *arXiv:2011.12843 [cs]* (Nov. 25, 2020). arXiv: 2011.12843. URL: http://arxiv.org/abs/2011.12843 (visited on 11/30/2020) (cit. on pp. 9, 39).

[Huszár *et al.* 2021] Ferenc Huszár *et al.* "Algorithmic Amplification of Politics on Twitter". In: *arXiv:2110.11010 [cs]* (Oct. 21, 2021). arXiv: 2110.11010. URL: http://arxiv.org/abs/2110.11010 (visited on 11/09/2021) (cit. on pp. 10, 39).

[Jiang *et al.* 2019] Ray Jiang, Silvia Chiappa, Tor Lattimore, András György, and Pushmeet Kohli. "Degenerate feedback loops in recommender systems". In: *AIES 2019 - Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society* (2019), pp. 383–390. ISSN: undefined. DOI: 10.1145/3306618.3314288. URL: https://www.mendeley.com/catalogue/c07bc3f3-281c-3ccb-b9b9-48a7fc7f2ece/?articleTrace=AAABsG80nAYfTD4HP10XPYWudyHkNafVnqCvn6Ig2oJqCCpBaS_wu6VTxyfAdMsg63hV_i76srILGk_K-cUnjMzRq5sM_flieDZo9zBeJAIulo-tCDBDZiQYD3Jhct1NDiXtVWxkivPLBiGtHYoWzRNabD2EjXxPO8EqBV5pBut6uzx23J9_1_QRQpa3fNNitIQHj3gF94ExSAlm6lVuHblzmuV-qpqRkgYPJpDNO-YfrtntvwAP3FRvgIsLQ8DL7-XY9fuXMxGeGIiTOkcbf3rc51ANKDonO9IKWvAJrUCoz6nKynOZYymZ1kv02PQGGXqFSeD_H4a4JkpyYSFqc48QLduCREyiXbme1KGk-_QCXMPu6y5-tfW_h-W28T_287qzhCx7UPZTvDK_nqvzK1rGt0_cZUz2BU34lBGHj6PZWG94rHR75PTtTSfEeDyagem8BxRApr-9q05eUHsdan8x_JPgxJjvEzTchxv958r6-dElNxQUNyBDwdUgxLQvLgUrfSPK8DhlqxWL7T2s4nC560IYd4Nt2toQ4-FoHbrFrfejqDZhGlLO8HX15Ti9Ue6-kp&dgcid=raven_md_suggest_email (visited on 04/20/2021) (cit. on pp. 10, 39).

[Kunaver and Požrl 2017] Matevž Kunaver and Tomaž Požrl. "Diversity in recommender systems - A survey". In: *Knowledge-Based Systems* 123 (May 1, 2017), pp. 154–162. ISSN: 0950-7051. DOI: 10.1016/j.knosys.2017.02.009. URL: http://www.sciencedirect.com/science/article/pii/S0950705117300680 (visited on 10/29/2020) (cit. on p. 7).

[Lecher and Yin 2022] Colin Lecher and Leon Yin. *One Year After the Capitol Riot, Americans Still See Two Very Different Facebooks - The Markup*. Section: Citizen Browser. URL: https://themarkup.org/citizen-browser/2022/01/06/one-year-after-the-capitol-riot-americans-still-see-two-very-different-facebooks (visited on 01/24/2022) (cit. on p. 11).

[Ledwich and Zaitsev 2019] Mark Ledwich and Anna Zaitsev. "Algorithmic Extremism: Examining YouTube's Rabbit Hole of Radicalization". In: *arXiv:1912.11211 [cs]* (Dec. 24, 2019). arXiv: 1912.11211. URL: http://arxiv.org/abs/1912.11211 (visited on 11/03/2020) (cit. on pp. 1, 10, 39).

[Mansoury *et al.* 2020] Masoud Mansoury, Himan Abdollahpouri, Mykola Pechenizkiy, Bamshad Mobasher, and Robin Burke. "Feedback Loop and Bias Amplification in Recommender Systems". In: *arXiv:2007.13019 [cs]* (July 25, 2020). arXiv: 2007.13019. URL: http://arxiv.org/abs/2007.13019 (visited on 06/20/2021) (cit. on p. 10).

[Matakos *et al.* 2020] A. Matakos, C. Aslay, E. Galbrun, and A. Gionis. "Maximizing the Diversity of Exposure in a Social Network". In: *IEEE Transactions on Knowledge and Data Engineering* (2020). Conference Name: IEEE Transactions on Knowledge and Data Engineering, pp. 1–1. ISSN: 1558-2191. DOI: 10.1109/TKDE.2020.3038711 (cit. on p. 8).

[Möller *et al.* 2018] Judith Möller, Damian Trilling, Natali Helberger, and Bram van Es. "Do not blame it on the algorithm: an empirical assessment of multiple recommender systems and their impact on content diversity". In: *Information, Communication & Society* 21.7 (July 3, 2018). Publisher: Routledge _eprint: https://doi.org/10.1080/1369118X.2018.1444076, pp. 959–977. ISSN: 1369-118X. DOI: 10.1080/1369118X.2018.1444076. URL: https://doi.org/10.1080/1369118X.2018.1444076 (visited on 02/05/2021) (cit. on p. 10).

[Munger and Phillips 2020] Kevin Munger and Joseph Phillips. "Right-Wing YouTube: A Supply and Demand Perspective". In: *The International Journal of Press/Politics* (Oct. 21, 2020). Publisher: SAGE Publications Inc, p. 1940161220964767. ISSN: 1940-1612. DOI: 10.1177/1940161220964767. URL: https://doi.org/10.1177/1940161220964767 (visited on 12/02/2020) (cit. on pp. 1, 4, 10, 39).

[Nguyen *et al.* 2014] Tien T. Nguyen, Pik-Mai Hui, F. Maxwell Harper, Loren Terveen, and Joseph A. Konstan. "Exploring the filter bubble: the effect of using recommender systems on content diversity". In: *Proceedings of the 23rd international conference on World wide web*. WWW '14. New York, NY, USA: Association for Computing Machinery, Apr. 7, 2014, pp. 677–686. ISBN: 978-1-4503-2744-2. DOI: 10.1145/2566486.2568012. URL: https://doi.org/10.1145/2566486.2568012 (visited on 10/29/2020) (cit. on p. 39).

[M. H. Ribeiro *et al.* 2020] Manoel Horta Ribeiro, Raphael Ottoni, Robert West, Virgílio A. F. Almeida, and Wagner Meira. "Auditing radicalization pathways on YouTube". In: *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*. FAT* '20. New York, NY, USA: Association for Computing Machinery, Jan. 27, 2020, pp. 131–141. ISBN: 978-1-4503-6936-7. DOI: 10.1145/3351095.3372879. URL: https://doi.org/10.1145/3351095.3372879 (visited on 10/29/2020) (cit. on pp. 4, 10, 13, 39).

[P. V. Ribeiro 2021] Paulo Victor Ribeiro. *Como a extrema direita burla punições do YouTube - e o Google finge que não vê*. The Intercept. Apr. 19, 2021. URL: https://theintercept.com/2021/04/19/como-a-extrema-direita-burla-punicoes-do-youtube-e-o-google-finge-que-nao-ve/ (visited on 08/09/2021) (cit. on pp. 11, 39).

[Ricci *et al.* 2011] Francesco Ricci, Lior Rokach, and Bracha Shapira. "Introduction to Recommender Systems Handbook". In: *Recommender Systems Handbook*. Ed. by Francesco Ricci, Lior Rokach, Bracha Shapira, and Paul B. Kantor. Boston, MA: Springer US, 2011, pp. 1–35. ISBN: 978-0-387-85820-3. DOI: 10.1007/978-0-387-85820-3_1. URL: https://doi.org/10.1007/978-0-387-85820-3_1 (visited on 02/28/2021) (cit. on pp. 3, 17).

[Roth *et al.* 2020] Camille Roth, Antoine Mazières, and Telmo Menezes. "Tubes and bubbles topological confinement of YouTube recommendations". In: *PLOS ONE* 15.4 (Apr. 21, 2020). Publisher: Public Library of Science, e0231703. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0231703. URL: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0231703 (visited on 01/11/2021) (cit. on p. 9).

[Sarwar *et al.* 2001] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. "Item-based collaborative filtering recommendation algorithms". In: *Proceedings of the 10th international conference on World Wide Web*. WWW '01. New York, NY, USA: Association for Computing Machinery, Apr. 1, 2001, pp. 285–295. ISBN: 978-1-58113-348-6. DOI: 10.1145/371920.372071. URL: https://doi.org/10.1145/371920.372071 (visited on 02/28/2021) (cit. on p. 16).

[Séraphin *et al.* 2017] Alava Séraphin, Frau-Meigs Divina, and Hassan Ghayda. *Youth and violent extremism on social media: mapping the research*. Google-Books-ID: PTRCDwAAQBAJ. UNESCO Publishing, Dec. 4, 2017. 167 pp. ISBN: 978-92-3-100245-8 (cit. on p. 4).

[Sinha *et al.* 2017] Ayan Sinha, David F. Gleich, and Karthik Ramani. "Deconvolving Feedback Loops in Recommender Systems". In: *arXiv:1703.01049 [cs]* (Mar. 3, 2017). arXiv: 1703.01049. URL: http://arxiv.org/abs/1703.01049 (visited on 11/08/2020) (cit. on p. 10).

[Sîrbu *et al.* 2019] Alina Sîrbu, Dino Pedreschi, Fosca Giannotti, and János Kertész. "Algorithmic bias amplifies opinion fragmentation and polarization: A bounded confidence model". In: *PLOS ONE* 14.3 (Mar. 5, 2019). Publisher: Public Library of Science, e0213246. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0213246. URL: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0213246 (visited on 10/29/2020) (cit. on p. 9).

[Stoica 2020] Ana-Andreea Stoica. "Algorithmic Fairness for Networked Algorithms". In: *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems*. AAMAS '20. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems, May 5, 2020, pp. 2214–2216. ISBN: 978-1-4503-7518-4. (Visited on 10/29/2020) (cit. on p. 8).

[Stoica and Chaintreau 2019] Ana-Andreea Stoica and Augustin Chaintreau. "Hegemony in Social Media and the effect of recommendations". In: *Companion Proceedings of The 2019 World Wide Web Conference*. WWW '19. New York, NY, USA: Association for Computing Machinery, May 13, 2019, pp. 575–580. ISBN: 978-1-4503-6675-5. DOI: 10.1145/3308560.3317589. URL: https://doi.org/10.1145/3308560.3317589 (visited on 10/29/2020) (cit. on p. 8).

[Stoica, Riederer, *et al.* 2018] Ana-Andreea Stoica, Christopher Riederer, and Augustin Chaintreau. "Algorithmic Glass Ceiling in Social Networks: The effects of social recommendations on network diversity". In: *Proceedings of the 2018 World Wide Web Conference*. WWW '18. Republic and Canton of Geneva, CHE: International World Wide Web Conferences Steering Committee, Apr. 23, 2018, pp. 923–932. ISBN: 978-1-4503-5639-8. DOI: 10.1145/3178876.3186140. URL: https://doi.org/10.1145/3178876.3186140 (visited on 10/29/2020) (cit. on pp. 5, 8, 14, 39).

[Su *et al.* 2016] Jessica Su, Aneesh Sharma, and Sharad Goel. "The Effect of Recommendations on Network Structure". In: *Proceedings of the 25th International Conference on World Wide Web*. WWW '16. Republic and Canton of Geneva, CHE: International World Wide Web Conferences Steering Committee, Apr. 11, 2016, pp. 1157–1167. ISBN: 978-1-4503-4143-1. DOI: 10.1145/2872427.2883040. URL: https://doi.org/10.1145/2872427.2883040 (visited on 10/29/2020) (cit. on p. 9).

[Tangherlini *et al.* 2020] Timothy R. Tangherlini, Shadi Shahsavari, Behnam Shahbazi, Ehsan Ebrahimzadeh, and Vwani Roychowdhury. "An automated pipeline for the discovery of conspiracy and conspiracy theory narrative frameworks: Bridgegate, Pizzagate and storytelling on the web". In: *PLOS ONE* 15.6 (June 16, 2020). Publisher: Public Library of Science, e0233879. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0233879. URL: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0233879 (visited on 10/29/2020) (cit. on p. 8).

[*TensorFlow Recommenders* 2021] *TensorFlow Recommenders*. TensorFlow. URL: https://www.tensorflow.org/recommenders (visited on 03/01/2021) (cit. on pp. 20, 21, 23, 41).

[Wong 2021] Julia Carrie Wong. "How Facebook let fake engagement distort global politics: a whistleblower's account". In: *The Guardian* (Apr. 12, 2021). ISSN: 0261-3077. URL: https://www.theguardian.com/technology/2021/apr/12/facebook-fake-engagement-whistleblower-sophie-zhang (visited on 08/09/2021) (cit. on pp. 11, 39).

[Yao *et al.* 2021] Sirui Yao *et al.* "Measuring Recommender System Effects with Simulated Users". In: *arXiv:2101.04526 [cs]* (Jan. 12, 2021). arXiv: 2101.04526. URL: http://arxiv.org/abs/2101.04526 (visited on 06/20/2021) (cit. on pp. 9, 13).

[Zhao *et al.* 2019] Zhe Zhao *et al.* "Recommending what video to watch next: a multi-task ranking system". In: *Proceedings of the 13th ACM Conference on Recommender Systems*. RecSys '19. New York, NY, USA: Association for Computing Machinery, Sept. 10, 2019, pp. 43–51. ISBN: 978-1-4503-6243-6. DOI: 10.1145/3298689.3346997. URL: https://doi.org/10.1145/3298689.3346997 (visited on 11/05/2020) (cit. on pp. 3, 7).

[Ziegler 2004] Cai-Nicolas Ziegler. *Book-Crossing Dataset*. Sept. 2004. URL: http://www2.informatik.uni-freiburg.de/~cziegler/BX/ (visited on 03/01/2021) (cit. on p. 14).