Towards Employing Recommender Systems for Supporting Data and Algorithm Sharing

Peter Müllner
Know-Center Gmbh
Graz, Austria
pmuellner@know-center.at

Stefan Schmerda
Know-Center Gmbh
Graz, Austria
sschmerda@know-center.at

Dieter Theiler
Know-Center Gmbh
Graz, Austria
dtheiler@know-center.at

Stefanie Lindstaedt
Know-Center Gmbh & TU Graz
Graz, Austria
slind@know-center.at

Dominik Kowald
Know-Center Gmbh & TU Graz
Graz, Austria
dkowald@know-center.at

ABSTRACT

Data and algorithm sharing is an imperative part of data- and AI-driven economies. The efficient sharing of data and algorithms relies on the active interplay between users, data providers, and algorithm providers. Although recommender systems are known to effectively interconnect users and items in e-commerce settings, there is a lack of research on the applicability of recommender systems for data and algorithm sharing. To fill this gap, we identify six recommendation scenarios for supporting data and algorithm sharing, where four of these scenarios substantially differ from the traditional recommendation scenarios in e-commerce applications. We evaluate these recommendation scenarios using a novel dataset based on interaction data of the OpenML data and algorithm sharing platform, which we also provide for the scientific community. Specifically, we investigate three types of recommendation approaches, namely popularity-, collaboration-, and content-based recommendations. We find that collaboration-based recommendations provide the most accurate recommendations in all scenarios. Plus, the recommendation accuracy strongly depends on the specific scenario, e.g., algorithm recommendations for users are a more difficult problem than algorithm recommendations for datasets. Finally, the content-based approach generates the least popularity-biased recommendations that cover the most datasets and algorithms.

CCS CONCEPTS

• **Information systems** → **Recommender systems**; **Collaborative filtering**.

KEYWORDS

recommender systems, data economy, AI-driven economy, data and algorithm sharing, popularity bias, collaborative filtering, content-based filtering

DE '22, December 9, 2022, Roma, Italy

ACM ISBN 978-1-4503-9923-4/22/12…$15.00
https://doi.org/10.1145/3565011.3569055

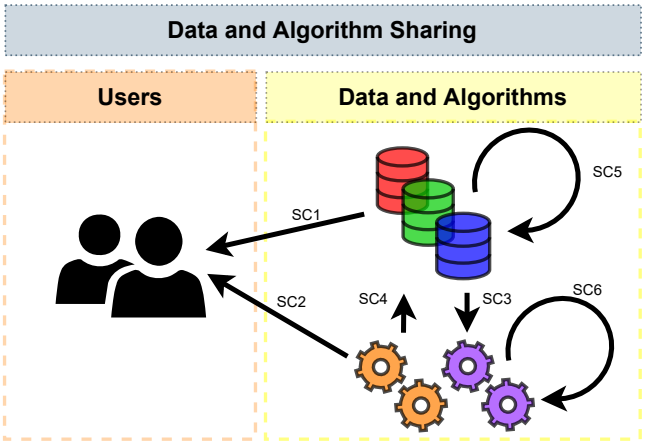

Figure 1: Recommendation scenarios SC1-SC6 that can support data and algorithm sharing. In addition to the traditional item-to-user scenarios SC1 and SC2, also item-to-item scenarios SC3-SC6 can occur.

ACM Reference Format:
Peter Müllner, Stefan Schmerda, Dieter Theiler, Stefanie Lindstaedt, and Dominik Kowald. 2022. Towards Employing Recommender Systems for Supporting Data and Algorithm Sharing. In *Data Economy (DE '22), December 9, 2022, Roma, Italy*. ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3565011.3569055

1 INTRODUCTION

Sharing data and algorithms is one important cornerstone in today's data- and AI-driven economy. To enable data and algorithm sharing, interconnecting three key-players is essential: data providers, algorithm providers, and users. *Data Providers* grant access to their data collections. *Algorithm Providers* allow applying their algorithms to a given piece of data. *Users* apply algorithms to data and, this way, connect data and algorithms. In general, data and algorithm providers may share their resources due to various reasons, e.g., to monetize the data or the algorithm, or to make them available for the research community. The powerful strength of data and algorithm sharing lies in the exploitation of shared resources, e.g., data shared by a data provider. For example, it might be advantageous for companies to gain access to the best-suited data to enhance

their AI pipeline. However, selecting the best-suited dataset is hard, which stems from the fact that the number of available datasets, publicly available over the Web or stored in private databases, has increased rapidly over the last decade [8, 11, 20, 31].

Although the deployment of recommender systems for applications in e-commerce, e.g., Amazon or Zalando, is a natural decision to address this choice overload, not much research is available on the applicability of recommender systems for data and algorithm sharing (see Section 2). This is especially true for beyond-accuracy objectives of recommender systems, such as popularity bias, which is currently an important topic in the research community. Recommender systems exhibiting popularity bias tend to exclude many datasets and algorithms from their recommendations and recommend popular items substantially more often than non-popular items [7, 13, 14].

To study to what extent recommender systems can support data and algorithm sharing, we identify six recommendation scenarios (see Figure 1). In these scenarios, we evaluate three recommendation methods, i.e., Most Popular, Collaborative Filtering, and Content-based Filtering, with respect to recommendation accuracy and popularity bias. The three main-contributions of this paper are as follows:

(1) We discuss six recommendation scenarios and outline how recommender systems can be applied to support data and algorithm sharing (see Section 3).
(2) We create and publish a novel dataset based on the OpenML platform, which allows studying recommender systems for data and algorithm sharing (see Section 4).
(3) We show that Collaborate Filtering yields the most accurate recommendations and Content-based Filtering can generate recommendations that cover the most datasets and algorithms (see Section 5).

2 RELATED WORK

Recommender systems for data and algorithms are of growing interest to both academia and industry in the field of data and AI-driven economies [8, 11, 25].

For example, Patra et al. [25] utilize Content-based Filtering for dataset recommendations in the genetics domain. Also, Jess et al. [11] design a recommender system for artificial data to help human decision-making in the industrial domain. The task of algorithm recommendations has been partially approached by Automated Machine Learning, which aims to automatically select an appropriate machine learning pipeline (including algorithms) for a given dataset and problem [9]. For example, Zschech et al. [37] recommend a data mining pipeline for a given problem. Vainshtein et al. [32] and Song et al. [30] exploit metadata and structural properties of datasets to recommend classification algorithms.

Numerous works exist that evaluate recommender systems for popularity bias, i.e., their inclination to recommend popular items [7, 22, 36]. For example, Mansoury et al. [22] show that recommender systems can seriously exacerbate existing biases, such as popularity bias. Also, Zhu et al. [36] simulate a recommender system to monitor the evolution of popularity bias. Within this dynamic setting, the authors studied factors that drive popularity bias.

Table 1: Profile data is used to generate recommendations, which are then evaluated against the ground truth data. In the item-to-user scenarios SC1 and SC2, profile and ground truth data are available via direct user-to-item interactions (e.g., user utilizes a dataset). However, for our remaining scenarios, i.e., SC3-SC6, profile and/or ground truth data is available only via indirect item-to-item interactions, or unavailable (✗), and needs to be constructed.

	Profile Data	Ground Truth
SC1: Datasets to Users	direct	direct
SC2: Alsorithms to Users	direct	direct
SC3: Datasets to Algorithms	indirect	indirect
SC4: Algorithms to Datasets	indirect	indirect
SC5: Datasets to Datasets	indirect	✗
SC6: Algorithms to Algorithms	indirect	✗

Data Market Austria (DMA)[1] is an example of a data- and AI-driven economy, in which a recommender system is employed to connect users, data, and algorithms [15]. However, the authors raise concerns regarding the dataset used in their study with respect to valid connections between users, datasets, and algorithms, and do not consider content-based recommendations. Plus, our work includes a beyond-accuracy evaluation study with respect to popularity bias.

3 RECOMMENDATION SCENARIOS

Recommender systems rely on (i) profile data for model training and (ii) ground truth data for model evaluation. In a traditional item-to-user recommendation scenario (SC1 and SC2), *profile data* refers to the user profile that represents a user's item preferences. *Ground truth data* represents the user's item preferences the recommender system aims to predict. Typically, the direct interactions between users and items (e.g., a user's utilization of a certain dataset) are used as the users' item preferences. However, for the remaining item-to-item recommendation scenarios (SC3-SC6) there is no direct item-to-item interaction data that can be used to generate recommendations, e.g., dataset to algorithm recommendations (see Table 1).

Thus, in the following, we detail our six recommendation scenarios that can occur in data and algorithm sharing (see Figure 1) and give examples how recommender systems can cope with the lack of direct interactions for item-to-item recommendation scenarios:

SC1: Datasets to Users. In SC1, recommendations help users (e.g., researchers) to identify datasets that are deemed to be relevant. As Figure 1 illustrates, there exists a direct interaction between users and datasets (e.g., a user uses a dataset to train an algorithm). Thus, the recommender system can leverage these interactions to generate recommendations.

SC2: Algorithms to Users. In SC2, recommendations help users (e.g., researchers) to identify algorithms that are deemed to be relevant. As in SC1, also in SC2, the recommender system can leverage the direct interactions between users and algorithms to generate recommendations.

[1] https://www.datamarket.at/

In addition to traditional item-to-user recommendations, also item-to-item recommendations can occur in data and algorithm sharing (see Figure 1). However, items, i.e., datasets and algorithms, do not directly interact with each other; a user has to run an algorithm on a given dataset. Therefore, we rely on direct user-to-dataset and user-to-algorithm interactions to indirectly interconnect datasets and algorithms.

SC3: Datasets to Algorithms. In SC3, recommendations help to identify suitable datasets to train a given algorithm. This scenario can occur when, e.g., algorithm providers or researchers aim to improve their algorithm via leveraging more datasets for training. In contrast to SC1 and SC2, indirect item-to-item interaction data has to be used to generate recommendations. Since users interact with algorithms and datasets, we can use user interactions to connect datasets and algorithms. Specifically, the profile and ground truth data of an algorithm consists of the datasets that users used to train the algorithm.

SC4: Algorithms to Datasets. In SC4, recommendations help to identify suitable algorithms that can be applied to a given dataset. This scenario can occur when dataset providers or researchers aim to find other algorithms that can be applied to their dataset, e.g., to extract a different kind of knowledge from the data. Similar to SC3, the profile and ground truth data of a dataset consists of the algorithms that users run on the dataset.

Finally, with SC5 and SC6, we have two additional item-to-item recommendation scenarios, but this time, the same item types are interlinked (e.g., datasets are recommended for a given dataset), which leads to a different situation with respect to the available ground truth data:

SC5: Datasets to Datasets. In SC5, data providers can find other datasets that are utilized by the same user-community. This is an important scenario in data economies, as this can identify datasets of competing data providers. For SC5, we build the profile data in the same way as in case of SC4, i.e., the profile of a dataset consists of the algorithms that users run on the dataset. However, for building the ground truth data, we cannot use this idea, since we need a set of relevant datasets for a given dataset. To build this set, we create a collaboration network, similarly as in [15]. This means that we create a link between two datasets if they have been used by the same user. Thus, the ground truth data of a given dataset consists of the datasets that have the largest user overlap with this dataset.

SC6: Algorithms to Algorithms. In this scenario, algorithm providers can find other algorithms that are utilized by the same user-community. Similar to SC5, this is an important scenario in AI-driven economies, as this can identify algorithms of competing algorithm providers. In case of SC6, we use the same idea as in case of SC5 to create profile and ground truth data. Hence, the profile of an algorithm consists of the datasets that users' used to train this algorithm and the ground truth data consists of the algorithms with the largest user overlap.

4 METHOD

4.1 OpenML Dataset

Due to the lack of available datasets to evaluate recommender systems for data and algorithm sharing, we gather data from the dataset and algorithm sharing platform OpenML, and share it with the research community.

Table 2: Descriptive statistics of our OpenML dataset $\mathcal{D}$. Many datasets and algorithms have no interactions. Thus, in contrast to content-based recommendation methods, interaction-based recommendation methods cannot recommend these datasets and algorithms.

Users	512
Algorithms	1,307
Datasets	573
Interactions	10,945
Avg. Interactions / User	21.38
Avg. Interactions / Algorithm	8.37
Avg. Interactions / Dataset	19.10
$\text{Algorithms}_{w/o\ Int.}$	11,037
$\text{Datasets}_{w/o\ Int.}$	2,104

Data crawling. In OpenML, users can upload datasets, algorithms, or their entire machine learning pipelines, e.g., a user u applied an algorithm a (e.g., a classification algorithm) to a dataset d. This represents user interactions between datasets and algorithms. Additionally, OpenML provides a powerful and convenient Python-based API[2], which makes it an ideal platform to evaluate our recommendation scenarios. Our data crawling procedure is as follows:

(1) We use *openml.datasets.list_datasets* to fetch all datasets and *openml.flows.list_flows* to obtain all algorithms alongside their textual descriptions. Herein, we ignore case sensitivity and apply stemming.
(2) Then, we fetch triples containing the user, algorithm, and task (e.g., classification) with *openml.runs.list_runs* to obtain user interactions and retrieve the dataset to which the user applied the algorithm by querying *openml.tasks.OpenMLTask*.
(3) Since users can apply the same algorithm to the same dataset multiple times, we cope with these repeated interactions [33] by merging all repetitions.

After these three steps, our novel dataset includes 8,637,795 interactions between 544 users, 2,186 datasets, and 5,660 algorithms, as well as 2,104 datasets and 11,037 algorithms without user interactions.

However, we notice that there exist users with an extraordinarily large number of interactions. Through close inspection, we observe that these users are used to test the platform (e.g., bots). Thus, we remove all users, whose number of interactions exceeds the point of maximal curvature of the logarithmic-transformed interaction-distribution [28]. Further descriptive statistics of our OpenML dataset $\mathcal{D}$ can be found in Table 2. Plus, to foster the reproducibility of our research, we provide the dataset freely via Zenodo[3].

Train- and test-set split. To evaluate the performance of our recommendation methods for our recommendation scenarios, we randomly split the interactions of each target entity (i.e., user, dataset, or algorithm depending on the scenario) in our dataset

[2]https://docs.openml.org/Python-API/
[3]https://doi.org/10.5281/zenodo.6517031

$\mathcal{D}$ into 80% profile data used for training and 20% ground truth data used for testing. With this, for each target entity, our recommendation algorithms can utilize sufficient data for training and testing, which enables a meaningful evaluation. As described in Section 3, in SC3 and SC4, we create this profile and ground truth data using indirect item-to-item interactions (i.e., via user interactions). In SC5 and SC6, we create ground truth data via constructing a collaboration network [15]. This means that we connect two datasets (SC5) or two algorithms (SC6) if they were used by the same user, and put the 10 datasets or algorithms into the test set with the largest user overlap.

4.2 Recommendation Methods

In the following, we present three recommendation methods [27] that we evaluate in our six recommendation scenarios for data and algorithm sharing. Furthermore, we note that all methods generate recommendation lists of size $n = 10$ and that the recommendation lists do not contain items that the target entity already knows, i.e., we filter items in the recommendation lists for which the target entity already has interactions in the profile data. We calculate all recommendations using the Java-based recommendation framework ScaR (Scalable Recommendation-as-a-service) [16, 18], and based on three built-in recommendation methods that we adapt to our data and algorithm sharing problem:

Most Popular (MP). This unpersonalized approach recommends datasets or algorithms that users interact with the most, i.e., the most popular datasets or algorithms. This way, only a small set of available items can be recommended, even though other items may be better suited for the target entity. However, MP is always capable of recommending items, even in user cold-start settings [17].

Collaborative Filtering (CF). CF exploits direct or indirect interaction data between users, data, and algorithms within the data- and AI-driven economy. For example, CF recommends a dataset d to a user u, if similar users (u's neighbors) have interacted with d. Thus, CF provides personalized recommendations generated for a target entity. Our CF variant is a user-based k-nearest neighbor approach with $k = 40$ neighbors, which is the default setting in the ScaR framework [16, 18].

Content-based Filtering (CB). MP and CF rely on interaction data to generate recommendations and, therefore, are prone to popularity bias [1, 6]. As a remedy, CB generates personalized recommendations by leveraging content data, i.e., textual descriptions, to identify datasets or algorithms that are deemed to be relevant for the target entity. We implement CB by using TF-IDF representations [4, 12] of the description text of datasets and algorithms. Here, we set the minimum term frequency to 1 and the minimum document frequency to 2, which are the default settings in the ScaR framework [16, 18].

4.3 Evaluation Criteria

We evaluate our three recommendation methods based on two evaluation criteria: (i) accuracy, and (ii) popularity bias:

Accuracy. To evaluate recommendation accuracy, we use five widely-used metrics [24]: Precision P@k, Recall R@k, Mean Reciprocal Rank MRR@k, Mean Absolute Precision MAP@k, and Normalized Discounted Cumulative Gain nDCG@k [10]. Here, P@k is the fraction of recommended items that are relevant, R@k is the fraction of relevant items that are recommended, MRR@k [26] is the average reciprocal position of the relevant items in target entities' recommendation lists, MAP@k measures the quality of the ranked recommendation list by penalizing relevant items that occur later in the ranking, and nDCG@k also takes the ranking into account but is based on cumulative gain [34].

Popularity Bias. To evaluate popularity bias, we use two metrics: (i) Item Space Coverage [29] (Cov@k) and (ii) Average Recommendation Popularity (RecPop@k). Cov@k is the fraction of the item catalog that is recommended to at least one target entity, and RecPop@k is the average popularity of the recommended items. An item's popularity is given by the number of interactions for this item.

5 RESULTS

In this section, we present the results of our experiments, in which we evaluate three recommendation methods in six recommendation scenarios along our two evaluation criteria (i) accuracy, and (ii) popularity bias.

5.1 Accuracy

Across our three recommendation methods Most Popular (MP), Collaborative Filtering (CF), and Content-based Filtering (CB), in Table 3, we observe that CF provides the most accurate recommendations in all our six recommendation scenarios. The most accurate recommendations across our six recommendation scenarios can be generated by CF in SC4 (Algorithms to Datasets), while the recommendations generated in SC6 (Algorithms to Algorithms) are the least accurate. This is interesting since SC4 is a recommendation scenario, in which profile and ground truth data can only be constructed using indirect item-to-item interactions, as discussed in Section 3. However, in SC4, there exists a large item catalog (i.e., 1,307 algorithms) that CF can recommend for a few target entities (i.e., 573 datasets). This small dataset-to-algorithm ratio can positively impact accuracy, since selected neighbors (i.e., similar datasets) used for generating recommendations tend to be more reliable due to more co-interacted algorithms [5].

In SC2 (Algorithms to Users), the same item catalog can be recommended to a similarly small number of target entities - in this case 512 users. However, recommendation accuracy is substantially smaller than for SC4. As shown in Table 2, users and datasets have a similar average number of interactions, but 50% of users have more than 6 interactions, while only 28% of datasets have more than 6 interactions. This suggests that generating recommendations for users is more difficult than generating recommendations for datasets, possibly due to users' larger profile data.

In the case of SC6 (Algorithms to Algorithms), the recommender system needs to cope with the largest item catalog across our six recommendation scenarios. Also, recommendations need to be generated for the very same large set of items (i.e., 1,307 algorithms can be recommended to 1,307 algorithms). Due to this sparse interaction space, all our three recommendation methods seem to struggle with providing accurate recommendations. However, the high accuracy in SC5 (Datasets to Datasets) shows that our approach for generating ground truth data in cases where the same item type

Table 3: Our results show that CF provides the most accurate recommendations in all six recommendation scenarios. However, as Cov@10 indicates, CF can only recommend a small fraction of the item catalog (i.e., datasets or algorithms). In contrast, CB can recommend the largest fraction of the item catalog, and provides the least popularity-biased recommendations.

Recommendation Scenario	Method	P@1	R@10	MRR@10	MAP@10	nDCG@10	Cov@10	RecPop@10
SC1 (Datasets to Users)	MP	0.00	0.22	0.04	0.04	0.08	0.01	593.79
	CF	**0.26**	**0.34**	**0.26**	**0.27**	**0.30**	0.06	181.50
	CB	0.05	0.05	0.03	0.02	0.04	**0.12**	**10.25**
SC2 (Algorithms to Users)	MP	0.03	0.11	0.05	0.05	0.07	0.00	265.75
	CF	**0.12**	**0.26**	**0.14**	**0.14**	**0.18**	0.02	90.51
	CB	0.02	0.06	0.02	0.03	0.03	**0.03**	**9.25**
SC3 (Datasets to Algorithms)	MP	0.00	0.12	0.02	0.02	0.04	0.01	555.20
	CF	**0.33**	**0.39**	**0.28**	**0.32**	**0.35**	0.06	143.36
	CB	0.00	0.13	0.06	0.06	0.09	**0.14**	**7.07**
SC4 (Algorithms to Datasets)	MP	0.01	0.29	0.12	0.13	0.18	0.00	270.62
	CF	**0.52**	**0.56**	**0.42**	**0.45**	**0.51**	0.01	97.56
	CB	0.01	0.03	0.01	0.01	0.02	**0.03**	**12.75**
SC5 (Datasets to Datasets)	MP	0.00	0.02	0.01	0.01	0.01	0.00	650.23
	CF	**0.17**	**0.44**	**0.17**	**0.20**	**0.28**	0.09	55.74
	CB	0.05	0.12	0.05	0.06	0.08	**0.28**	**14.88**
SC6 (Algorithms to Algorithms)	MP	0.01	0.02	0.01	0.01	0.01	0.00	278.32
	CF	**0.07**	**0.24**	**0.08**	**0.09**	**0.14**	0.02	55.01
	CB	0.04	0.12	0.04	0.05	0.07	**0.04**	**7.87**

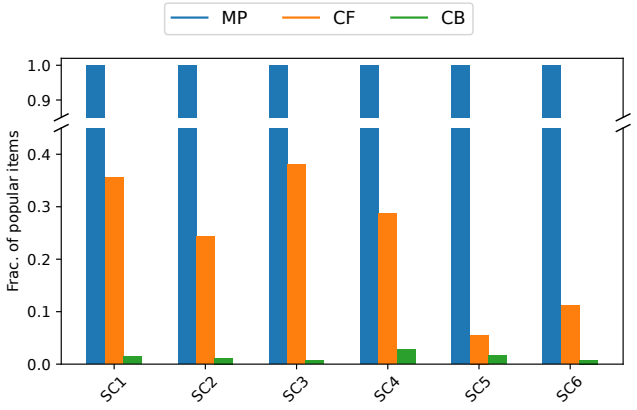

Figure 2: The fraction of popular items in a user's recommendation list. MP generates the most popularity-biased recommendations, while CB recommends the least popular items. Also, CF tends to recommend both, popular and non-popular items.

is recommended to the same item type is suitable for evaluating recommendations for data and algorithm sharing.

5.2 Popularity Bias

In our popularity bias experiments, Cov@10 in Table 3 suggests that MP can only recommend a small fraction of the item catalog, since MP always recommends the - in our case - 10 most popular items to each target entity. This way, the exploration potential of the item catalog, which represents the datasets and algorithms in the data and AI-driven economy, is limited. In contrast to MP, CB recommends the largest fraction of the item catalog in all six recommendation scenarios and, therefore, allows exploring a larger part of the data- and AI-driven economy. Plus, CB generates the least popularity-biased recommendations, since RecPop@10 exhibits a smaller value than in case of MP and CF. MP and CF are interaction-based recommendation methods, and since many items have no interactions (see Table 2), only a small part of the item catalog can be covered. There exist approaches to make popularity bias less serious, e.g., re-ranking schemes [2, 35], that penalize popular items and recommend more unpopular items. However, only content-based approaches are able to recommend items without interactions, i.e., cold-start items [19, 21].

In Figure 2, we discuss popularity bias in more detail and investigate the fraction of recommendations for popular items (i.e., the 10 most popular items a target entity has not rated yet). Similar to our results in Table 3, we can observe that in all six recommendation scenarios, MP provides the most popularity-biased recommendations, while CB tends to recommend items with low popularity, due to its ability to recommend cold-start items.

In general, CB mostly recommends non-popular items and MP recommends only popular items. CF tends to recommend both, popular and non-popular items and thus, provides more popularity-balanced recommendations. Our finding that CF provides more accurate and popularity-balanced recommendations than MP and CB is in line with recent research that shows that accurate recommendations should also take non-popular items into account in addition to popular items [3, 14, 23].

6 CONCLUSION

In this work, we evaluate the applicability of recommender systems for supporting data and algorithm sharing. We create a novel dataset based on the OpenML dataset and algorithm sharing platform, to enable an offline evaluation of three standard recommendation methods in six recommendation scenarios. Plus, we discuss our results along two criteria: recommendation accuracy and popularity bias. We find that Collaborative Filtering can generate more accurate dataset and algorithm recommendations than Most Popular and Content-based Filtering. Moreover, Content-based Filtering exhibits popularity bias to the smallest extent and can recommend many of the datasets and algorithms that are ignored by Most Popular and Collaborative Filtering. Overall, our work discusses how recommender systems can be applied within data- and AI-driven economies to support data and algorithm sharing.

Limitations and future work. We recognize two limitation of this work: we do not investigate the aspect of monetization of data and algorithm sharing in data- and AI-driven economies, and we do not test whether the target entities are satisfied with the utility of their recommendations. For example, for algorithm to user recommendations, algorithms are recommended that are considered relevant or interesting for a specific user. However, it is unclear whether the user is satisfied with the performance of the algorithm, e.g., its performance for classification tasks. With respect to monetization, our work focuses on data and algorithm sharing itself, and how recommender systems can support the interconnection of users, data, and algorithms. However, we acknowledge that, e.g., a recommended dataset might be relevant for a given user, but could exceed the user's financial possibilities. Thus, developing recommender systems that are aware of financial constraints remains an interesting avenue for future research. Moreover, in this work, we focus on three broad families of recommender systems, i.e., popularity-, collaboration-, and content-based approaches. However, our future work will also incorporate more specialized approaches as, e.g., deep learning or matrix factorization. Also, we will acknowledge that data and algorithm providers may have privacy-related, legal, ethical, or economical concerns when making resources available through recommendations. Thus, we will work on how these concerns can be respected in a recommender system, e.g., by incorporating privacy-preserving technologies.

Acknowledgements. This work is supported by the H2020 project TRUSTS (GA: 871481) and the "DDAI" and "DDIA" COMET Modules within the COMET – Competence Centers for Excellent Technologies Programme, funded by the Austrian Federal Ministry for Transport, Innovation and Technology (bmvit), the Austrian Federal Ministry for Digital and Economic Affairs (bmdw), the Austrian Research Promotion Agency (FFG), the province of Styria (SFG) and partners from industry and academia. The COMET Programme is managed by FFG.

REFERENCES

[1] Himan Abdollahpouri. 2019. Popularity bias in ranking and recommendation. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*. 529–530.

[2] Himan Abdollahpouri, Robin Burke, and Bamshad Mobasher. 2017. Controlling popularity bias in learning-to-rank recommendation. In *Proceedings of the eleventh ACM conference on recommender systems*. 42–46.

[3] Himan Abdollahpouri, Masoud Mansoury, Robin Burke, and Bamshad Mobasher. 2019. The unfairness of popularity bias in recommendation. In *Workshop on Recommendation in Multi-stakeholder Environments (RMSE 2019), in conjunction with RecSys 2019*.

[4] Palakorn Achananuparp, Xiaohua Hu, and Xiajiong Shen. 2008. The evaluation of sentence similarity measures. In *International Conference on data warehousing and knowledge discovery*. Springer, 305–316.

[5] Gediminas Adomavicius and Jingjing Zhang. 2012. Impact of data characteristics on recommender systems performance. *ACM Transactions on Management Information Systems (TMIS)* 3, 1 (2012), 1–17.

[6] Alejandro Bellogín, Pablo Castells, and Iván Cantador. 2017. Statistical biases in information retrieval metrics for recommender systems. *Information Retrieval Journal* 20, 6 (2017), 606–634.

[7] Mehdi Elahi, Danial Khosh Kholgh, Mohammad Sina Kiarostami, Sorush Saghari, Shiva Parsa Rad, and Marko Tkalčič. 2021. Investigating the impact of recommender systems on user-based and item-based popularity bias. *Information Processing & Management* 58, 5 (2021), 102655.

[8] Raul Castro Fernandez, Pranav Subramaniam, and Michael J. Franklin. 2020. Data Market Platforms: Trading Data Assets to Solve Data Problems. *Proc. VLDB Endow.* 13, 12 (jul 2020), 1933–1947. https://doi.org/10.14778/3407790.3407800

[9] Xin He, Kaiyong Zhao, and Xiaowen Chu. 2021. AutoML: A survey of the state-of-the-art. *Knowledge-Based Systems* 212 (2021), 106622.

[10] Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems (TOIS)* 20, 4 (2002), 422–446.

[11] Torben Jess, Philip Woodall, Vijay Dodwani, Mark Harrison, Duncan McFarlane, Eric Nicks, and William Krechel. 2015. An industrial data recommender system to solve the problem of data overload. In *23rd European Conference on Information Systems (ECIS 2015)*.

[12] Karen Sparck Jones. 1972. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation* (1972).

[13] Dominik Kowald, Peter Muellner, Eva Zangerle, Christine Bauer, Markus Schedl, and Elisabeth Lex. 2021. Support the underground: characteristics of beyond-mainstream music listeners. *EPJ Data Science* 10, 1 (2021), 14.

[14] Dominik Kowald, Markus Schedl, and Elisabeth Lex. 2020. The unfairness of popularity bias in music recommendation: A reproducibility study. In *European conference on information retrieval*. Springer, 35–42.

[15] Dominik Kowald, Matthias Traub, Dieter Theiler, Heimo Gursch, Emanuel Lacic, Stefanie Lindstaedt, Roman Kern, and Elisabeth Lex. 2019. Using the Open Meta Kaggle Dataset to Evaluate Tripartite Recommendations in Data Markets. In *REVEAL Workshop colocated with ACM Conference on Recommender Systems*.

[16] Emanuel Lacic, Dominik Kowald, Lukas Eberhard, Christoph Trattner, Denis Parra, and Leandro Balby Marinho. 2013. Utilizing online social network and location-based data to recommend products and categories in online marketplaces. In *Mining, Modeling, and Recommending'Things' in Social Media*. Springer, 96–115.

[17] Emanuel Lacic, Dominik Kowald, Matthias Traub, Granit Luzhnica, Jörg Peter Simon, and Elisabeth Lex. 2015. Tackling Cold-Start Users in Recommender Systems with Indoor Positioning Systems. In *Poster Proceedings of the 9th {ACM} Conference on Recommender Systems*. ACM.

[18] Emanuel Lacic, Matthias Traub, Dominik Kowald, and Elisabeth Lex. 2015. ScaR: Towards a Real-Time Recommender Framework Following the Microservices Architecture. In *Proceedings of the Workshop on Large Scale Recommender Systems (LSRS2015) at RecSys 2015*.

[19] Xuan Nhat Lam, Thuc Vu, Trong Duc Le, and Anh Duc Duong. 2008. Addressing cold-start problem in recommendation systems. In *Proceedings of the 2nd international conference on Ubiquitous information management and communication*. 208–211.

[20] Fan Liang, Wei Yu, Dou An, Qingyu Yang, Xinwen Fu, and Wei Zhao. 2018. A survey on big data market: Pricing, trading and protection. *Ieee Access* 6 (2018), 15132–15154.

[21] Blerina Lika, Kostas Kolomvatsos, and Stathes Hadjiefthymiades. 2014. Facing the cold start problem in recommender systems. *Expert Systems with Applications* 41, 4, Part 2 (2014), 2065–2073. https://doi.org/10.1016/j.eswa.2013.09.005

[22] Masoud Mansoury, Himan Abdollahpouri, Mykola Pechenizkiy, Bamshad Mobasher, and Robin Burke. 2020. Feedback loop and bias amplification in recommender systems. In *Proceedings of the 29th ACM international conference on information & knowledge management*. 2145–2148.

[23] Mohammadmehdi Naghiaei, Hossein A Rahmani, and Mahdi Dehghan. 2022. The Unfairness of Popularity Bias in Book Recommendation. In *Workshop on Algorithmic Bias in Search and Recommendation (Bias 2022), in conjunction with ECIR 2022*.

[24] Denis Parra and Shaghayegh Sahebi. 2013. Recommender systems: Sources of knowledge and evaluation metrics. In *Advanced techniques in web intelligence-2*. Springer, 149–175.

[25] Braja Gopal Patra, Kirk Roberts, and Hulin Wu. 2020. A content-based dataset recommendation system for researchers—a case study on Gene Expression Omnibus (GEO) repository. *Database* 2020 (2020).

[26] Dragomir R Radev, Hong Qi, Harris Wu, and Weiguo Fan. 2002. Evaluating Web-based Question Answering Systems.. In *LREC*. Citeseer.

[27] Francesco Ricci, Lior Rokach, and Bracha Shapira. 2011. Introduction to recommender systems handbook. In *Recommender systems handbook*. Springer, 1–35.

[28] Ville Satopaa, Jeannie Albrecht, David Irwin, and Barath Raghavan. 2011. Finding a "Kneedle" in a Haystack: Detecting Knee Points in System Behavior. In *2011 31st International Conference on Distributed Computing Systems Workshops*. IEEE, Minneapolis, MN, USA, 166–171. https://doi.org/10.1109/ICDCSW.2011.20

[29] Thiago Silveira, Min Zhang, Xiao Lin, Yiqun Liu, and Shaoping Ma. 2019. How good your recommender system is? A survey on evaluations in recommendation. *International Journal of Machine Learning and Cybernetics* 10, 5 (2019), 813–831.

[30] Qinbao Song, Guangtao Wang, and Chao Wang. 2012. Automatic recommendation of classification algorithms based on data set characteristics. *Pattern recognition* 45, 7 (2012), 2672–2689.

[31] Florian Stahl, Fabian Schomm, Gottfried Vossen, and Lara Vomfell. 2016. A classification framework for data marketplaces. *Vietnam Journal of Computer Science* 3, 3 (2016), 137–143.

[32] Roman Vainshtein, Asnat Greenstein-Messica, Gilad Katz, Bracha Shapira, and Lior Rokach. 2018. A hybrid approach for automatic model recommendation. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. 1623–1626.

[33] Chenyang Wang, Min Zhang, Weizhi Ma, Yiqun Liu, and Shaoping Ma. 2019. Modeling item-specific temporal dynamics of repeat consumption for recommender systems. In *The World Wide Web Conference*. 1977–1987.

[34] Yining Wang, Liwei Wang, Yuanzhi Li, Di He, Wei Chen, and Tie-Yan Liu. 2013. A theoretical analysis of NDCG ranking measures. In *Proceedings of the 26th annual conference on learning theory (COLT 2013)*, Vol. 8. Citeseer, 6.

[35] Yu Zheng, Chen Gao, Xiang Li, Xiangnan He, Yong Li, and Depeng Jin. 2021. Disentangling User Interest and Conformity for Recommendation with Causal Embedding. In *Proceedings of the Web Conference 2021* (Ljubljana, Slovenia) *(WWW '21)*. Association for Computing Machinery, New York, NY, USA, 2980–2991. https://doi.org/10.1145/3442381.3449788

[36] Ziwei Zhu, Yun He, Xing Zhao, and James Caverlee. 2021. Popularity Bias in Dynamic Recommendation. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 2439–2449.

[37] Patrick Zschech, Kai Heinrich, Richard Horn, and Daniel Höschele. 2019. Towards a Text-based Recommender System for Data Mining Method Selection. 25th Americas Conference on Information Systems (AMCIS).