RecRec: Algorithmic Recourse for Recommender Systems

Sahil Verma
vsahil@cs.washington.edu
University of Washington
Seattle, WA, USA

Ashudeep Singh
ashudeepsingh@pinterest.com
Pinterest, Inc.
San Francisco, CA, USA

Varich Boonsanong
varicb@cs.washington.edu
University of Washington
Seattle, WA, USA

John P. Dickerson
john@cs.umd.edu
University of Maryland
College Park, MD, USA

Chirag Shah
chirags@uw.edu
University of Washington
Seattle, WA, USA

Abstract

Recommender systems play an essential role in the choices people make in domains such as entertainment, shopping, food, news, employment, and education. The machine learning models underlying these recommender systems are often enormously large and black-box in nature for users, content providers, and system developers alike. It is often crucial for all stakeholders to understand the model's rationale behind making certain predictions and recommendations. This is especially true for the content providers whose livelihoods depend on the recommender system. Drawing motivation from the practitioners' need, in this work, we propose a recourse framework for recommender systems, targeted towards the content providers. Algorithmic *recourse* in the recommendation setting is a set of actions that, if executed, would modify the recommendations (or ranking) of an item in the desired manner. A recourse suggests actions of the form: "if a feature changes X to Y, then the ranking of that item for a set of users will change to Z." Furthermore, we demonstrate that RecRec is highly effective in generating valid, sparse, and actionable recourses through an empirical evaluation of recommender systems trained on three real-world datasets. To the best of our knowledge, this work is the first to conceptualize and empirically test a generalized framework for generating recourses for recommender systems.

CCS Concepts

• **Information systems** → **Recommender systems**; • **Computing methodologies** → *Machine learning*.

Keywords

Algorithmic Recourse, Recommender Systems, Explainable Recommender Systems

ACM Reference Format:
Sahil Verma, Ashudeep Singh, Varich Boonsanong, John P. Dickerson, and Chirag Shah. 2023. RecRec: Algorithmic Recourse for Recommender Systems. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management (CIKM '23), October 21–25, 2023, Birmingham, United Kingdom.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3583780.3615181

CIKM '23, October 21–25, 2023, Birmingham, United Kingdom

ACM ISBN 979-8-4007-0124-5/23/10.
https://doi.org/10.1145/3583780.3615181

1 INTRODUCTION

Recommender systems are ubiquitous in online platforms today. They have a major influence on our choices in domains ranging from entertainment, social media, and shopping to news and education. These systems operate by filtering items from a large set to provide the most relevant ones to the user. Recommender systems can broadly be classified into two categories, content filtering and collaborative filtering [2, 8, 42, 49]. Content filtering represents each item using a set of features and recommends items to users based on the similarity to items consumed by the user in the past [3, 10, 31]. For example, if a user has purchased a cell phone recently, a content filtering system would recommend a phone case to the user based on its similarity to the phone. Collaborative filtering-based recommender systems recommend items to a user based on other users' interests who have a similar user history. For example, if a user recently bought a cell phone, a collaborative filtering-based recommender system would recommend the user with a phone case based on the information that other users who bought a cell phone also bought a phone case in the past.

Modern recommender systems are black-box models that has spurred the development of inherently interpretable recommender models or techniques to explain the factors influencing the recommendations [64]. We have also seen the rapid adoption of incorporating explainability in real-world recommender systems. Facebook offers "Why am I seeing this ad?" for every sponsored advertisement on its platform, and Amazon offers reasons why a product is recommended to you tab. These explanations are broadly termed as *feature attribution explanations* as they highlight a part of the features that lead to the recommendations.

All these explanations are primarily geared towards the end-users of the recommender platforms However, recommender systems are usually multi-stakeholder platforms with content providers and the system developers' interests baked into the system [1, 46, 65]. And these stakeholders also need transparency into the system. Since the content providers are dependent on the platform for their livelihood, they are interested in understanding the factors that influence their product's rank in the recommendations [9, 22, 24]. There have been several studies to understand the perspective of content providers offering services on several kinds of such platforms. Jhaver et al. [25] did a study with several Airbnb hosts to understand their perspectives. Rahman [37] interviewed freelancers

working on Upwork. Razaq et al. [38] interviewd sellers on handmade product platform Etsy. Most content providers expressed the helplessness they face in understanding the factors that influence their product's rank on the platform [16, 25, 37, 38], and would like to gain transparency into it (see Appendix A.2.1).

The kind of explainability that the content providers seek is similar to counterfactual explanations in classification systems [53, 55]. In general, counterfactual explanations describe a causal situation of the form: 'If I change X to Y, the outcome will change to Z'. Counterfactual explanations are frequently employed to answer questions like "What change in my features would help me to get the loan?". In recommender systems, a set of factors that improves the rank of a product offer a causal explanation of what changes would lead to an improved ranking for it. This set of factors are termed as *recourse* if the content providers can alter them to improve their product's rank. Since studies have shown that providing transparency into the algorithmic system improves the user's trust and adoption of the platforms [28], the motivating reason for incorporating counterfactual explanations in recommender systems is even more compelling.

The primary contribution of our work is to conceptualize a generalized framework for generating algorithmic recourse-based explanations for recommender systems. In this work, we use the terms counterfactual explanations and algorithmic recourse interchangeably, as we are using counterfactuals to provide recourses to the content providers, system developers, or curious end-users of a recommender system. Under this framework, we propose a novel algorithm, RecRec, that generates algorithmic recourses for a real-world recommender systems. RecRec casts the problem of finding recourses as an optimization problem that is solved using gradient descent in the feature space.

We first establish the desirable properties of a recourse in recommender systems (Section 2) and then cast the recourse generation problem as an optimization problem (Section 3), which RecRec solves. We conduct experiments with three different recommender systems and show RecRec's efficacy in generating recourses that achieve high success rate while satisfying the other desirable properties (Section 4). We make the following key contributions:

(1) We establish the desirable properties for algorithmic recourses in recommender systems (Section 2).
(2) We propose a novel approach called RecRec to generate recourses for a broad class of recommender system architectures (Section 3).
(3) We empirically demonstrate the effectiveness of RecRec through extensive experiments on three recommender systems trained on real-world datasets (Section 4).

2 DESIRABLE PROPERTIES OF ALGORITHMIC RECOURSE

To be effective for a content provider, a recourse in a recommender system setting should satisfy several desirable properties:

(1) *Valid:* A recourse when executed should lead to an increased exposure for the concerned item among the users of the target group, and therefore should have an improved rank after the recourse.
(2) *Sparse Changes:* A recourse should not change many features of the item. Being close to the original features makes the recourse more easily achievable [33].
(3) *Minimal side-effect* : A recourse should ideally only move the concerned item to an improved rank and have minimal side-effect on the ranks of the other items [43] (specially near the top ranks).

3 RecRec's ALGORITHM TO GENERATE RECOURSE

This section formulates the recourse from the perspective of a content provider wanting to change the features of an item so that it gets more exposure for users in a target group, i.e., within the top-k recommendations for these users.

Given an item's original features, r, the goal is to find updated features r', such that the item, *item*, is recommended within the *top-k* ranks for a group of target users In content filtering based systems, the features of an item are its attributes that are used to measure the similarity between different items, e.g. genres of a movie or book. Table 1 lists the notation used in this section.

Table 1: Notations for RecRec's Algorithm

Notation	Description
U:	Set of users
I:	Set of items
$\mathcal{I}$:	$\{v_i : i \in I, v_i \in \mathbb{R}^f\}$, v_i is the item features for item i.
$\mathcal{R}_j \in \mathbb{R}^{\lvert I_j \rvert}$:	$\mathcal{R}_j[k]$ denotes the rating given by user j to item $I_j[k]$
S:	Set of target users
a:	The item for which recourse is being sought

ALGORITHM 1: Generate recourse to move an item to the *top-k* recommendations for a target group of users.

Input : Item features $\mathcal{I}$, the target user group S, ratings given by each user $\mathcal{R}$, the concerned item a, the desired rank of the item (e.g. top-10), hyperparameter λ, hyperparameter LearningRate
Output: New item feature for concerned item a: v_a^*

Function Compute_Recourse($\mathcal{I}$, S, $\mathcal{R}$, a, DesiredRank, LearningRate, λ)
  iterations $\leftarrow 0$
  $v_a, v_a' \leftarrow \mathcal{I}[a]$
  $S' \leftarrow S$
  while iterations $<$ maxiterations **do**
    loss $\leftarrow -\left(\sum_{j \in S'} v_a \mathcal{I}_j^\top \odot \mathcal{R}_j\right) + \lambda * \left\|v_a' - v_a\right\|_1$
    // In each iteration, perform a gradient descent step for the loss
    $v_a' \leftarrow$ gradient_descent(LearningRate, loss)
    // Get the updated rank of the item for each user in S
    newrank $\leftarrow$ get_updated_ranks($\mathcal{I}$, v_a', $\mathcal{R}$, S) // Remove users from S for whom the concerned item is within the desired rank
    for $u \in S$ **do**
      if newrank$[u] \in$ DesiredRank **then**
        $S' \leftarrow S \setminus u$
  return v_a'
Function get_updated_ranks($\mathcal{I}$, v_a', $\mathcal{R}$, S)
  // Compute the new score of each item for each user
  for $u \in S$ **do**
    newscore $\leftarrow \emptyset$
    for $z \in \mathcal{I}$ **do**
      newscore$[z] \leftarrow \sum v_z \mathcal{I}_j^\top \odot \mathcal{R}_j$
    // sort the newscore in descending order to generate new ranks
    newrank$[u] \leftarrow \arg\text{sort}_{z \in \mathcal{I}}$ newscore$[z]$
  return newrank

Equation (1) shows the objective function that we need to optimize to generate a recourse that achieves the desired change in the rank of the item while not changing its feature too much.

$$v_a^* = \arg\max_{v_a'} \sum_{j \in S} v_a' \mathcal{I}_j^\top \odot \mathcal{R}_j \qquad s.t. \left\|v_a' - v_a\right\|_0 \leq \epsilon \tag{1}$$

We use gradient descent to optimize the objective function for which we need encode the twofold goal as a constrained optimization problem with sparsity inducing L1 norm as the constraint (with λ as the hyperparameter).

$$\arg\min_{v'_a} -\Big(\sum_{j \in S} v_a \mathcal{I}_j^\top \odot \mathcal{R}_j\Big) + \lambda * \left\|v'_a - v_a\right\|_1 \tag{2}$$

Algorithm 1 provides the algorithm to generate recourses for content filtering based recommender systems. The algorithm takes as input the features of all item $\mathcal{I}$, the target user group S, the ratings given by the users $\mathcal{R}$, the item for which a recourse is desired a, and the desired rank for the item $DesiredRank$. It runs the gradient descent algorithm until convergence. In each iteration, the algorithm computes the loss given in eq. (2) and updates the features of the item using the gradient descent algorithm. The first term of the loss function is only computed for the subset of users in the target group S for whom the concerned item has not yet been ranked within the desired rank (line 9). This helps in two ways: a) encourages the algorithm to change the item features to move the item within the desired rank for a larger number of users from the target group, and b) limits the change in the users' recommendation lists (which is another desirable property of a recourse).

Iterative Hard Thresholding We use L1 norm to induce sparsity in the change between the original and the recourse item features. However, this might not be sufficient to ensure that the recourse is sparse. Therefore, after the convergence of algorithm 1, we iteratively set the values of the features with the smallest absolute difference with the original features to the original feature value at those indices, a process termed as iterative hard thresholding [7]. This leads to a tradeoff between the number of users for whom the item is moved within the desired rank (*success rate*) and the sparsity of the recourse. We continue iteratively hard thresholding until we start losing more than a certain percentage of the success rate (in our experiments we set this threshold to 20%).

4 EVALUATION

We performed experiments using three real-world recommender systems to measure RecRec's efficacy, efficiency, and side-effect when generating recourses for items at various ranks:

(1) *MovieLens-100K [21]:* This movie recommendation dataset has about 1000 movies and 1700 users who gave a total of 100K ratings. Each movie has information such as its summary, actors, directors, and genre. We consider the movie summary as mutable and others as immutable features. Using standard NLP data processing we featurize each summary in about 9500 dimensions. Dot product between the feature vectors of two movies determine their similarity. If two movies are very similar and a user has liked one of them, the recommender system will recommend the other. MovieLens also provides the rating a user has given to certain movies which we use to weigh similarity between movies for providing more accurate recommendations. The dataset also provides user metadata, like age and occupation. In experiments, we use age ranges to group users into 5 equi-sized groups that content providers want to target.
(2) *AliEC Ads [50]:* This dataset is an ad click prediction dataset provided by Alibaba. It has about 5600 ads and 13200 users who interacted with 1.4 million ads. For each ad, it provides information like its category, brand, and price. All features are considered mutable. We one-hot encode all categorical features and use price after normalization to feature each ad in about 2300 dimensions. Again, the dot product between feature vectors of two ads determine their similarity. The dataset also provides user metadata such as age level and gender. In experiments, we use age level to group users into five equi-sized groups that content providers want to target.
(3) *Goodreads [56, 57]:* This book recommendation dataset has about 4300 books and 11200 users who have a total of 1.3 million ratings. Each book has features like its short description, genre, number of reviews, hardcover or ebook format. All features are considered mutable. Using standard NLP data processing, we featurize each movie description and other features in about 17400 dimensions. The dot product measures the similarity between two books, and user ratings are used to weigh similarity. The dataset does not provide user metadata, and therefore we group users into five equi-sized groups based on the number of ratings they provided.

4.1 Experimental Methodology

For all three recommender systems, we group the users into 5 equi-sized group either based on available metadata (like age) or based on the number of ratings they provided. Our experimental setup portrays a scenario where a content provider wants to increase the exposure of their item to a group of users, the *target group*. RecRec provides a recourse to the content provider and after its execution, if the rank of their item improves to being within the top-k recommendations for users in the target group, we would say that the item's exposure has increased and the provided recourse was valid. In experiments, we consider an item to have increased exposure for a user if after the recourse it is within the top-10 recommendation for that user. Even though the content providers would want to target a large group of users, optimizing the losses in eq. (2) for all users in the target group can be computationally expensive. Therefore, we experiment using a sampling procedure. We randomly sample a small percentage of the users in the target group and minimize the loss only over them. However, we report the percentage of users in the *entire target group* that get an increased exposure to the item. Intuitively, if we sample more users from the target group, a higher percentage of users would see the item within their top-10 recommendations.

Metrics. We report the following metrics for RecRec:

(1) *Success Rate:* For each recommender system, we report the percentage for users from the target group who see the concerned item withing their top-10 recommendations. We report this various sampling sizes. A higher value for this metric is better.
(2) *Number of changes required:* This metric is computed as the L0 norm of the difference between the original features and the new features after recourse. A lower value for this metric is better.
(3) *Side-effect on user recommendations:* To estimate the impact of a recourse on the users' original recommendations, we measure the similarity between the them and the new recommendations. We use rank-biased overlap (RBO) as a measure of similarity between the two ranked lists. Since the change in recommendations at top ranks matter more, we weigh them more when measuring RBO similarity (p = 0.5). A higher value for this metric is better.

All the metrics are reported for items at various original ranks. Specifically, we generate recourses for items whose original ranks

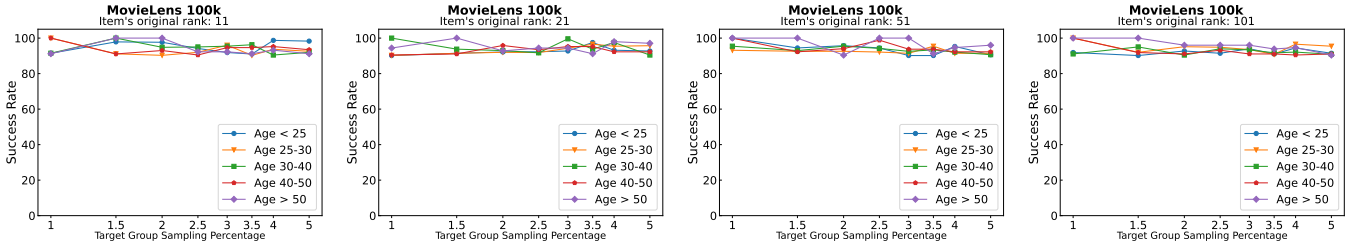

Figure 1: RecRec's success rate for items at original ranks 11, 21, 51, and 101 for the recommender system trained on MovieLens-100K. RecRec gets 100% success rate for all user groups and items at all original ranks with very small sampling size starting from 1% of the user group.

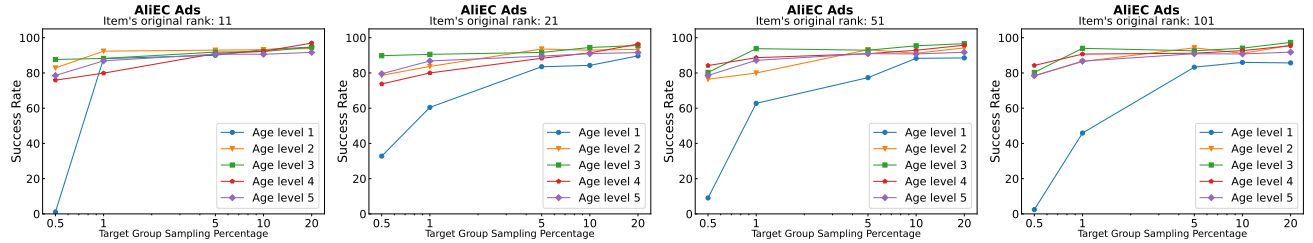

Figure 2: RecRec's success rate for items at original ranks 11, 21, 51, and 101 for the recommender system trained on AliEC Ads. RecRec gets more than 80% success for most user groups and items at all original ranks with very small sampling size starting from 1% of the user group, and increases to 100% for with 20% sampling.

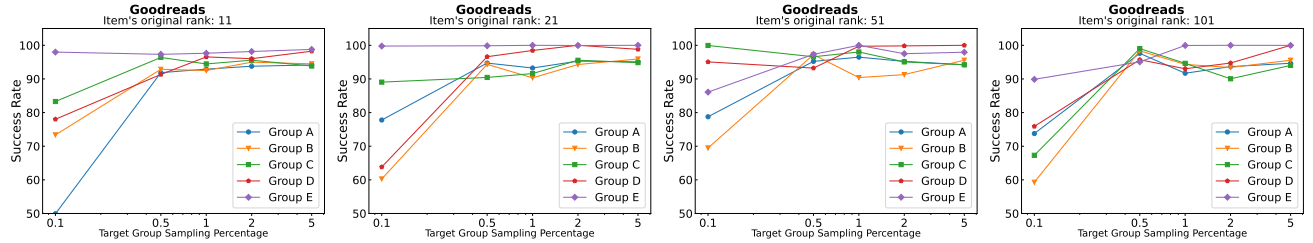

Figure 3: RecRec's success rate for items at original ranks 11, 21, 51, and 101 for the recommender system trained on Goodreads. RecRec gets more than 90% success for all user groups and items at all original ranks with very small sampling size starting from 0.5% of the user group, and increases to 100% for with 2-5% sampling.

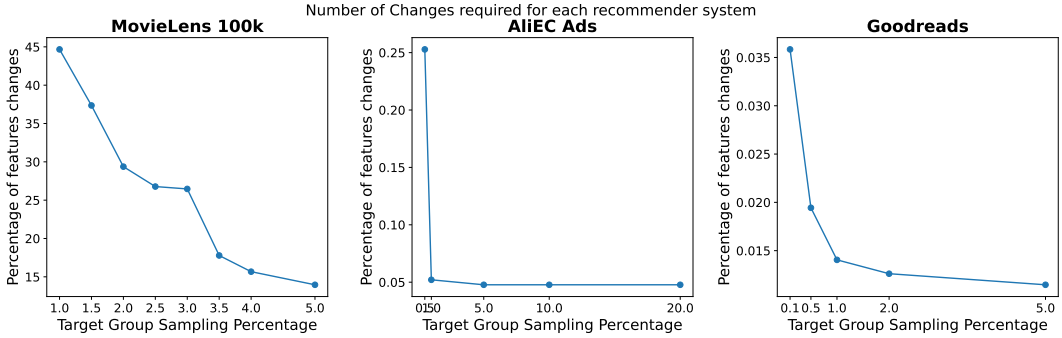

Figure 4: The percentage of item's features that need to be changed to execute the recourse. The plots are for the recommender systems trained on MovieLens-100K, AliEC Ads, and Goodreads. With increasing sampling percentage, the number of changes required to get a recourse decreases, and eventually becomes negligible.

are 11, 21, 51, and 101 (item rank averaged over the users in the target group). For rigorosity, we ensure that an item whose rank is already within top-10 for more than 1% of the users in the target group is not considered for getting a recourse.

Appendix A discusses the results from the experiments and Appendix B concludes the paper.

References

[1] Himan Abdollahpouri, Robin Burke, and Bamshad Mobasher. 2017. Recommender Systems as Multistakeholder Environments. In *Proceedings of the 25th Conference*

on User Modeling, Adaptation and Personalization (Bratislava, Slovakia) *(UMAP '17)*. Association for Computing Machinery, New York, NY, USA, 2 pages. https://doi.org/10.1145/3079628.3079657

[2] Gediminas Adomavicius and Alexander Tuzhilin. 2005. Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. *IEEE Transactions on Knowledge and Data Engineering* 17, 6 (jun 2005), 734–749.

[3] Charu C Aggarwal. 2016. Content-based recommender systems. In *Recommender systems*. Springer, 139–166.

[4] Muhammad Aurangzeb Ahmad, Ankur Teredesai, and Carly Eckert. 2018. Interpretable Machine Learning in Healthcare. In *2018 IEEE International Conference on Healthcare Informatics (ICHI)*. 447–447. https://doi.org/10.1109/ICHI.2018.00095

[5] Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. 2016. Concrete Problems in AI Safety. https://doi.org/10.48550/ARXIV.1606.06565

[6] Solon Barocas, Moritz Hardt, and Arvind Narayanan. 2019. *Fairness and Machine Learning*. fairmlbook.org. http://www.fairmlbook.org.

[7] Thomas Blumensath and Mike E. Davies. 2009. Iterative hard thresholding for compressed sensing. *Applied and Computational Harmonic Analysis* 27, 3 (2009), 265–274. https://www.sciencedirect.com/science/article/pii/S1063520309000384

[8] Jesús Bobadilla, Fernando Ortega, Antonio Hernando, and Abraham Gutiérrez. 2013. Recommender systems survey. *Knowledge-based systems* 46 (2013), 109–132.

[9] Eliane Léontine Bucher, Peter Kalum Schou, and Matthias Waldkirch. 2020. Pacifying the algorithm - Anticipatory compliance in the face of algorithmic management in the gig economy. *Organization* 28 (2020), 44 – 67.

[10] Laurent Candillier, Kris Jack, Françoise Fessant, and Frank Meyer. 2009. State-of-the-art recommender systems. In *Collaborative and Social Information Retrieval and Access: Techniques for Improved User Modeling*. IGI Global, 1–22.

[11] Nicholas Carlini and David A. Wagner. 2017. Towards Evaluating the Robustness of Neural Networks. *2017 IEEE Symposium on Security and Privacy (SP)* (2017), 39–57.

[12] Alexandra Chouldechova and Aaron Roth. 2020. A Snapshot of the Frontiers of Fairness in Machine Learning. *Commun. ACM* 63, 5 (apr 2020), 82–89. https://doi.org/10.1145/3376898

[13] Anupam Datta, Matt Fredrikson, Klas Leino, Kaiji Lu, Shayak Sen, and Zifan Wang. 2021. *Machine Learning Explainability and Robustness: Connected at the Hip*. Association for Computing Machinery, New York, NY, USA, 4035–4036. https://doi.org/10.1145/3447548.3470806

[14] Sarah Dean, Sarah Rich, and Benjamin Recht. 2020. Recommendations and User Agency: The Reachability of Collaboratively-Filtered Information. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency* (Barcelona, Spain) *(FAT* '20)*. Association for Computing Machinery, New York, NY, USA, 10 pages. https://doi.org/10.1145/3351095.3372866

[15] Finale Doshi-Velez and Been Kim. 2017. Towards A Rigorous Science of Interpretable Machine Learning. arXiv:1702.08608 [stat.ML]

[16] Motahhare Eslami, Aimee Rickman, Kristen Vaccaro, Amirhossein Aleyasen, Andy Vuong, Karrie Karahalios, Kevin Hamilton, and Christian Sandvig. 2015. "I Always Assumed That I Wasn't Really That Close to [Her]": Reasoning about Invisible Algorithms in News Feeds. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems* (Seoul, Republic of Korea) *(CHI '15)*. Association for Computing Machinery, New York, NY, USA, 10 pages. https://doi.org/10.1145/2702123.2702556

[17] Azin Ghazimatin, Oana Balalau, Rishiraj Saha Roy, and Gerhard Weikum. 2020. PRINCE: Provider-Side Interpretability with Counterfactual Explanations in Recommender Systems. In *Proceedings of the 13th International Conference on Web Search and Data Mining (WSDM '20)*. Association for Computing Machinery, New York, NY, USA, 9 pages. https://doi.org/10.1145/3336191.3371824

[18] Leilani H. Gilpin, David Bau, Ben Z. Yuan, Ayesha Bajwa, Michael Specter, and Lalana Kagal. 2018. Explaining Explanations: An Overview of Interpretability of Machine Learning. In *2018 IEEE 5th International Conference on Data Science and Advanced Analytics (DSAA)*. 80–89. https://doi.org/10.1109/DSAA.2018.00018

[19] Leif Hancox-Li. 2020. Robustness in Machine Learning Explanations: Does It Matter?. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency* (Barcelona, Spain) *(FAT* '20)*. Association for Computing Machinery, New York, NY, USA, 640–647. https://doi.org/10.1145/3351095.3372836

[20] Moritz Hardt, Eric Price, and Nathan Srebro. 2016. Equality of Opportunity in Supervised Learning. arXiv:1610.02413 [cs.LG]

[21] F. Maxwell Harper and Joseph A. Konstan. 2015. The MovieLens Datasets: History and Context. *ACM Trans. Interact. Intell. Syst.* 5, 4, Article 19 (Dec. 2015), 19 pages. https://doi.org/10.1145/2827872

[22] Maya Holikatti, Shagun Jhaver, and Neha Kumar. 2019. Learning to Airbnb by Engaging in Online Communities of Practice. *Proc. ACM Hum.-Comput. Interact.* 3, CSCW (2019), 19 pages. https://doi.org/10.1145/3359330

[23] Kenneth Holstein, Jennifer Wortman Vaughan, Hal Daumé, Miro Dudik, and Hanna Wallach. 2019. *Improving Fairness in Machine Learning Systems: What Do Industry Practitioners Need?* Association for Computing Machinery, New York, NY, USA, 1–16. https://doi.org/10.1145/3290605.3300830

[24] Mohammad Hossein Jarrahi and Will Sutherland. 2019. Algorithmic Management and Algorithmic Competencies: Understanding and Appropriating Algorithms in Gig Work. In *Information in Contemporary Society*. Springer International Publishing, Cham, 578–589.

[25] Shagun Jhaver, Yoni Karpfen, and Judd Antin. 2018. Algorithmic Anxiety and Coping Strategies of Airbnb Hosts. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (Montreal QC, Canada) *(CHI '18)*. Association for Computing Machinery, New York, NY, USA, 12 pages. https://doi.org/10.1145/3173574.3173995

[26] Vassilis Kaffes, Dimitris Sacharidis, and Giorgos Giannopoulos. 2021. Model-Agnostic Counterfactual Explanations of Recommendations. In *Proceedings of the 29th ACM Conference on User Modeling, Adaptation and Personalization (UMAP '21)*. Association for Computing Machinery, New York, NY, USA, 6 pages. https://doi.org/10.1145/3450613.3456846

[27] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. 2017. Adversarial Machine Learning at Scale. arXiv:1611.01236 [cs.CV]

[28] Min Kyung Lee. 2018. Understanding perception of algorithmic decisions: Fairness, trust, and emotion in response to algorithmic management. *Big Data & Society* 5 (2018).

[29] Min Kyung Lee, Daniel Kusbit, Evan Metsky, and Laura Dabbish. 2015. Working with Machines: The Impact of Algorithmic and Data-Driven Management on Human Workers. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems* (Seoul, Republic of Korea) *(CHI '15)*. Association for Computing Machinery, New York, NY, USA, 10 pages. https://doi.org/10.1145/2702123.2702548

[30] Zachary C. Lipton. 2018. The Mythos of Model Interpretability: In Machine Learning, the Concept of Interpretability is Both Important and Slippery. *Queue* 16, 3 (jun 2018), 31–57. https://doi.org/10.1145/3236386.3241340

[31] Pasquale Lops, Marco De Gemmis, and Giovanni Semeraro. 2011. Content-based recommender systems: State of the art and trends. *Recommender systems handbook* (2011), 73–105.

[32] Scott M. Lundberg and Su-In Lee. 2017. A Unified Approach to Interpreting Model Predictions. In *Proceedings of the 31st International Conference on Neural Information Processing Systems* (Long Beach, California, USA) *(NIPS'17)*. Curran Associates Inc., Red Hook, NY, USA, 4768–4777.

[33] Tim Miller. 2019. Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence* 267 (2019). https://doi.org/10.1016/j.artint.2018.07.007

[34] Christoph Molnar, Giuseppe Casalicchio, and B. Bischl. 2020. Interpretable Machine Learning - A Brief History, State-of-the-Art and Challenges. In *PKDD/ECML Workshops*.

[35] Harsha Nori, Samuel Jenkins, Paul Koch, and Rich Caruana. 2019. InterpretML: A Unified Framework for Machine Learning Interpretability. arXiv:1909.09223 [cs.LG]

[36] Nicolas Papernot, Fartash Faghri, Nicholas Carlini, Ian Goodfellow, Reuben Feinman, Alexey Kurakin, Cihang Xie, Yash Sharma, Tom Brown, Aurko Roy, Alexander Matyasko, Vahid Behzadan, Karen Hambardzumyan, Zhishuai Zhang, Yi-Lin Juang, Zhi Li, Ryan Sheatsley, Abhibhav Garg, Jonathan Uesato, Willi Gierke, Yinpeng Dong, David Berthelot, Paul Hendricks, Jonas Rauber, Rujun Long, and Patrick McDaniel. 2018. Technical Report on the CleverHans v2.1.0 Adversarial Examples Library. arXiv:1610.00768 [cs.LG]

[37] Hatim A. Rahman. 2021. The Invisible Cage: Workers' Reactivity to Opaque Algorithmic Evaluations. *Administrative Science Quarterly* 66, 4 (2021), 945–988. https://doi.org/10.1177/00018392211010118

[38] Lubna Razaq, Beth Kolko, and Gary Hsieh. 2022. Making crafting visible while rendering labor invisible on the Etsy platform. In *Designing Interactive Systems Conference 2021* (Virtual Event, USA) *(DIS '22)*. Association for Computing Machinery, New York, NY, USA, 15 pages.

[39] Cynthia Rudin, Chaofan Chen, Zhi Chen, Haiyang Huang, Lesia Semenova, and Chudi Zhong. 2021. Interpretable Machine Learning: Fundamental Principles and 10 Grand Challenges. *ArXiv* abs/2103.11251 (2021).

[40] Philipp Schmidt and Felix Biessmann. 2019. Quantifying Interpretability and Trust in Machine Learning Systems. arXiv:1901.08558 [cs.LG]

[41] Kumba Sennaar. 2019. Machine Learning for Recruiting and Hiring – 6 Current Applications. https://emerj.com/ai-sector-overviews/machine-learning-for-recruiting-and-hiring/. Accessed: 2020-10-15.

[42] Kunal Shah, Akshaykumar Salunke, Saurabh Dongare, and Kisandas Antala. 2017. Recommender systems: An overview of different approaches to recommendations. In *2017 International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS)*. IEEE, 1–4.

[43] Guy Shani and Asela Gunawardana. 2011. *Evaluating Recommendation Systems*. Springer US, Boston, MA, 257–297. https://doi.org/10.1007/978-0-387-85820-3_8

[44] Gagandeep Singh, Timon Gehr, Matthew Mirman, Markus Püschel, and Martin T Vechev. 2018. Fast and Effective Robustness Certification. *NeurIPS* 1, 4 (2018), 6.

[45] Saurav Singla. 2020. Machine Learning to Predict Credit Risk in Lending Industry. https://www.aitimejournal.com/@saurav.singla/machine-learning-to-predict-credit-risk-in-lending-industry. Accessed: 2020-10-15.

[46] Özge Sürer, Robin Burke, and Edward C. Malthouse. 2018. Multistakeholder Recommendation with Provider Constraints. In *Proceedings of the 12th ACM Conference on Recommender Systems* (Vancouver, British Columbia, Canada) *(RecSys '18)*. Association for Computing Machinery, New York, NY, USA, 9 pages. https://doi.org/10.1145/3240323.3240350

[47] Juntao Tan, Shuyuan Xu, Yingqiang Ge, Yunqi Li, Xu Chen, and Yongfeng Zhang. 2021. Counterfactual Explainable Recommendation. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. Association for Computing Machinery, New York, NY, USA, 10 pages.

[48] Jason Tashea. 2017. Courts Are Using AI to Sentence Criminals. That Must Stop Now. https://www.wired.com/2017/04/courts-using-ai-sentence-criminals-must-stop-now/. Accessed: 2020-10-15.

[49] Poonam B Thorat, RM Goudar, and Sunita Barve. 2015. Survey on collaborative filtering, content-based filtering and hybrid recommendation system. *International Journal of Computer Applications* 110, 4 (2015), 31–36.

[50] Tianchi. 2018. Ad Display/Click Data on Taobao.com. https://tianchi.aliyun.com/dataset/dataDetail?dataId=56

[51] Khanh Hiep Tran, Azin Ghazimatin, and Rishiraj Saha Roy. 2021. *Counterfactual Explanations for Neural Recommenders*. Association for Computing Machinery, New York, NY, USA, 1627–1631. https://doi.org/10.1145/3404835.3463005

[52] Kush R. Varshney. 2021. *Trustworthy Machine Learning*. Chappaqua, NY, USA. http://trustworthymachinelearning.com.

[53] Sahil Verma, John Dickerson, and Keegan Hines. 2020. Counterfactual Explanations for Machine Learning: A Review. arXiv:2010.10596 [cs.LG]

[54] Sahil Verma and Julia Rubin. 2018. Fairness Definitions Explained. In *Proceedings of the International Workshop on Software Fairness* (Gothenburg, Sweden) *(FairWare '18)*. Association for Computing Machinery, New York, NY, USA, 1–7. https://doi.org/10.1145/3194770.3194776

[55] Sandra Wachter, Brent Mittelstadt, and Chris Russell. 2018. Counterfactual Explanations without Opening the Black Box: Automated Decisions and the GDPR. arXiv:1711.00399 [cs.AI]

[56] Mengting Wan and Julian J. McAuley. 2018. Item recommendation on monotonic behavior chains. In *Proceedings of the 12th ACM Conference on Recommender Systems, RecSys 2018, Vancouver, BC, Canada, October 2-7, 2018*. ACM, 86–94. https://doi.org/10.1145/3240323.3240369

[57] Mengting Wan, Rishabh Misra, Ndapa Nakashole, and Julian J. McAuley. 2019. Fine-Grained Spoiler Detection from Large-Scale Review Corpora. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*. Association for Computational Linguistics, 2605–2610. https://doi.org/10.18653/v1/p19-1248

[58] Adrian Weller. 2017. Transparency: Motivations and Challenges. https://doi.org/10.48550/ARXIV.1708.01870

[59] Huan Xu and Shie Mannor. 2012. Robustness and generalization. *Machine learning* 86, 3 (2012), 391–423.

[60] Shuyuan Xu, Yunqi Li, Shuchang Liu, Zuohui Fu, and Yongfeng Zhang. 2020. Learning Post-Hoc Causal Explanations for Recommendation.

[61] Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez Rodriguez, and Krishna P. Gummadi. 2017. Fairness Beyond Disparate Treatment & Disparate Impact: Learning Classification without Disparate Mistreatment. In *Proceedings of the 26th International Conference on World Wide Web* (Perth, Australia) *(WWW '17)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE, 1171–1180. https://doi.org/10.1145/3038912.3052660

[62] Richard Zemel, Yu Wu, Kevin Swersky, Toniann Pitassi, and Cynthia Dwork. 2013. Learning Fair Representations. In *Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28* (Atlanta, GA, USA) *(ICML'13)*. JMLR.org, 325–333.

[63] Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric Xing, Laurent El Ghaoui, and Michael Jordan. 2019. Theoretically Principled Trade-off between Robustness and Accuracy. In *Proceedings of the 36th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 97)*, Kamalika Chaudhuri and Ruslan Salakhutdinov (Eds.). PMLR, 7472–7482. https://proceedings.mlr.press/v97/zhang19p.html

[64] Yongfeng Zhang and Xu Chen. 2020. Explainable Recommendation: A Survey and New Perspectives. *Found. Trends Inf. Retr.* 14 (2020), 1–101.

[65] Yong Zheng. 2019. Multi-Stakeholder Recommendations: Case Studies, Methods and Challenges *(RecSys '19)*. Association for Computing Machinery, New York, NY, USA, 2 pages. https://doi.org/10.1145/3298689.3346951

A RESULTS AND DISCUSSION

Figures 1 to 3 show the success rate of RecRec for items at different original ranks for the recommender systems trained on MovieLens-100K, AliEC Ads, and Goodreads respectively. We see that for all the three recommender systems, RecRec achieves very high success rate (more than 80% for all datasets) at very low sampling of 1% or less of the target group. The success rate improves to 100% for all datasets when the sampling percentage is increased to 5-20% depending on the dataset.

Figure 4 shows the percentage of item features that need to be changed to execute the recourse. Across all the three recommender systems, user groups, and original item ranks, the number of changes required to get a recourse decreases with increasing sampling percentage, and eventually becomes negligible.

The similarity between the original and new recommendations for all users was consistently very high. For MovieLens-100K, the average RBO metric (across user groups, original ranks, and sampling percentages) was 99.43%, for AliEC Ads, it was 99.99%, and for Goodreads, it was 99.92%. Therefore, the recourses generated by RecRec do not change the user's recommendations significantly.

Therefore, the metrics demonstrate that RecRec is able to generate valid recourses for recommender systems with a small sampling of the target group. The recourses it generates do not require a large number of changes to the item's features, and the new recommendations for the target users do not change significantly from the original recommendations.

B CONCLUSIONS

We propose a novel approach called RecRec that generates recourses targeted towards content providers in a content filtering-based recommender system platforms. Previous research has established how important it is for the content providers to understand the factors that influence the ranking of their product on the recommender system based platforms [16, 25, 37, 38], and thereby improve their product's ranking. To the best of our knowledge, RecRec is the first approach to generate recourses for recommender systems. Experimental results with three datasets used to train the recommender systems demonstrate RecRec's ability to generate recourses that satisfy the desiderata (Section 2) of a recourse. It generates recourses that have a high success rate, require a small number of changes to the item's features, and cause little to no side-effect on the target users' recommendations.

Appendix A RELATED WORK

Machine Learning (ML) is being increasingly used to automate decisions. Some of the applications where ML is being used are highly critical and directly affect humans, for example, loan approval [45], criminal justice [48], and hiring [41]. The nascent field of trustworthy ML aims to detect bias in ML models (and counteract it), understand the factors that the ML model is using in making predictions, ensure the models respect privacy and security, and frame policies and regulations that the ML models should abide by [5, 6, 58]. Research has established a few dimensions of trustworthy ML, like, fairness [6, 12, 20, 23, 54, 61, 62], interpretability [4, 15, 18, 30, 34, 35, 39, 40], robustness [11, 13, 19, 27, 36, 44, 59, 63]. In this work, we focus on interpretability and refer the readers to work by Barocas et al. [6] and Varshney [52] for a broad discussion of trustworthy machine learning.

A.1 Interpretability in ML

Interpretability is the branch of trustworthy ML that aims to provide human consumable explanations for the predictions made by ML models that are used for tasks such as classification, regression, and recommendation. Most of the research in this area has focused on the interpretability of classification models. Interpretability for classification models can be achieved by either developing inherently interpretable ML models (e.g., logistic regression, shallow decision trees) or a post-hoc explanation of complex ML models (e.g., random forests, neural networks). Post-hoc explanations can be further bifurcated into generating feature attributions using techniques like SHAP [32] or generating counterfactual explanation-based recourses [55]. Feature attribution explanations highlight the features that might have been important in making a prediction. On the other hand, counterfactual explanations provide a counterfactual situation that would have led to a different prediction from the ML model. Miller [33] in a social science study remarked that when people ask 'Why P?' questions, they are typically asking 'Why P rather than Q?', where Q is implicit in the context of the application. An example of this case is the question a person whose loan request has been rejected would ask: 'Why has my loan request been rejected?', which actually means: 'Why has my loan request been rejected instead of being accepted?'. And counterfactual explanations are a way to answer this question. They would respond, for example, by saying that 'had your income been $3000 higher, you would have gotten the loan'. This simultaneously also provides a recourse to the affected individual, who now knows that they can get the loan if they can increase their income by $3000.

A.2 Interpretability in Recommender Systems

Literature in interpretability for recommender systems has focused on highlighting the factors that might have contributed to a recommendation. This is similar to feature attribution based explanations for classification models. Interpretability research for recommender systems can be categorized into user-based, item-based, and feature-based explanations. In user-based explanations, a high rating for the item provided by a group of users similar to the user is given as an explanation for the recommended item. In item-based explanations, the recommended item is explained by its similarity to the items that the user has liked or purchased in the past. Feature-based explanations highlight the features of the recommended item that the user has shown interest in the past, for example, the cast for movie recommendations. The approaches that generate these explanations can either be model-specific or model agnostic. We refer the readers to a survey on the explainability of recommender systems for a more comprehensive discussion [64] on this topic. Similar to feature attribution explanation for classification models, a noticeable characteristic of the aforementioned explanations for recommender systems is that they are not actionable.

RecRec, on the other hand, generates recourses targeted towards the content providers of the recommender systems. This is the main contribution of this work. It provides counterfactual features that would lead to a different ranking of a specific item in the recommendation list for a target group of users.

A.2.1 Previous studies on need for recourse for content providers In this subsection we continue to discuss related work that has highlighted the need for recourse for content providers. Jhaver et al. [25] did a study with several Airbnb hosts to understand their perspectives. They clearly expressed the need for transparency and recourse on the platform. One of the hosts said: *"I feel less motivated because I don't think that it's clear what I need to do, and I think that it's frustrating seeing the search: lots of listings that are worse than mine are in higher positions."* Several hosts performed A/B testing with different factors like pricing adjustments, calendar updates, location, type of room, amenities like free parking, changing descriptions of the property, allowing dogs, allowing short-term vs. long-term guests, etc., to understand which factors can help improve their ranking. Rahman [37] interviewed freelancers working on Upwork. Freelancers also struggled in understanding what factors go into the ranking and how they can influence it. One of the freelancers said: *"all I can think about is figuring out how to raise my score"*. Similar to Airbnb hosts, freelancers tried and tested changing different attributes of their profile, like taking technical tests provided by Upwork, opening and closing contracts, having shorter projects, and inflating the hourly working rate to improve their ranking. A precisely similar need and behavior was observed when studies were conducted with sellers on Facebook MarketPlace [16], freelancers on other platforms like TaskRabbit and Fiverr [9, 24], drivers using Uber and Lyft [29], and sellers on handmade product platform Etsy [38]. Several freelancers who Rahman [37] interviewed mentioned that owing to the black-box behavior of the algorithmic freelancing platform, they made frequent efforts to take the work offline or pause the work to pacify the algorithm, and some even quit. Jarrahi and Sutherland [24] had the same observations in their interviews with Upwork's freelancers. Providing recourses to content providers of a platform would help them understand what actions they can take to improve their product's ranking and get transparency into the current ranking.

A.2.2 Counterfactual Explanations in Recommender Systems There have been recent proposals for some approaches that seek to generate explanations for recommended item in a counterfactual manner [17, 26, 47, 51, 60]. All these approaches explain a recommendation by finding the smallest change in the user's interaction history that would replace the top recommendation with anything else. For example, an explanation for the top-recommended movie *The Godfather II* is that the user had previously liked *Goodfellas* and *The Godfather*. Had the user not liked these two movies, *The God-Father II* would not be the top recommendation (it could still be recommended but at a different rank). RecRec is distinct from these works in several ways:

- RecRec provides recourse to the content providers of the recommender system, while the aforementioned approaches provide explanations to the users of the recommender system.
- RecRec provides a set of actions that can be executed to get a favorable rank for an item while the aforementioned approaches do not provide that. They only find the smallest change that would replace the top recommendation with anything else, not something the content provider or the user wants.
- RecRec can make suggestions to change features that are *not in* the user's history, while the aforementioned approaches only alter the user's history.
- RecRec is able to provide a recourse for items at any rank (in order to get them to an improved rank) in the recommended list, while the aforementioned approaches provide an explanation for only the top-ranked item.

Our work also has subtle similarity to the work by Dean et al. [14], where they define reachability as the feasibility of the end-user of a recommender system modifying their rating in order to get an item recommended. RecRec is distinct from Dean et al. [14]'s work in the following ways:

- Their work is concerned with only the end-user of a recommender system, while RecRec is targeted towards the content providers of the recommender system.
- The goal of their work is to audit the recommender system to understand whether it could cause polarization or filter bubbles, while the goal of RecRec is to provide recourse to the content providers of the recommender system.
- Their approach is limited to matrix factorization based recommender systems, while RecRec generalizes not only to all architectures of collaborative filtering recommender systems, but also to content based recommender systems.