How Dataset Characteristics Affect the Robustness of Collaborative Recommendation Models

Yashar Deldjoo, Tommaso Di Noia, Eugenio Di Sciascio, Felice Antonio Merra*
{yashar.deldjoo,tommaso.dinoia,eugenio.disciascio,felice.merra}@poliba.it
Polytechnic University of Bari
Bari, Italy

ABSTRACT

Shilling attacks against collaborative filtering (CF) models are characterized by several fake user profiles mounted on the system by an adversarial party to harvest recommendation outcomes toward a malicious desire. The vulnerability of CF models is directly tied with their reliance on the underlying interaction data —like user-item rating matrix (URM) — to train their models and their inherent inability to distinguish genuine profiles from non-genuine ones. The majority of works conducted so far for analyzing shilling attacks mainly focused on properties such as confronted recommendation models, recommendation outputs, and even users under attack. The under-researched element has been the impact of data characteristics on the effectiveness of shilling attacks on CF models.

Toward this goal, this work presents a systematic and in-depth study by using an analytical modeling approach built on a regression model to test the hypothesis of whether URM properties can impact the outcome of CF recommenders under a shilling attack. We ran extensive experiments involving 97200 simulations on three different domains (movie, business, and music), and showed that URM properties considerably affect the robustness of CF models in shilling attack scenarios. Obtained results can be of great help for the system designer in understanding the cause of variations in a recommender system performance due to a shilling attack.

CCS CONCEPTS

• **Information systems → Recommender systems**.

KEYWORDS

Recommender Systems; Security of Recommender Systems

ACM Reference Format:
Yashar Deldjoo, Tommaso Di Noia, Eugenio Di Sciascio, Felice Antonio Merra. 2020. How Dataset Characteristics Affect the Robustness of Collaborative Recommendation Models. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '20), July 25–30, 2020, Virtual Event, China.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3397271.3401046

*The authors are in alphabetical order. Corresponding author: Felice Antonio Merra (felice.merra@poliba.it).

SIGIR '20, July 25–30, 2020, Virtual Event, China

ACM ISBN 978-1-4503-8016-4/20/07…$15.00
https://doi.org/10.1145/3397271.3401046

1 INTRODUCTION

Collaborative filtering (CF) recommendation models play a pivotal role in online services in increasing traffic and promoting sales. They are widely adopted by various e-commerce and consumer-oriented services to recommend a whole range of items, including products, music, movies, news articles, friends, restaurants, and various others. Their key assumption is that users who shared similar preferences in the past will likely agree in the future as well. Then, from an algorithmic point of view, these models keep track of users' historical behavior data (users' interactions and stated preferences) and find similar behavioral patterns to offer personalized new suggestions. Three decades of academic and industrial research on recommender systems (RS), have resulted in many variations of CF with demonstrated success in many real-world applications [18, 51].

Notwithstanding their great achievement, CF can expose the underlying system subject to threats from a robustness standpoint. Their vulnerability relates to several factors: (i) the open nature of these systems and their reliance on the so-called "wisdom of the crowd" (i.e., what others think toward an item or group of items to compute recommendations), (ii) their inherent inability to distinguish genuine user profiles from fake ones. These leave space for a third party to manipulate recommendation outcomes offered to normal users through attack profile injection attack (aka shilling attack) [19]. The motivation for such attacks is often malicious, e.g., personal gain, market penetration, and even for causing mischief on an underlying system. For instance, fake social media accounts could be created by an adversarial party to spread news articles about a specific party or belief system or false reviews could be provided about a product to promote (push attack) or demote (nuke attack) awareness and reach to that product, and so engagement with new consumers.

Previous studies [5, 28] have shown that a surprisingly modest number of fake profile attacks (around 3%) mounted on CF models are sufficient to manipulate a prediction shift up to 30%, signifying the impact that such handcrafted attack profiles can have on faulting recommendation results. As CF models assist users in many decision-making and mission-critical tasks, such non-robust measures could have far-reaching consequences, impacting peoples' lives and leaving usability of RS questionable.

Prior researches in shilling attacks cover three main directions: attack designs, detection algorithms, and defense strategies. We provide a brief description of each direction here. Attacks toward CF models were originally introduced in [44]. This research paved the way towards a formal definition of several attack strategies such as random, average, popular, bandwagon, and love-hate attacks [41]. Majority of these attacks, and the ones introduced later,

were not algorithm-agnostic and, depending on the task, they assumed some knowledge of the attacker on the confronted recommendation model (e.g., memory-based, model-based), recommendation outputs, properties of items (e.g., rating mean and entropy [32]) and even users under attack (e.g., group of users [10]). Detection strategies aim to identify between attack profiles mounted on the system from genuine profiles, e.g., by leveraging users' properties [4, 55]. Robust algorithms attempt to diminish the influence of known attacks mostly by implementing matrix factorization-based models [53], or making use of users, and items, trust information [5].

To date, many research articles have been written on the subject of shilling attacks on recommender models, which can be classified in one of the dimensions outlined above. A common characteristic of this literature is that the analysis of experiments orientates to "*win-lose*" predicting scenarios, trying to find an answer to questions such as "*Which attack models impact more the performance of certain recommendation models?*", "*Which amount of knowledge on a specific recommendation-model is required for specific attack A to influence recommendation algorithm B?*". Little effort has been made on providing an explanatory study on which dataset characteristics impact the effectiveness of attacks. For instance, it is known that RS performance can be affected based on the *sparsity of the dataset*, meaning that a highly dense dataset can impact the quality of CF models in ways that are different from a highly sparse dataset. However, whether this data characteristic can have a similar impact on the effectiveness of the profile injection attack remains far more under-researched.

Motivated by this observation, the work at hand puts its attention outside the subject of *proposing another attack strategy against recommendation model*, instead it focuses on the central question "*Given popular shilling attack types and CF models already recognized by the community, which dataset characteristics can explain an observed change in the performance of recommendation?*"

We present a systematic and in-depth analysis of the impact of dataset characteristics for the robustness of CF by utilizing a regression-based model. Via a large-scale experiment on three domains, we evaluate how three classes of data characteristics —rating structure, rating value, and rating distribution— may influence the robustness of CF algorithms measured in terms of stability metrics.

Our work most closely resembles the work done by Adomavicius and Zhang [2], which studies the influence of rating data characteristics on the recommendation performance of popular collaborative RS. Their work differs from ours because we utilize the explanatory model to explain the variation in *robustness* of CF models (or effectiveness of attack strategies) with respect to data characteristics.

The contributions of this work are multi-fold:

(1) *Modeling*: we present a systematic, in-depth exploratory research and analysis of the impact of dataset characteristics on the robustness performance of popular CF models subjected to famous shilling attack strategies. To investigate the relationship between data characteristics and the robustness of CF models, we use *regression-based explanatory* modeling.

(2) *Data characteristics*: unlike prior works on shilling attacks [5, 36], we validate the correlation between data characteristics and attack effectiveness on a *suite of data characteristics* extracted from the user-rating matrix (URM), going beyond well-recognized properties such as data sparsity.

(3) *Experiments*: we conduct *extensive* empirical analysis on six popular attack strategies against three well-known CF models across three real-world datasets. In total 97200 attack simulation are conducted to solve the coefficient related to different explanatory regression problems (see Section 4). We rely on a statistical significance test with informed *p-value* to validate the hypothesis if the demonstrated set of data characteristics have an impact on the final model output.

2 BACKGROUND TECHNOLOGIES

A recommendation problem can be defined as finding a utility function to automatically predict how much a user will like an item that is unknown to her.

Definition 1 (Recommendation Problem). *Let $\mathcal{U}$ and $\mathcal{I}$ denote a set of users and items in a system, respectively. Given a utility function $g : \mathcal{U} \times \mathcal{I} \rightarrow \mathbb{R}$ a **Recommendation Problem** is defined as*

$$\forall u \in \mathcal{U}, i'_u = \underset{i \in \mathcal{I}}{\operatorname{argmax}}\, g(u, i)$$

with i'_u being an item not enjoyed by u before. We further define $R \in \mathbb{R}^{|\mathcal{U}| \times |\mathcal{I}|}$ as the user-item rating matrix (URM), where each entity $r_{ui} \in \mathbb{R}$ represents a continues-valued rating assigned by user $u \in \mathcal{U}$ to item $i \in \mathcal{I}$. We denote with $\mathcal{K}$ the set of (u, i) pairs for which r_{ui} is known and therefore $|\mathcal{K}|$ represents the total number of ratings.

The solution to a recommendation problem heavily depends on the selected utility function g —usually, but not necessarily, a machine learning model— and on the information encoded in the matrix R as well as on the way ratings are distributed within the dataset represented by R. The open nature of URM makes the recommendation problem vulnerable to the injection of malicious users [19]. This injection, named shilling attack, is defined below.

Definition 2 (Shilling Profile). *Let $\mathcal{I}_S$ denote the selected item set, $\mathcal{I}_F$ the filler set, $\mathcal{I}_\phi$ the unrated-item set, $\mathcal{I}_T$ the target item set, and given a Recommendation Problem, a **Shilling Profile** ($\mathcal{SP}$) is defined as:*

$$\mathcal{SP} = \mathcal{I}_S + \mathcal{I}_F + \mathcal{I}_\phi + \mathcal{I}_T \tag{1}$$

where $\mathcal{I}_S$ contains items identified by the attacker to exploit the owned knowledge to maximize the effectiveness of the attack, $\mathcal{I}_F$ holds randomly selected items for which rating scores are assigned to make the attack imperceptible. $\mathcal{I}_\phi$ includes items without ratings in the fake user profile, and $\mathcal{I}_T$ is the item is to push or nuke. The $\mathcal{SP}$ composition varies based on attack strategies.

Table 1 shows the state-of-the-art explored attack strategies.

3 PROBLEM FORMALIZATION

In this section, we describe core contribution of this work, the explanatory framework proposed to investigate the impact of data characteristics on attacks' effectiveness.

3.1 Independent Variables

This work focuses only on rating-based CF models as recommendation models confronted to/exposed with shilling attacks. CF uses only the URM to compute recommendations. For this reason, all

Table 1: Attack strategies and their profile composition (*push* attacks).

Attack Type	I_S		I_F		I_ϕ	I_T
	Items	Rating	Items	Ratings		
Random [32]	$\emptyset$		$\frac{\sum_{u \in U} \lvert I_u \rvert}{\lvert U \rvert} - 1$	$rnd(N(\mu, \sigma^2))$	$I - I_F$	max
Love-Hate [36]	$\emptyset$		$\frac{\sum_{u \in U} \lvert I_u \rvert}{\lvert U \rvert} - 1$	min	$I - I_F$	max
Bandwagon [43]	$(\frac{\sum_{u \in U} \lvert I_u \rvert}{\lvert U \rvert})/2 - 1$	max	$(\frac{\sum_{u \in U} \lvert I_u \rvert}{\lvert U \rvert})/2$	$rnd(N(\mu, \sigma^2))$	$I - I_S - I_F$	max
Popular [45]	$\frac{\sum_{u \in U} \lvert I_u \rvert}{\lvert U \rvert} - 1$	min **if** $\mu_f < \mu$ **else** $min + 1$	$\emptyset$		$I - I_S$	max
Average [32]	$\emptyset$		$\frac{\sum_{u \in U} \lvert I_u \rvert}{\lvert U \rvert} - 1$	$rnd(N(\mu_f, \sigma_f^2))$	$I - I_F$	max
P. Knowledge [42]	$\frac{\sum_{u \in U} \lvert I_u \rvert}{\lvert U \rvert} - 1$	max	$\emptyset$		$I - I_S$	max

where (μ, σ) are the dataset average rating and rating variance, (μ_f, σ_f) are the filler item i_f rating average and variance, and min and max are the minimum and maximum rating value. rnd function generates one integer (i.e., rating) from a discrete uniform distribution.

the IVs representing dataset characteristics presented in this work are related to URM characteristics and are inspired/taken from [2]. We categorize these features according to (i) structure of URM, (ii) rating frequency of URM and, (iii) rating values of URM.

3.1.1 IVs based on URM structure. The IVs that describe the structure of URM are *SpaceSize*, *Shape*, and *Sparsity*. Computing these IVs requires knowledge about the number of real users $|\mathcal{U}|$, the number of real items $|\mathcal{I}|$, and the number of ratings $|\mathcal{K}|$.

Definition 3 ($SpaceSize_{log}$). *Given a Recommendation Problem, we define $SpaceSize_{log}$ as in the following.*

$$x_1 = \log_{10}(\frac{|\mathcal{U}| \cdot |\mathcal{I}|)}{sc}) \tag{2}$$

The scaling factor (sc) is a parameter that can be set to limit the range of $|\mathcal{U}| \cdot |\mathcal{I}|$ into a small range. The $\log_{10}$ operation normalizes the distribution of this variable.

$SpaceSize_{log}$ is a variant of the original *SpaceSize*, and it is introduced in [2]. It is noteworthy that it may affect the performance of the underlying CF model and the mounted shilling profiles. For instance, under comparable density values, higher URM SpaceSize might imply a bigger chance of finding similar neighbor users or items. Therefore, as both attack and recommendation models rely on the identification of like-minded users (neighbor users) or similarly rated items (neighbor items), we deem URM SpaceSize to be an impactful dataset characteristic on evaluating the performance of shilling attacks applied on CF models. For instance, for the small dataset generated in this work during the simulations, typical values were in the range of *thousands* to *millions* with a wide variety. All of this can impact the accuracy of the regression model's coefficients calculated. Similar to [2], we set $sc = 1000$ in this work.

Definition 4 ($Shape_{log}$). *Given a Recommendation Problem, we define $Shape_{log}$ as in the following.*

$$x_2 = \log_{10}(\frac{|\mathcal{U}|}{|\mathcal{I}|}). \tag{3}$$

$Shape_{log}$ can impact the effectiveness of shilling profile injection attacks. For example, in domains where the $Shape\,(URM) << 1$ (i.e., $|\mathcal{U}| << |\mathcal{I}|$) there are more candidate neighbor users than candidate neighbor items for memory-based CF models. This situation might work in the advantage of user-based CF than item-based CF. Moreover, under a similar number of ratings, changing the shape implies changing the average number of ratings per item $|\mathcal{R}| \div |\mathcal{I}|$. We conjecture that this circumstance may impact the performance of CF under attacks since the construction of the shilling profile is mainly based on altering the popularity of targeted items. The logarithm transformation in $Shape_{log}$ is applied to normalize the $|\mathcal{U}| \div |\mathcal{I}|$ distribution (e.g., the minimum and maximum values of shape in MovieLens dataset are (0.366, 30.039) before the log transformation, (-0,437, 1,478) after the transformation.).

Definition 5 ($Density_{log}$). *Given a Recommendation Problem, we define $Density_{log}$ as in the following.*

$$x_3 = \log_{10}(\frac{|\mathcal{K}|}{|\mathcal{U}| \times |\mathcal{I}|}). \tag{4}$$

Data sparsity[1], which relates to data density, according to $density = 1 - sparsity$ is a well-recognized issue in the community of RS. Data sparsity refers to situations where the fraction of unrated items significantly exceeds the fraction of rated ones, and not a sufficient information is available for CF models to be trained and to make predictions. The reasons for sparsity can be described by the fact judging is a cognitively expensive task in nature; furthermore, in some domains the size of the catalog can be huge with many unpopular or unseen items, and finally, the extreme case of cold-start problem, where new users or items register in the system without ratings [39]. Data sparsity can harm the performance of CF in different ways. For instance, it can reduce the chance of discovering neighbors in memory-based CF because the possibility of having co-rated items is lower in sparse URM. Model-based CF can suffer significantly from the sparsity problem to train [13]. A large amount of research focuses on investigating and alleviating the sparsity problem in CF recommender systems by proposing various solutions [9, 22, 26]. In [15], the authors identify a potential impact of dataset density on the effectiveness of shilling attacks.

3.1.2 IVs based on rating frequency of the URM. Another important characteristic of a URM is the rating frequency distribution. The idea is that in many real applications, a small number of items receive a large number of ratings (short heads or popular items), while a large number receive low or few feedbacks (long tails), causing the rating distribution to be skewed. It turns out that the commercial profit from recommending long-tail items is more significant than short-head items [6]. However, these long-tail items have less chance to be recommended since they have less historical feedbacks [40]. We

[1]We describe data sparsity since it is a more common term in the literature of RS, but everything mentioned about sparsity related to density in a reverse manner.

examine this URM characteristic because in a very skewed situation (e.g., few items with a large number of ratings), the possibility to alter recommendations could be very low because popular items will be recommended by themselves.

Definition 6 ($Gini_{item}$, $Gini_{user}$). *Given a Recommendation Problem, let $|\mathcal{K}_i|$ be to the number of ratings received by the item i, let $|\mathcal{K}_u|$ be to the number of ratings given by the user u, we define $Gini_{item}$ and $Gini_{user}$ respectively as in the following:*

$$x_4 = 1 - 2\sum_{i=1}^{|\mathcal{I}|}\left(\frac{|\mathcal{I}| + 1 - i}{|\mathcal{I}| + 1}\right) \times \left(\frac{|\mathcal{K}_i|}{|\mathcal{K}|}\right) \tag{5}$$

$$x_5 = 1 - 2\sum_{u=1}^{|\mathcal{U}|}\left(\frac{|\mathcal{U}| + 1 - u}{|\mathcal{U}| + 1}\right) \times \left(\frac{|\mathcal{K}_u|}{|\mathcal{K}|}\right) \tag{6}$$

We use the Gini coefficient that measures the concentration of items, or users', ratings to capture the rating frequency distribution. The equal popularity (e.g., all users give the same number of ratings) is represented with the value of the Gini coefficients to 0, while the total inequality (e.g., only one user has given all ratings) is represented with the value to 1.

3.1.3 IVs based on rating values of the URM. While the previous dataset characteristics relate to the structure of the URM and the distribution of ratings assigned to items, they disregard the actual values of the ratings themselves. The most common statistics representing rating values are *rating mean* and *rating variance*. Similar to [2], we disregard the overall rating means because most CF models involve a pre-processing step that centralizes the rating around the mean rating value, effectively removing its effect. Therefore, we study the effect of rating variance by investigating the following measure.

Definition 7 (Std_{rating}). *Given a Recommendation Problem, let $\bar{r}$ bet the global mean value of the ratings in the URM, we define Std_{rating} as in the following.*

$$x_6 = \sqrt{\frac{\sum_{i=1}^{|\mathcal{K}|}(r_i - \bar{r})^2}{|\mathcal{K}| - 1}} \tag{7}$$

We investigate the possible influence of Std_{rating} on the robustness analysis motivated by the connection between high rating variance and recommendation performance claimed by Herlocker et al. in [24] and the linear and negative impact on the accuracy performance reported in [2].

3.2 Dependent Variables

The dependent variable (DV) represents the effectiveness of the attack on RS.

Definition 8 (Incremental Overall Hit Ratio). *Let HR@k be the metric value before the attack, $\hat{HR}$@k the value after an attack, the Incremental Overall Hit Ratio is defined as*

$$\Delta_{HR@k} = \hat{HR}@k - HR@k \tag{8}$$

in which

$$HR@k(\mathcal{I}_T, \mathcal{U}_T) = \frac{\sum_{i_t \in \mathcal{I}_T} hit(i_t, \mathcal{U}_T)}{|\mathcal{I}_T|} \tag{9}$$

where $hit(i_t, \mathcal{U}_T)$ is defined as the fraction of users in $\mathcal{U}_T$ for which item i_t is present in the top-k recommendations [3].

Evaluation metric for shilling attack effectiveness can be classified according to: *prediction accuracy* and *stability*. Recommendation accuracy measures if the actual rating predicated by the recommendation model was altered due to the attack. Recommendation stability measures if the recommendation model recommends different products due to the attack irrespective of their actual rating value [42]. The Incremental Overall Hit Ratio is a stability metric introduced for the explanatory modeling analysis. We have also performed the experiments for predictive accuracy (using the metric *prediction shift* [3]). However, due to space limitations, we plan to present them in an extended version of this work.

3.3 Explanatory Framework

Statistical models can be used for two purposes: (i) explanatory modeling (EM), and (ii) predictive modeling (PM). EM seeks to test the "causal hypothesis" in a theoretical construct, which means if a set of underlying effects measured by $\mathbf{X}$ are the cause for an underlying effect measured by y. The goal of PM, on the other hand, is to predict "new" or "future" observations given their input values ($\mathbf{X}$) [50]. Furthermore, (i), the model is carefully constructed to support "interpretability" of the relationship between $\mathbf{X}$ and y, while in (ii) the model is "constructed from data". Prior works on shilling attacks have been largely focused on a PM approach to improve the predictive performance of attacks [4, 19, 42]. Instead, in this work, we choose a different approach and adopt an EM approach to test the causal hypothesis between underlying factors representing data characteristics ($\mathbf{X}$) and the underlying effect represented by attack performance (y). Grounded on [2], we use a formal work based on the *regression model* as a classical interpretable EM function.

Given a dataset d for a particular domain (music, business, and movie), an attack strategy as identified in Table 1, a CF recommendation model g (item-based CF, user-based, and SVD), the goal is to test the hypothesis whether the factors related to dataset characteristics measured by $\mathbf{X}$ (independent variables) can explain the effect on the RS performance measured by y (dependent variable). In our case, the dependent variable (as we will see in Section 3.2) is represented by a metric able to measure the effects of a shilling attack. A regression model is used to model the causal relationship in the presented framework

$$y_i = \epsilon_i + \theta_0 + \sum_{d=1}^{D-1}\theta_d x_{d,i} + \sum_{c=1}^{C}\theta_c x_{c,i} \tag{10}$$

in which C is the number of data characteristic factors, θ_c is the regression coefficient of the c-th independent variable (IV) and $x_{c,i} \in \mathbb{R}$ represents the value of the c-th independent variable for the i-th training example, and $y_i \in \mathbb{R}$ is the measurement corresponding to i-th training example (the measured dependent variable). $\sum_{d=1}^{D-1}\theta_d x_{d,i}$ is a dummy term introduced only for the between-datasets analysis (cf. Section 4.2.2), whose role is to capture information about dataset variation, where D is the number of the datasets in the across datasets study, $x_{d,i}$ is a binary (0, 1) dummy variable representing whether sample i belongs to the dataset d or not, and θ_d is the regression coefficient associated with the dataset d (see [2] for more information).

In a more compact way, we have

$$\boldsymbol{y} = \boldsymbol{\epsilon} + \theta_0 + \boldsymbol{\theta}_d\mathbf{X}_d + \boldsymbol{\theta}_c\mathbf{X}_c \tag{11}$$

where under mean-centered data, θ_0 represents the expected value of $\boldsymbol{y}$ (the performance metric under analysis), $\boldsymbol{\theta}_d = [\theta_1, \theta_2, ..., \theta_{D-1}]$ is the vector containing coefficients of the dummy variable $\mathbf{X}_d$ related to the dataset of each training example, $\boldsymbol{\theta}_c = [\theta_1, \theta_2, ..., \theta_C]$ is the vector of the regression coefficient associated with the IVs, and $\mathbf{X}_c$ is the matrix containing the independent variables values (data characteristic values computed based on URM). We apply the regression framework to address two explanatory analysis: (i) within-dataset analysis and, (ii) between-dataset analysis presented in the following.

Within-dataset analysis: The within-dataset analysis addresses the task of analyzing the impact of URM characteristics for each combination of datasets, type of attacks, and recommendation models. The regression coefficients in the linear explanatory model are computed under the ordinary least squares (OLS) optimization model. The OLS minimization problem is defined as

$$(\theta_0^*, \boldsymbol{\theta}_c^*) = \min_{\theta_0, \boldsymbol{\theta}_c} \frac{1}{2} \|\boldsymbol{y} - \theta_0 - \boldsymbol{\theta}_c \mathbf{X}_c\|_2^2 \tag{12}$$

Section 4.2.1 analyzes the regression model results for the within-dataset analysis.

Between-dataset analysis: We extend the within-dataset analysis explanatory model to a between-dataset analysis with the goal to examine a domain-independent perspective about the impact of data characteristics on the model output. The minimization problem is defined as:

$$(\theta_0^*, \boldsymbol{\theta}_d^*, \boldsymbol{\theta}_c^*) = \min_{\theta_0, \boldsymbol{\theta}_d, \boldsymbol{\theta}_c} \frac{1}{2} \|\boldsymbol{y} - \theta_0 - \boldsymbol{\theta}_d \mathbf{X}_d - \boldsymbol{\theta}_c \mathbf{X}_c\|_2^2 \tag{13}$$

where the constant term θ_0 represents the reference dataset (in our experimental evaluation we consider ML-20M) and the dummy term $\boldsymbol{\theta}_d \mathbf{X}_d$ provides a binding to the other $D-1$ datasets (i.e., Yelp and LFM-1b in our experiments) [2]. Section 4.2.2 presents the regression model results for the between-dataset analysis.

4 EXPERIMENTS

In this section, we present experimental settings and the discussion of the results.

4.1 Experimental Settings

Datasets: We conducted shilling attacks against CF models on three real-world datasets, ML-20M [20], Yelp [21], and LFM-1b [49]. The datasets have properties that are considerably different from each other — for instance, considering the domains and structural properties of the dataset (see Table 2)—, effectively allowing us to analyze and validate the experimental results under a diverse set of data characteristics.

The ML-20M [20] dataset is a 20 million-sized version of MovieLens (ML) dataset. Each item (movie) is rated on a 0-5 Likert scales. ML is among the most commonly adopted datasets for the offline evaluation of RS, and ML-20M is the largest stable version among different variations of the dataset.

The Yelp [21] dataset contains users' ratings, reviews, and check-in on businesses (e.g., restaurants) collected from Yelp.com. We used a pre-processed version of the dataset provided by [21] that contains only integer rating values in the range (1, 5) assigned by users to businesses.

The LFM-1b [49] dataset is a music domain dataset containing more than one billion of listening events (e.g., playing a track of an artist) fetched from January 2013 to August 2014 from the Last.fm online music system. LFM-1b provides implicit feedback, user-artist play counts, which were converted to explicit feedback into the range (1, 5), following the procedure proposed in [33].

Table 2: The statistics related to the dataset used in this work.

Dataset	$\|U\|$	$\|I\|$	$\|R\|$	$density$
ML-20M	138,493	26,744	20,000,263	0.0054
Yelp	25,677	25,778	705,994	0.0010
LFM-1b	120,175	521,232	25,285,767	0.0004

Compared CF Recommendation Models: We studied the impact of data properties on the effectiveness of the attacks against the following CF recommendation models:

User-*k*NN [29] computes the unknown preference score $\hat{\mathbf{r}}_{ui}$ for user u and item i as an aggregate of the ratings of the users who have rated item i and are most similar to user u.

$$\hat{r}_{ui} = b_{ui} + \frac{\sum_{v \in \mathcal{U}_i^k(u)} \text{dist}(u, v) \cdot (r_{vi} - b_{vi})}{\sum_{v \in \mathcal{U}_i^k(u)} \text{dist}(u, v)} \tag{14}$$

where $b_{ui} = \mu + b_u + b_i$, and μ, b_u, b_i respectively are the overall average rating, the observed bias of user u and item i

Item-*k*NN [29] calculates $\hat{\mathbf{r}}_{ui}$ as an aggregate of the ratings of the items, which are most similar to item i.

$$\hat{r}_{ui} = b_{ui} + \frac{\sum_{j \in \mathcal{I}_u^k(i)} \text{dist}(i, j) \cdot (r_{uj} - b_{uj})}{\sum_{j \in \mathcal{I}_u^k(i)} \text{dist}(i, j)} \tag{15}$$

For the both above *k*NN approaches, we used the formulations that adjust user and item effects — systematic inclinations for some users to provide higher ratings than others, and for some items to collect ratings higher than others items — subtracting biases (i.e., b_{vi}, b_{uj}) from each rating [30]. We set number of neighbors k equal to 40, and we used the pearson correlation as *dist* function.

SVD [31] uses matrix factorization (MF) model as the core predictor that factorizes the user-item rating matrix (URM) to learn users' preferences by fitting the previously observed interactions. The predicted rating is computed as $\hat{r}_{ui} = b_{ui} + \mathbf{q}_i^T \mathbf{p}_u$, where $\mathbf{q}_i \in \mathbb{R}^H$ and $\mathbf{p}_u \in \mathbb{R}^H$ are the item and user latent vectors learned by the model. We set the number of hidden factors (H) to 100, the default value in [27].

Shilling attacks strategies: We use six popular shilling attack strategies in order to study the impact of data characteristics on the performance of each attack independently: Random attack, Love-hate attack, Bandwagon attack, Popular attack, Average attack, Perfect-knowledge attack. We provide the technical description of each attack in Table 1.

Sample generation: Based on the regression-based explanatory model formalized by Eqs. 12 and 13, the goal is to solve regression model coefficients using characteristics generated from various datasets with different structures and content values. Obviously, the scale and diversity of datasets can have a large impact on the accuracy of coefficients computed and more importantly on the generalizability of final findings. Toward this aim, motivated by the approach presented in [2], we adopt a sample (i.e., dataset)

generation strategy where for a given original dataset (URM), the goal is to generate $\mathcal{N}$ different samples i.e., (smaller dataset urm_n) with different characteristics. The sampling procedure is specified in Algorithm 1.

Algorithm 1 Sample generation procedure

Input: URM
Results: $\mathcal{N}$ sub-datasets (urm_n)
$n \leftarrow 1$
while $n \leq \mathcal{N}$ **do**
  Random shuffle the row of the URM
  $num_{users} \leftarrow rnd([100, 2500])$
  $num_{items} \leftarrow rnd([100, 2500])$
  $urm_n \leftarrow$ Selection of num_{users}, num_{items} from URM
  if $density(urm_n) \in [0.0005, 0.01]$ **then**
    $n \leftarrow n + 1$

For a given recommendation model (User-kNN, Item-kNN, and SVD), an attack strategy (six attack strategies), an attack size (1%m 2.5%, and 5%), and each dataset (ML-20M, Yelp, and LFM-1b), we generate $\mathcal{N} = 600$ samples (sub-datasets) resulting in a total number of 162 study cases (i.e., 54 for each attack size) obtained by performing 97200 attack simulation experiments. We force the densities of the generated urm_n to be in a predefined range of $[0.0005, 0.01]$) to obtain realistic density values. Table 3 summarize the statistics related to each IV (data characteristics) for the 600 generated data-samples.

Table 3: Statistics of Independent Variables ($\mathcal{N} = 600$)

Data	IVs	Min	Max	Mean	σ
ML-20M	$SpaceSize_{log}$	2K	2M	594K	537K
	$Shape_{log}$	0.366	30.039	2.985	2.773
	$Density_{log}$	0.010	0.070	0.019	0.007
	$Gini_{user}$	0.266	0.631	0.547	0.059
	$Gini_{item}$	0.528	0.831	0.737	0.052
	Std_{rating}	0.902	1.183	1.050	0.030
Yelp	$SpaceSize_{log}$	240	3M	618K	695K
	$Shape_{log}$	0.331	3.509	1.225	0.510
	$Density_{log}$	0.002	0.071	0.007	0.007
	$Gini_{user}$	0.052	0.563	0.390	0.089
	$Gini_{item}$	0.068	0.634	0.432	0.090
	Std_{rating}	0.988	1.299	1.151	0.035
LFM-1b	$SpaceSize_{log}$	168	589K	98K	120K
	$Shape_{log}$	0.800	9.685	2.444	1.026
	$Density_{log}$	0.004	0.085	0.016	0.014
	$Gini_{user}$	-0.000	0.422	0.255	0.088
	$Gini_{item}$	0.121	0.819	0.590	0.124
	Std_{rating}	0.577	1.204	0.950	0.077

K = thousand, M = milion

Parameters and settings: Before building the regression model, the dataset characteristics are mean-centered. We set the recommendation list $k = 10$ for all experiments. We execute experiments considering three quantities of added fake users equal to 1%, 2.5%, and 5% of the number of users in each data sample. However, since larger attack size is impactful in every condition, it is less meaningful to analyze the impact of data characteristics when attacks are consistently effective in all experimental cases. Therefore, we focus our attention only on the smaller size of injected profiles (1%). Finally, we select the number of attacked items at 0.05% of the number of items in each data sample. To ensure a general analysis of the framework, inspired by [5], we randomly select the same number of target items from all items' popularity quartiles.

Explanatory regression model evaluation: While prior research in shilling attacks on RS largely focus on predictive modeling tasks, i.e., to assess the effectiveness of attacks measured in terms of stability (e.g., $HR@k$) and predictive accuracy [3] metrics, in this work we build an explanatory statistical model with the goal to *validate the hypothesis* if there exists an underlying relationship between data characteristics and the explanatory model output $\Delta_{HR@k}$. After validating this hypothesis, our secondary goal is to compute the *significance* and *directionality* of this relationship. Thus, the evaluation metrics presented here aim toward assessing the outcome of the explanatory model:

Coefficient of determination (R^2). R^2 is a common metric in statistics to measure how well the data fit a (linear) regression model [46]. R^2 represents the proportion of variation in the DV that is explained by the IVs. R^2 values range from 0 to 1, 1 means that the DV is completely explained by IVs, while 0 indicates that the model explains none of the variability in the output. For instance, an R^2 of 0.58 means that IVs explains the 58% of variations in the DV.[2]

Significance of the measured regression coefficients. The *p-value* for each regression coefficient tests the null-hypothesis that the coefficient is equal to 0 (i.e., the IV does not influence the DV). A small *p-value* ($p < 0.05$) indicates that there is enough evidence to reject the null-hypothesis (i.e., an effect) and we can assert that the findings are "statistically significant". To help the reader, in the result Tables 4 and 5, we use the signs * ($p < 0.05$), ** ($p < 0.01$) and *** ($p < 0.001$) to report which of the coefficient computed are statistically significant. It should be noted that we rely only on statistically significant results in presenting a discussion about the results and drawing the final conclusions.

Directionality of the measured regression coefficients. The sign of the regression coefficient indicates whether there is a positive relation between variation on an IV and DV or a negative relationship. This information might be used by a system designer to understand and anticipate potential variations in the robustness performance against shilling attacks of the maintained recommender system.

Evaluation Questions: To better understand the merits of the proposed explanatory framework, our aim is to answer the following evaluation questions through the course of experiments:

Q1: Is there an underlying relationship between the presented set of dataset characteristics (IVs) computed from the URM (cf. Section 3.1) and the effectiveness of shilling attack on CF models (DV) measured in terms of $\Delta_{HR@k}$?
Q2: How *significant* is the impact of each IV on the effectiveness of shilling attacks measured in terms of $\Delta_{HR@k}$? What is the *directionality* of this impact (positive or negative)?
Q3: Do the demonstrated IVs present a consistent behavior when data samples are combined from datasets of all domains (i.e., a domain-independent behavior)?

[2] In explaining the results presented in Table 4 and 5, we rely on ($adj.R^2$) a modified version of R^2, which unlike the latter is not affected by new features added rather if the new feature truly contributes to the overall performance.

Table 4: Regression results for the within dataset analysis (attack size 1%).

$\Delta_{HR@10}$		User-*k*NN			Item-*k*NN			SVD		
		ML-20M	Yelp	LFM-1b	ML-20M	Yelp	LFM-1b	ML-20M	Yelp	LFM-1b
Random	$R^2(adj.R^2)$	0.761(0.758)	0.838(0.835)	0.673(0.668)	0.820(0.818)	0.815(0.812)	0.666(0.662)	0.843(0.841)	0.908(0.907)	0.790(0.788)
	$Constant$	**.179*****	**.609*****	**.717*****	.262***	.610***	.715***	.482***	.524***	.688***
	$SpaceSize_{log}$	-0.063***	.041	-0.629***	.008	.003	-0.520***	.040*	.368***	-0.368***
	$Shape_{log}$	.184***	.248***	.288*	.139***	.198***	.125	.207***	.275***	.192
	$Density_{log}$	-0.189***	-0.316*	-1.546***	-0.174***	-0.376**	-1.366***	-0.274***	.393***	-1.047***
	$Gini_{users}$	.277	-0.012	1.901***	-0.223	.030	.891	.178	-0.660**	.988*
	$Gini_{item}$	-0.102	-0.485	1.753***	-0.305	-0.241	1.784***	.102	-1.270***	1.355***
	Std_{rating}	-0.072	.287	-0.152	-0.120	.326	.012	-0.240	.311*	-0.108
Love-Hate	$R^2(adj.R^2)$	0.806(0.803)	0.839(0.837)	0.673(0.668)	0.841(0.839)	0.822(0.820)	0.665(0.661)	0.825(0.823)	0.911(0.910)	0.789(0.787)
	$Constant$	.267***	.657***	.717***	.419***	.662***	.716***	.655***	.578***	.688***
	$SpaceSize_{log}$	-0.027*	.042	-0.628***	.125***	.028	-0.506***	.073***	.393***	-0.364***
	$Shape_{log}$	.209***	.131*	.287*	.103***	.065	.105	.059***	.105*	.194
	$Density_{log}$	-0.198***	-0.290*	-1.544***	-0.071*	-0.316*	-1.337***	-0.209***	.434***	-1.044***
	$Gini_{users}$	.347	.114	1.896***	-0.852***	-0.002	.831	-0.231	-0.920***	.972*
	$Gini_{item}$	-0.430	-0.150	1.754***	-0.583**	.043	1.806***	.985***	-0.764*	1.309***
	Std_{rating}	-0.179	.239	-0.151	-0.188	.259	.022	-0.168	.278	-0.073
Bandwagon	$R^2(adj.R^2)$	0.777(0.774)	0.835(0.833)	0.673(0.668)	0.818(0.816)	0.813(0.810)	0.665(0.661)	0.841(0.839)	0.914(0.912)	0.786(0.784)
	$Constant$	.180***	.607***	.717***	.244***	.609***	.715***	.435***	.522***	.689***
	$SpaceSize_{log}$	-0.068***	.040	-0.635***	-0.015	-0.002	-0.514***	-0.006	.372***	-0.366***
	$Shape_{log}$	.190***	.266***	.293*	.145***	.212***	.116	.244***	.278***	.206*
	$Density_{log}$	-0.188***	-0.305*	-1.559***	-0.192***	-0.382**	-1.354***	-0.314***	.401***	-1.047***
	$Gini_{users}$	.342	.110	1.919***	-0.080	.104	.869	.602***	-0.680**	.976*
	$Gini_{item}$	-0.041	-0.483	1.755***	-0.158	-0.251	1.797***	.268	-1.278***	1.276***
	Std_{rating}	-0.036	.284	-0.151	-0.087	.315	.016	-0.290	.321*	-0.066
Popular	$R^2(adj.R^2)$	0.860(**0.859**)	0.787(**0.784**)	0.670(**0.665**)	0.913(**0.912**)	0.762(**0.759**)	0.660(**0.655**)	0.774(**0.772**)	0.866(**0.864**)	0.784(**0.781**)
	$Constant$	.589***	.810***	.724***	.537***	.788***	.705***	.724***	.775***	.694***
	$SpaceSize_{log}$	-0.020	-0.411***	-0.617***	.099***	-0.441***	-0.517***	.009	-0.238**	-0.391***
	$Shape_{log}$	.187***	.028	.268*	.176***	.084	.118	.084***	.041	.210*
	$Density_{log}$	-0.335***	-1.175***	-1.521***	-0.162***	-1.261***	-1.361***	-0.337***	-0.898***	-1.092***
	$Gini_{users}$	.225	1.050**	1.872***	-0.318	1.129**	.879	-0.055	.199	1.082*
	$Gini_{item}$	.491	1.735***	1.764***	-0.225	1.305***	1.715***	1.346***	1.195**	1.347***
	Std_{rating}	-0.182	.002	-0.129	-0.353*	.049	.017	-0.043	.237	-0.062
Average	$R^2(adj.R^2)$	0.759(0.756)	0.831(0.829)	0.673(0.668)	0.819(0.816)	0.813(0.811)	0.666(0.661)	0.845(0.843)	0.910(0.909)	0.790(0.788)
	$Constant$	.187***	.609***	.717***	.276***	.608***	.715***	.502***	.523***	.690***
	$SpaceSize_{log}$	-0.063***	.048	-0.632***	.018	.010	-0.513***	.046**	.373***	-0.339***
	$Shape_{log}$	.182***	.260***	.291*	.136***	.201***	.114	.189***	.273***	.167
	$Density_{log}$	-0.189***	-0.290*	-1.553***	-0.162***	-0.359**	-1.352***	-0.271***	.405***	-0.991***
	$Gini_{users}$	.296	.074	1.907***	-0.265	.028	.857	.095	-0.652**	.833*
	$Gini_{item}$	-0.072	-0.522	1.755***	-0.284	-0.243	1.796***	.258	-1.267***	1.317***
	Std_{rating}	-0.065	.299	-0.150	-0.114	.312	.019	-0.242	.322*	-0.079
Perfect Knowledge	$R^2(adj.R^2)$	0.790(0.787)	0.836(0.834)	0.676(0.671)	0.828(0.826)	0.823(0.821)	0.670(0.666)	0.847(0.845)	0.914(0.913)	0.793(0.790)
	$Constant$	.267***	.608***	.717***	.275***	.603***	.711***	.479***	.519***	.688***
	$SpaceSize_{log}$	-0.039**	.071	-0.609***	.006	.027	-0.504***	.034*	.382***	-0.358***
	$Shape_{log}$	.187***	.256***	.271*	.137***	.247***	.104	.207***	.275***	.198*
	$Density_{log}$	-0.139***	-0.247	-1.508***	-0.181***	-0.335**	-1.337***	-0.277***	.427***	-1.032***
	$Gini_{users}$	.399	-0.067	1.815***	-0.184	.093	.789	.240	-0.714**	.969*
	$Gini_{item}$	.270	-0.582	1.746***	-0.226	-0.506	1.771***	.216	-1.284***	1.276***
	Std_{rating}	-0.135	.269	-0.132	-0.142	.342	-0.004	-0.289	.290	-0.085

$^{***}p \leq .001$, $^{**}p \leq .01$, $^{*}p \leq .05$

4.2 Results and Discussion

We begin our experimental analysis by answering the above-outlined evaluation questions, in two separate studies: *Within-Dataset Analysis*, and *Between-Dataset Analysis*, presented subsequently.

4.2.1 Within-Dataset Analysis. The first part of the study contemplates answering Q1: "*Is there an underlying relationship between the described set of IVs computed from the URM and the DV?*"

Given a dataset, a recommendation model and an attack strategy, we build an explanatory-regression model to explain the relationship between the six IVs and the DV. Regression results for the within-dataset analysis across different dimensions are summarized in Table 4. The results obtained for the adjusted coefficient of determination ($adj.R^2$) in Table 4 reveal that the six dataset characteristics can explain more than 60% of the variation in $\Delta_{HR@k}$ irrespective of the attack type, CF model and domain (dataset). For instance, by focusing at one randomly selected attack (e.g., the Popular attack), against User-*k*NN, Item-*k*NN, and SVD models on samples extracted from ML-20M, one can note that the six IVs can respectively explain 85.9%, 91.2% and 77.2% of the variations in $\Delta_{HR@k}$. The corresponding $adj.R^2$ values for three models on Yelp are 78.4%, 75.9%, and 86.4%, and for LFM-1b 66.5%, 65.5% and 78.1%. The $adj.R^2$ coefficient reaches maximum values for the SVD model on samples extracted from Yelp ($adj.R^2 > 85\%$), while minimum on User-*k*NN for LFM-1b ($66\% < adj.R^2 < 67\%$). These results provide (strong) empirical evidence to support the hypothesis that the six identified IVs can explain a substantial portion of the variations in the attack impact measured by $\Delta_{HR@k}$ independently from <attack, dataset, model> combination. The explanatory power is highest for the model-based SVD approach (when comparing the global behavior of each CF model), however not a similar observation could be made in favor of a singular attack strategy.

The second part of the study concerns answering Q2: "*How do these data characteristics impact variations of $\Delta_{HR@k}$ in terms of the significance and directionality?*"

In addition to investigating the underlying relationship between data characteristics and attacks' effectiveness against RS, our study has a pragmatic goal, which entails providing insights about the directionality and significance of the impact of such properties. We discuss this aspect for each, or group, of IVs in the following:

Constant term: As mentioned in Section 4, all the IVs are mean-centered before building the regression model. Therefore, the constant term represents the *expected attack impact* measured in terms of $\Delta_{HR@k}$ for a given <attack, CF-model, dataset> triplet. For example, considering the random attacks on User-kNN, for a random sample (sub-dataset) with average dataset characteristics extracted from ML-20M, Yelp, and LFM-1b, the expected $\Delta_{HR@k}$ are 0.179, 0.609 and 0.717, respectively. The knowledge about expected performance can be useful in giving the system designer with a predictive knowledge about attacks' impacts under average conditions. However, it remains outside the main focus of this work, as we aim for explanatory performance (rather predictive) of the system; we nevertheless report these results for the sake of completeness.

Impact of data characteristics: The first observation is that unlike the findings in [2], which show a consistent significant behavior for all the aforementioned IVs in explanation of the general performance of RS (not for shilling attacks), the significance of the computed regression coefficients for the IVs tends to vary for each IV or group of IVs. The results show that the regression coefficients computed for the **structural URM characteristics** (i.e., $SpaceSize_{log}$, $Shape_{log}$, $Density_{log}$) are statistically significant. This suggests that there is enough statistical evidence to support the hypothesis that structural URM characteristics can explain the variations in the DV ($p < 0.05, 0.01, 0.001$), which is equal to state that there is an underlying relations between these three IVs and the DV. However, results for the other IVs vary depending on <attack, CF-model, dataset> triplet, or can be insignificant as in the case of Std_{rating}. For instance, the coefficients for Gini indices (i.e., $Gini_{user}$ and $Gini_{item}$) are mostly significant for shilling attacks against SVD particularly for samples drawn from the Yelp and LFM-1b datasets. The coefficients for Std_{rating} are insignificant ($p\text{-}value > 0.05$) in all experimented cases, except for two cases <Random/Average attack, SVD, Yelp>, implying that this dataset characteristic, which deals directly with rating values of the URM, plays a insignificant role on the impact of shilling attacks against CF models.

In summary, the results of within-dataset analysis provide strong statistical evidence that ***structural*** *URM characteristics (i.e., $Shape_{log}$, $SpaceSize_{log}$, $Density_{log}$) play a pivotal role on the impact of attacks targeted on CF models for all cases in the <attack, CF, dataset> triplet; rating frequency features play a significant role mostly for attacks targeted on model-based SVD recommendation. Finally, the role of Std_{rating} feature (dealing directly with rating values) cannot be confirmed as they showed no evidence of having a significant impact.*

Given the statistical significance of the regression coefficients for a number of IVs, the next step is to explore the *directionality* of this impact. Results summarized in Table 4 show that the effect of $Density_{log}$ is **negative** on $\Delta_{HR@k}$ across majority of the cases in <attacks, CF-model, dataset> triplet (except the ones on <SVD, Yelp>. This result is interesting and is consistent with findings in RS literature that increasing the density (or decreasing sparsity) of the URM not only improves the general performance of CF models (as recognized in the prior research [2, 12]), but also **reduces** the likelihood of attacks' effectiveness. One plausible explanation can be as follows: if we fix[3] number of users and items and increase the number of genuine ratings for instance by asking users to rate more, the accuracy of similarities computed is improved due to using more genuine ratings. As these similarities are generally vulnerable to the insertion of fake profiles, adding more genuine feedbacks can help to decrease the impact of attacks.

Additionally, we can note that $SpaceSize_{log}$ has a **negative** impact on $\Delta_{HR@k}$ in **neighborhood models**, which means that increasing the space size of the URM makes neighborhood models less vulnerable against attacks. Higher $SpaceSize_{log}$ (under fixed sparsity) means larger number of users/items and ratings. This provides neighborhood models with more non-malicious candidate users (and items) to compute similarities and can reduce the effect of attacks. Finally and on the contrary, $Shape_{log}$ presents a consistently

[3]Note that in providing these examples, we fix other IVs and focus on the impact of one singular IV.

Table 5: Regression results for the between dataset analysis (attack size 1%).

$\Delta_{HR@10}$		User-kNN	Item-kNN	SVD
Random	$R^2(adj.R^2)$	0.832(0.831)	0.814(0.813)	0.843(0.843)
	ML-20M (Constant)	.179***	.262***	.482***
	Yelp	.429***	.347***	.041***
	LFM-1b	.537***	.452***	.204***
	$SpaceSize_{log}$	-0.197***	-0.096***	.047***
	$Shape_{log}$	.153***	.108***	.204***
	$Density_{log}$	-0.729***	-0.550***	-0.253***
	$Gini_{user}$	.552***	-0.008	.101
	$Gini_{item}$	.728***	.439***	-0.032
	Std_{rating}	-0.057	.058	-0.029
Love-Hate	$R^2(adj.R^2)$	0.817(0.816)	0.774(0.773)	0.833(0.832)
	ML-20M (Constant)	.267***	.419***	.655***
	Yelp	.390***	.243***	-0.077***
	LFM-1b	.449***	.295***	.031***
	$SpaceSize_{log}$	-0.142***	.040***	.113***
	$Shape_{log}$	.174***	.090***	.083***
	$Density_{log}$	-0.620***	-0.289***	-0.137***
	$Gini_{user}$	.679***	-0.218	-0.285**
	$Gini_{item}$	.429***	.021	.122
	Std_{rating}	-0.073	.055	-0.032
Bandwagon	$R^2(adj.R^2)$	0.831(0.831)	0.818(0.817)	0.848(0.848)
	ML-20M (Constant)	.179***	.244***	.435***
	Yelp	.427***	.364***	.087***
	LFM-1b	.537***	.470***	.253***
	$SpaceSize_{log}$	-0.199***	-0.118***	.008
	$Shape_{log}$	.161***	.115***	.235***
	$Density_{log}$	-0.730***	-0.591***	-0.331***
	$Gini_{user}$	.589***	.082	.267*
	$Gini_{item}$	.720***	.497***	-0.019
	Std_{rating}	-0.059	.058	-0.018
Popular	$R^2(adj.R^2)$	0.744(0.742)	0.741(0.740)	0.800(0.799)
	ML-20M (Constant)	.589***	.537***	.725***
	Yelp	.222***	.252***	.051***
	LFM-1b	.133***	.166***	-0.032***
	$SpaceSize_{log}$	-0.059***	.051***	.040**
	$Shape_{log}$	.191***	.169***	.111***
	$Density_{log}$	-0.445***	-0.252***	-0.283***
	$Gini_{user}$	.544***	-0.050	-0.140
	$Gini_{item}$	.229*	-0.258*	.288**
	Std_{rating}	-0.124	-0.011	-0.017
Average	$R^2(adj.R^2)$	0.828(0.827)	0.810(0.809)	0.844(0.843)
	ML-20M (Constant)	.187***	.275***	.502***
	Yelp	.421***	.332***	.020***
	LFM-1b	.529***	.438***	.186***
	$SpaceSize_{log}$	-0.193***	-0.082***	.065***
	$Shape_{log}$	.152***	.107***	.192***
	$Density_{log}$	-0.718***	-0.522***	-0.219***
	$Gini_{user}$	.559***	-0.039	.011
	$Gini_{item}$	.717***	.407***	-0.062
	Std_{rating}	-0.054	.059	-0.013
Perfect Knowledge	$R^2(adj.R^2)$	0.812(0.811)	0.813(0.812)	0.847(0.846)
	ML-20M (Constant)	.266***	.274***	.479***
	Yelp	.341***	.328***	.039***
	LFM-1b	.449***	.434***	.207***
	$SpaceSize_{log}$	-0.141***	-0.088***	.049***
	$Shape_{log}$	.167***	.109***	.206***
	$Density_{log}$	-0.613***	-0.540***	-0.250***
	$Gini_{user}$	.479***	-0.035	.087
	$Gini_{item}$	.546***	.387***	-0.048
	Std_{rating}	-0.061	.048	-0.031

$^{***}p \leq .001$, $^{**}p \leq .01$, $^{*}p \leq .05$

positive influence on the efficacy of the attacks. This is a novel insight. We can explain it by considering the following example: increasing $Shape_{log}$ leads to an increased number of users with respect to items (i.e., decreasing items) in this way it could be easier to push the target item to higher positions inside the recommendation list (i.e., fewer items contribute to the recommendation).

4.2.2 Between-Dataset Analysis: The goal of the within-dataset analysis presented in the previous section was to investigate the impact of data characteristics on shilling attacks for each <attack, CF-model, dataset> triplet. In this section, we aim to provide a *domain-independent* analysis of the same study (impact of data characteristics on attacks' effectiveness) by combing rating scores of all three datasets. The regression model and the OLS follow Eq. 11 and 13 and we replicate the exact procedure described in [2]. Note that the DV here contains rating samples from all three datasets. Results of the between-dataset analysis are summarized in Table 5. The adjusted *coefficients of determination* $adj.R^2$ in Table 5 are consistent with those in within-dataset analysis in most experimental cases. For instance, $adj.R^2$ tells us that the selected IVs explain more than 80% of the variation in $\Delta_{HR@k}$ independently form <attack, CF-model> pair. Furthermore, results still support that structural URM properties have a statistically significant impact on each CF model. In fact, the *p-values* of $SpaceSize_{log}$, $Shape_{log}$, and $Density_{log}$ regression coefficients are less than 0.001 in each pair of experiments. Moreover, the *directionality* analysis of structural IVs in Table 5 is consistent with the insights drawn from previous study. In summary, for all CF models, $Shape_{log}$ has a positive impact, $Density_{log}$ has a negative influence, while the impact of $SpaceSize_{log}$ is (for most cases) negative on neighborhood recommenders and positive on model-based models. Additionally, Table 5 shows rating frequency IVs have not shared statistically significant impact on the DV.

In summary, the results of the between-dataset analysis support those presented in the within-dataset analysis. Given the heterogeneity of domains and variety of attack and CF models tested, this can be interpreted by the fact that ***effects*** *of data characteristics studied in this work are* ***NOT domain-specific*** *and the insights/conclusions obtained from this study can be applied to a broad range of domains for most popular attack and CF models.*

5 RELATED WORK

Collaborative Recommendation: Recommendation techniques are generally classified into collaborative filtering (CF), content-based filtering (CBF), and hybrid [1, 48]. CBF models recommend items similar to those users preferred in the past based on the item's characteristics (e.g., item content). CF leverages users' collective behavior data such as interactions and stated preferences to compute recommendations. Hybrid models combine CF and CBF techniques under a unique framework. In this study, we focus on two broadly adopted classes of CF models, namely: neighborhood-based [17, 23] and model-based [47, 52] approaches.

While neighborhood models rely on computing *similarities* from user behavioral data (i.e., user-user or item-item similarities), to predict unknown user preferences, model-based recommenders transform items and users into a shared latent factor space whose interactions explain the observed interactions. Depending on the type of interaction, model-based CF can be for example classified according to linear MF approaches [14, 47] and non-linear models such as the ones based deep neural networks [54]. These different functioning mechanisms of the neighborhood and model-based models, make them suitable choices for the study of shilling attacks, as they were considered in this study.

Security of Collaborative Recommender: Attacks on RS can be conducted based on two main classes: (i) hand-crafted, and (ii) machine-learned [16]. The former relies on hand-engineered fake user profiles (typically a rating profile) injected to the system, while the latter are machine-learning attacks optimized to find minimal perturbation of the URM to alter recommendation performance [34]. The focus of the present work is on shilling attacks, however, the results/insights obtained by this study can be extended and applied to other e.g., machine-learned attacks.

Shilling attacks against recommendation models can be categorized based on various dimensions: the intent behind the attack (push or nuke) [32, 36], and the attacker's knowledge, i.e., informed [19, 37] and semantic-enhanced [7] attacks. A large number of research articles has been produced in the context of shilling attacks, which can be broadly categorized into three research directions: (i) attack types [19, 32, 35], (ii) detection strategies [4, 11, 38] and (iii) robustness evaluation [42]. A common characteristic of the prior literature is that they are mostly focused on the algorithmic and procedural exploration and analysis of attack and defense strategies. The user-rating matrix (URM) (and properties extracted from it) is the pivotal information source of CF and attack models. A substantial amount of works explored the effects of different data characteristics measured from URM on recommendation accuracy. For instance, the *sparsity* of the dataset has been widely studied since it largely influences recommendation accuracy [2, 12], and so the *skeweness* of data (i.e., the distribution of feedback across items) has been demonstrated to influence the problem of predicting customer behavior and suggesting matching products [8, 25]. However, we believe that there exists a lack of systematic and large-scale analysis of the impact of dataset characteristics (e.g., sparsity, size, rating skewness) on the robustness of collaborative models against shilling attacks. The goal of this work is to fill this gap by investigating the effects of URM data characteristics on an attack performance metric with explanatory-based regression models.

6 CONCLUSION

The goal of this work is to study the impact of data characteristics on the effectiveness of most famous shilling attacks against popular CF methods. We considered a suite of data characteristics, which can be classified according to: (i) the structure of the URM, (ii) the rating frequency distribution, and (iii) the rating values. We used a regression-based explanatory model and relied on statistical significance with informed *p-value* in order to verify the impact of data characteristics.

Results of extensive experiments provide sufficient statistical evidence to accept the hypothesis that, first, the identified data characteristics can account for a considerable portion of variations in attack performance (global perspective) and, second, that there remain considerable differences in the significance (and directionality) of this impact among features. For instance, while URM structural

properties (size, shape, density) consistently indicate having an impact on the model output, the rating property (standard deviation of ratings) does not show an effect. Distribution properties (Gini user and item) show a higher impact on memory-based models. As the proposed explanatory framework can support a system designer in evaluating the robustness performance looking at the dataset characteristics, we plan to extend the set of studied characteristics (e.g., user-item relations), CF models (e.g., deep learning approaches) and adversarial machine-learned attacks [16].

REFERENCES

[1] 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 Trans. Knowl. Data Eng.* 17, 6 (2005), 734–749.

[2] Gediminas Adomavicius and Jingjing Zhang. 2012. Impact of data characteristics on recommender systems performance. *ACM Trans. Management Inf. Syst.* 3, 1 (2012), 3:1–3:17.

[3] Charu C. Aggarwal. 2016. *Recommender Systems - The Textbook.* Springer.

[4] Mehmet Aktukmak, Yasin Yilmaz, and Ismail Uysal. 2019. Quick and accurate attack detection in recommender systems through user attributes. In *RecSys*. ACM, 348–352.

[5] Santiago Alonso, Jesús Bobadilla, Fernando Ortega, and Ricardo Moya. 2019. Robust Model-Based Reliability Approach to Tackle Shilling Attacks in Collaborative Filtering Recommender Systems. *IEEE Access* 7 (2019), 41782–41798.

[6] Chris Anderson. 2006. *The long tail: Why the future of business is selling less of more.* Hachette Books.

[7] Vito Walter Anelli, Yashar Deldjoo, Tommaso Di Noia, Eugenio Di Sciascio, and Felice Antonio Merra. 2020. SAShA: Semantic-Aware Shilling Attacks on Recommender Systems exploiting Knowledge Graphs. In *17th European Semantic Web Conference ESWC 2020*. Springer.

[8] Chidanand Apté, Bing Liu, Edwin P. D. Pednault, and Padhraic Smyth. 2002. Business applications of data mining. *Commun. ACM* 45, 8 (2002), 49–53.

[9] Robin D. Burke. 2007. Hybrid Web Recommender Systems. In *The Adaptive Web, Methods and Strategies of Web Personalization*. 377–408.

[10] Robin D. Burke, Bamshad Mobasher, Runa Bhaumik, and Chad Williams. 2005. Segment-Based Injection Attacks against Collaborative Filtering Recommender Systems. In *ICDM*. IEEE Computer Society, 577–580.

[11] Paul-Alexandru Chirita, Wolfgang Nejdl, and Cristian Zamfir. 2005. Preventing shilling attacks in online recommender systems. In *WIDM*. ACM, 67–74.

[12] Paolo Cremonesi, Yehuda Koren, and Roberto Turrin. 2010. Performance of recommender algorithms on top-n recommendation tasks. In *Proc. of the 2010 ACM Conference on Recommender Systems, RecSys 2010*. 39–46.

[13] Paolo Cremonesi, Antonio Tripodi, and Roberto Turrin. 2011. Cross-Domain Recommender Systems. In *ICDM Workshops*. IEEE Computer Society, 496–503.

[14] Scott C. Deerwester, Susan T. Dumais, Thomas K. Landauer, George W. Furnas, and Richard A. Harshman. 1990. Indexing by Latent Semantic Analysis. *JASIS* 41, 6 (1990), 391–407.

[15] Yashar Deldjoo, Tommaso Di Noia, and Felice Antonio Merra. 2019. Assessing the Impact of a User-Item Collaborative Attack on Class of Users. In *ImpactRS@RecSys*, Vol. 2462. CEUR-WS.org.

[16] Yashar Deldjoo, Tommaso Di Noia, and Felice Antonio Merra. 2020. Adversarial Machine Learning in Recommender Systems: State of the art and Challenges. *CoRR* abs/2005.10322 (2020). arXiv:2005.10322

[17] David Goldberg, David A. Nichols, Brian M. Oki, and Douglas B. Terry. 1992. Using Collaborative Filtering to Weave an Information Tapestry. *Commun. ACM* 35, 12 (1992), 61–70.

[18] Carlos A. Gomez-Uribe and Neil Hunt. 2016. The Netflix Recommender System: Algorithms, Business Value, and Innovation. *ACM Trans. Management Inf. Syst.* 6, 4 (2016), 13:1–13:19.

[19] Ihsan Gunes, Cihan Kaleli, Alper Bilge, and Huseyin Polat. 2014. Shilling attacks against recommender systems: a comprehensive survey. *Artif. Intell. Rev.* 42, 4 (2014), 767–799.

[20] F. Maxwell Harper and Joseph A. Konstan. 2016. The MovieLens Datasets: History and Context. *TiiS* 5, 4 (2016), 19:1–19:19.

[21] Xiangnan He, Zhankui He, Xiaoyu Du, and Tat-Seng Chua. 2018. Adversarial Personalized Ranking for Recommendation. In *SIGIR*. ACM, 355–364.

[22] Benjamin Heitmann and Conor Hayes. 2010. Using Linked Data to Build Open, Collaborative Recommender Systems. In *2010 AAAI Spring Symposium Series*.

[23] Jonathan L. Herlocker, Joseph A. Konstan, Al Borchers, and John Riedl. 2017. An Algorithmic Framework for Performing Collaborative Filtering. *SIGIR Forum* 51, 2 (2017), 227–234.

[24] Jonathan L. Herlocker, Joseph A. Konstan, and John Riedl. 2000. Explaining collaborative filtering recommendations. In *CSCW 2000, Philadelphia, PA, USA, December 2-6, 2000*. 241–250.

[25] Chun-Nan Hsu, Hao-Hsiang Chung, and Han-Shen Huang. 2004. Mining Skewed and Sparse Transaction Data for Personalized Shopping Recommendation. *Machine Learning* 57, 1-2 (2004), 35–59.

[26] Zan Huang, Hsinchun Chen, and Daniel Dajun Zeng. 2004. Applying associative retrieval techniques to alleviate the sparsity problem in collaborative filtering. *ACM Trans. Inf. Syst.* 22, 1 (2004), 116–142.

[27] Nicolas Hug. 2017. Surprise, a Python library for recommender systems.

[28] Dietmar Jannach, Markus Zanker, Alexander Felfernig, and Gerhard Friedrich. 2010. *Recommender Systems - An Introduction*. Cambridge University Press.

[29] Yehuda Koren. 2010. Factor in the neighbors: Scalable and accurate collaborative filtering. *TKDD* 4, 1 (2010), 1:1–1:24.

[30] Yehuda Koren and Robert M. Bell. 2015. Advances in Collaborative Filtering. In *Recommender Systems Handbook*. 77–118.

[31] Yehuda Koren, Robert M. Bell, and Chris Volinsky. 2009. Matrix Factorization Techniques for Recommender Systems. *IEEE Computer* 42, 8 (2009), 30–37.

[32] Shyong K. Lam and John Riedl. 2004. Shilling recommender systems for fun and profit. In *WWW*. ACM, 393–402.

[33] Kibeom Lee and Kyogu Lee. 2015. Escaping your comfort zone: A graph-based recommender system for finding novel recommendations among relevant items. *Expert Syst. Appl.* 42, 10 (2015), 4851–4858.

[34] Bo Li, Yining Wang, Aarti Singh, and Yevgeniy Vorobeychik. 2016. Data Poisoning Attacks on Factorization-Based Collaborative Filtering. In *NIPS*. 1885–1893.

[35] Bamshad Mobasher, Robin Burke, Runa Bhaumik, and Chad Williams. 2005. Effective attack models for shilling item-based collaborative filtering systems. In *Proceedings of the WebKDD Workshop*. Citeseer, 13–23.

[36] Bamshad Mobasher, Robin Burke, Runa Bhaumik, and Chad Williams. 2007. Toward trustworthy recommender systems: An analysis of attack models and algorithm robustness. *ACM Transactions on Internet Technology (TOIT)* 7, 4 (2007).

[37] Bamshad Mobasher, Robin D. Burke, Runa Bhaumik, and Jeff J. Sandvig. 2007. Attacks and Remedies in Collaborative Recommendation. *IEEE Intelligent Systems* 22, 3 (2007), 56–63.

[38] Bamshad Mobasher, Robin D. Burke, Runa Bhaumik, and Chad Williams. 2007. Toward trustworthy recommender systems: An analysis of attack models and algorithm robustness. *ACM Trans. Internet Techn.* 7, 4 (2007).

[39] Yashar Moshfeghi, Benjamin Piwowarski, and Joemon M Jose. 2011. Handling data sparsity in collaborative filtering using emotion and semantic based features. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*. ACM, 625–634.

[40] Xia Ning, Christian Desrosiers, and George Karypis. 2015. A Comprehensive Survey of Neighborhood-Based Recommendation Methods. In *Recommender Systems Handbook*. 37–76.

[41] Michael P O'Mahony. 2004. *Towards robust and efficient automated collaborative filtering*. Ph.D. Dissertation. Citeseer.

[42] Michael P. O'Mahony, Neil J. Hurley, Nicholas Kushmerick, and Guenole C. M. Silvestre. 2004. Collaborative recommendation: A robustness analysis. *ACM Trans. Internet Techn.* 4, 4 (2004), 344–377.

[43] Michael P. O'Mahony, Neil J. Hurley, and Guenole C. M. Silvestre. 2005. Recommender Systems: Attack Types and Strategies. In *AAAI*. 334–339.

[44] Michael P O'Mahony, Neil J Hurley, and Guenole CM Silvestre. 2002. Promoting recommendations: An attack on collaborative filtering. In *International Conference on Database and Expert Systems Applications*. Springer, 494–503.

[45] Michael P O'Mahony, Neil J Hurley, and Guenole CM Silvestre. 2003. An evaluation of the performance of collaborative filtering. In *14th Irish Artificial Intelligence and Cognitive Science (AICS 2003) Conference*. Citeseer.

[46] C. Radhakrishna Rao. 1973. *Linear Statistical Inference and its Applications, Second Editon*. Wiley.

[47] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. 452–461.

[48] Francesco Ricci, Lior Rokach, and Bracha Shapira. 2011. Introduction to Recommender Systems Handbook. In *Recommender Systems Handbook*. 1–35.

[49] Markus Schedl. 2016. The LFM-1b Dataset for Music Retrieval and Recommendation. In *ICMR*. ACM, 103–110.

[50] Galit Shmueli et al. 2010. To explain or to predict? *Statistical science* (2010).

[51] Brent Smith and Greg Linden. 2017. Two Decades of Recommender Systems at Amazon.com. *IEEE Internet Computing* 21, 3 (2017), 12–18.

[52] Hao Wang, Naiyan Wang, and Dit-Yan Yeung. 2015. Collaborative Deep Learning for Recommender Systems. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Sydney, NSW, Australia, August 10-13, 2015*. 1235–1244.

[53] Fuzhi Zhang, Yuanli Lu, Jianmin Chen, Shaoshuai Liu, and Zhoujun Ling. 2017. Robust collaborative filtering based on non-negative matrix factorization and R_1-norm. *Knowl.-Based Syst.* 118 (2017), 177–190.

[54] Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tay. 2019. Deep Learning Based Recommender System: A Survey and New Perspectives. *ACM Comput. Surv.* 52, 1 (2019), 5:1–5:38.

[55] Wei Zhou, Junhao Wen, Qingyu Xiong, Min Gao, and Jun Zeng. 2016. SVM-TIA a shilling attack detection method based on SVM and target item analysis in recommender systems. *Neurocomputing* 210 (2016), 197–205.