Evaluation of recommendation systems continues evolving, especially in recent years. There have been several attempts to standardize the assessment processes and propose replacement metrics better oriented toward measuring effective personalization. However, standard evaluation tools merely possess the capacity to provide a general overview of a system’s performance; they lack consistency and effectiveness in their use, as evidenced by most recent studies on the topic. Furthermore, traditional evaluation techniques fail to detect potentially harmful data on small subsets. Moreover, they generally lack explainable features to interpret how such minor variations could affect the system’s performance. This proposal focuses on data clustering for recommender evaluation and applies a cluster assessment technique to locate such performance issues. Our new approach, named group validation, aids in spotting critical performance variability in compact subsets of the system’s data and unravels hidden weaknesses in predictions where such unfavorable variations generally go unnoticed with typical assessment methods. Group validation for recommenders is a modular evaluation layer that complements regular evaluation and includes a new unique perspective to the evaluation process. Additionally, it allows several applications to the recommender ecosystem, such as model evolution tests, fraud/attack detection, and the capacity for hosting a hybrid model setup.

CCS Concepts: • Information systems → Recommender systems; Presentation of retrieval results;

Additional Key Words and Phrases: Recommender systems, offline evaluation, model validation, data clustering

ACM Reference Format:
Wissam Al Jurdi, Jacques Bou Abdo, Jacques Demerjian, and Abdallah Makhoul. 2024. Group Validation in Recommender Systems: Framework for Multi-layer Performance Evaluation. ACM Trans. Recomm. Syst. 2, 1, Article 8 (March 2024), 25 pages. https://doi.org/10.1145/3640820

The Lebanese University Research Program has supported this work.

Authors’ addresses: W. Al Jurdi, University Bourgogne Franche-Comté, FEMTO-ST Institute, CNRS, Montbéliard, France; LaRRIS, Faculty of Sciences, Lebanese University, Fanar, Lebanon; e-mail: wissam.aljurdi@st.ul.edu.lb, wissam.al_jurdi@univ-fcomte.fr; J. Bou Abdo, University of Cincinnati - University of Nebraska at Kearney, Kearney, Nebraska; e-mail: bouabdjs@ucmail.uc.edu; J. Demerjian, LaRRIS, Faculty of Sciences, Lebanese University, Fanar, Lebanon; e-mail: jacques.demerjian@ul.edu.lb; A. Makhoul, University Bourgogne Franche-Comté, FEMTO-ST Institute, CNRS, Montbéliard, France; e-mail: abdallah.makhoul@univ-fcomte.fr.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

© 2024 Copyright held by the owner/author(s). Publication rights licensed to ACM.

ACM Trans. Recomm. Syst., Vol. 2, No. 1, Article 8. Publication date: March 2024.
1 INTRODUCTION

Recommendation systems analyze information in datasets to predict what may interest users. This process utilizes available data features such as search and interaction details. Although it is relatively simple to achieve with current research and suitable models or workflows, evaluating effectiveness and testing performance as machine learning models evolve with data is still a significant challenge [1–3] that has captured considerable attention [4–6]. Microsoft’s best practices for building recommendation systems have been helpful in this regard [7]. As a result, there have been some modern proposed approaches to tackle the evaluation challenges in recommender systems; those include the Simpson’s paradox [2], benchmarking frameworks and toolkits [8–10], dataset-oriented design [11], metric selection criteria [4, 12], and metric adaptations (predominantly in search settings) [13–16]. Although these topics tend to be diverse, almost all revolve around one prevalent idea: The tools and techniques for assessing models still require significant improvements. In short, there have not been enough attempts to consolidate the knowledge of recommender systems nor to systematically define the implications of evaluating recommenders for different tasks and under various contexts.

The topic of evaluation challenges in recommender systems is not recent and dates back to 2004. At the time, Herlocker et al. [17] abstracted all the factors considered in evaluating a recommender model’s effectiveness and proved the presence of potential biases in most reported evaluation results. Back then, the study highlighted the pitfalls that researchers ought to avoid when attempting to implement recommendation solutions, especially during the phase of trained models’ effectiveness evaluation. Some of those issues included are undefined user goals for a particular recommender system, evaluation of models on incompatible datasets, and incorrect use of evaluation metrics that are mostly not optimized to measure the performance based on the system’s prediction goals. Fast-forward to today, the same factors are still causing significant issues, as evidenced by recent comprehensive studies on the topics of noise and evaluation, such as those by Parapar et al. [14], Al Jurdi et al. [18], and Ovaisi et al. [3]. In the research and experiments of Reference [18], systematic proof shows how the tools and metrics employed to test contemporary model techniques and rank them in order of performance embody inconsistent and unreliable results. Some examples of the outcome include contradicting performance metric results, a sporadic evaluation metric selection process, and unsystematic model-data combinations in experimentation.

Further, the experiments demonstrate how randomly eliminating data from a dataset (in the training and evaluation phase) could result in shockingly comparable performance results with algorithms that specialize in identifying critical noisy data that severely impact performance. Even if not directly correlated, this aspect is perfectly aligned with the recent research by Sun et al. [8], which shows how a simple baseline model can outperform its superior and more complicated counterpart with the proper setup. Highly similar issues are also presented by Parapar et al. [14] in their proposal of a new unified metric that combines diversity and accuracy and can be used as a replacement for the outdated and less effective traditional measures.

Building on the arguments presented in References [3, 14, 18], and in an attempt to address the issues mentioned above debated in Reference [18], our work focuses on the objective of evaluating a recommender performance on sub-groups as opposed to solely relying on the traditional applications of evaluation metrics. While the current evaluation mechanisms can provide a general indicator about how the performance is expected to be in a live setting, an “averaged” evaluation, even with reasonably optimized metric results, can fail to reflect actual model behaviors on different data groups of a dataset regardless of how robust the metric is. This concept was first presented in general domain-independent (non-personalization) machine-learning studies such as the ones conducted by McMahan et al. and Chung et al. [19–22] in a proposal of an automatic data slicing mechanism for evaluating subset levels of the data. The results show that it is crucial to track
the performance on a more granular level to understand better the effectiveness of a particular model given a predefined set of prediction objectives. Ultimately, this aids us in better assessing the effectiveness of an algorithm on all parts of the data available for training and testing.

The concept of data slicing to uncover hidden performance issues has proven very effective [21]. Such a mechanism is not yet utilized in the recommender system domain, albeit briefly touched on in References [2, 3, 18]. The closest to the data slicing concept in recommenders is the research of Ovaisi et al. [3], since their evaluation toolkit supports sub-population evaluation (e.g., gender-based). However, it is not holistic and lacks the definition of an evaluation strategy. This sub-population evaluation scenario had been inspired by the idea of fairness in ranking evaluation discussed in the research of Singh et al. [23]. In our work, we focus on adapting a tailored and modular slicing evaluation framework, called group validation, to evaluate the performance of recommender models on a granular level. Our proposed mechanism will adapt the data slicing technique [21] and work through clustering datasets into groups and then identify which of those groups a given model performs the worst using a systematic evaluation strategy. Additionally, the group validation framework allows the possibility to track a recommender’s model performance across small data clusters and permits having practical applications such as:

- Enhancing automated decisions for model evolution.
- Detecting noise/fraud, such as malicious and natural noise.
- Creating a dynamic and hybrid model structure in a live environment.

The prime contribution of this chapter is implementing a modular evaluation framework for a distinct model validation process in recommender systems. Our applied method in this framework is based on data clustering\(^1\) followed by experimental tests to identify weaknesses in model performance on certain data groups. As will be proven throughout this study, such weaknesses can usually be hidden from metric results that report overall system performances. This research extends the previous works on evaluation and noise management [2, 18, 21] and adapts the slicing theory in the assessment from References [21, 22] to the recommender systems domain. Additionally, based on the theoretical analysis and the experiments conducted to position the framework against the current evaluation process, the following main points can be concluded:

- Clusters can be more sensitive to certain types of feedback, allowing the framework to detect adverse effects, something not possible solely with generic evaluation.
- Certain user feedback can evolve in a way that negatively affects other users. Cluster evaluation like that in the group validation framework can help localize this effect.

In the upcoming section, we present state-of-the-art works and position our work within this context, shedding light on the latest evaluation approaches in the field. Section 3 introduces our group validation framework with a comprehensive description of the proposed method, which comprises a clustering technique and a cluster assessment process. Afterward, in Section 4, we present an applied investigation of the group validation mechanism and include our experimental results and analysis. In Section 5, we offer some possible applications of the group validation approach. We conclude the work in Section 6.

2 BACKGROUND AND RELATED WORK

In this section, we cover all the studies we encountered that correlate with the proposed concept of group validation in recommender systems. The analysis is mainly grouped into three main pillars: the general slice-based evaluation in machine learning, a more related notion termed the Simpson’s

\(^{1}\)The “groups” in the group validation process are mainly clusters. Both words might appear interchangeably throughout the text.
Paradox, and finally, a short discussion around the evaluation benchmarking in recommenders to validate whether they discuss evaluation on different parts of data or not.

2.1 Slice-based Evaluation in Machine Learning

A series of studies focusing on data slicing while covering clustering methods for model evaluation was introduced by Chung et al. [20–22]. This series of work is the closest to the group-based method proposed in this research, as both leverage the concept of evaluating performance on smaller data pieces instead of a complete dataset evaluation. However, there are several core differences, as we will describe in what follows. The authors’ final method, presented in Reference [21] and called SliceFinder, is not developed in a way that would work for the recommender systems domain but rather for general classification problems. The reason for that is the use of the mechanism of features in the dataset to create subsets that make sense to the user in the end; such features are not present in recommender and personalization datasets where we usually have limited information about the users in an interactive system. SliceFinder is an interactive framework for identifying problematic slices using statistical techniques, and the slice evaluation mechanism is based on a general classification loss function.

This classification loss function returns a performance score for a set of examples by comparing $h$’s prediction $h(x_i^f)$ with the actual label $y_i$. The difference between the tests conducted to identify the least performing slices in this case and the method used for determining the critical groups with our mechanism is the core loss function and the evaluation metrics employed based on the recommender optimization and goal. In the group-based validation case, the loss function is connected to the evaluation metric that will eventually be used to measure the system’s performance. Moreover, we define several loss functions based on the recommender system’s optimization. As for the slicing mechanism, the authors in Reference [22] applied several methods to implement an automated data slicing technique, such as decision trees (non-overlapping) and lattice searching (overlapping), which also generated meaningful data chunks (also referred to as literals). Our group validation framework is quite different in this regard, as the datasets for recommender systems are different and include unique features and usually much less information than the features utilized in the study of SliceFinder. We are currently using a clustering approach based on the minimal features available in the type of datasets used in this study’s experiments. If we applied SliceFinder, then we would end up with many data point groups that do not make sense to the user (example of an easy-to-understand slice reported by SliceFinder: country = DE $\cap$ gender = Male). Such features are usually lacking when dealing with recommender system datasets. However, in a future study, we will tackle this issue by generalizing the proposed architecture to cover implicit feedback data in most e-commerce applications.

In the proposal of a comprehensive and rigorous framework for reproducible recommender evaluation by Anelli et al. [9], the authors mentioned the idea of statistical tests on data groups in the third section of the study. Throughout the work, it was emphasized that there is a need to compute fine-grained (such as per-user or per-partition) results and retain them for each recommendation model. As a result, their framework, called Elliot, was designed for multi-recommender evaluation and handling fine-grained results. Elliot brings the opportunity to compute two statistical hypothesis tests, i.e., Wilcoxon and Paired t-test, activating a flag in the configuration file. The proposal did not mention the partitioning techniques used other than the idea that the partitions might be per user. Partitioning per user is possible in our proposed method, and the groups could be generated by centering the clusters around users, similar to the k-Nearest Neighbors (KNN) [24] fashion. For instance, every user will form a distinct group. KNN is a non-parametric supervised learning method used for classification and regression where the input consists of the k closest training examples in a dataset. In our proposal, the fundamental idea of the introduced framework is to
report groups that might make more sense to track the recommender’s performance. That way, the evaluation process allows one to check the worst-performing groups and analyze the reason behind the results.

2.2 Subset Scanning

Subset Scanning is a highly efficient and accurate event and pattern detection framework that can operate on both spatial and non-spatial datasets [25, 26]. The approach optimizes a score function (such as likelihood ratio statistic) over subsets of the data, making it a flexible and scalable solution that can adapt to various real-world constraints (e.g., spatial adjacency, irregular shapes). By evaluating the score function over subsets, Subset Scanning can identify events and patterns quickly, significantly reducing the search space and making it more computationally efficient than traditional methods. Moreover, Subset Scanning builds upon spatial and space-time scan statistics, providing accurate results even when the valid spatial region of interest does not align perfectly with predefined search regions [27]. Subset Scanning is a powerful and practical tool for researchers across various domains, offering a valuable approach for detecting events and patterns with speed and accuracy.

Subset Scanning and the slice-based evaluation introduced in the previous section are two close techniques used in data analysis, each with a unique purpose and application. While Subset Scanning is primarily used for event detection and pattern identification, slice-based evaluation is commonly employed in machine learning for model development and validation. Subset Scanning identifies subsets of data that exhibit significant deviations from the expected distribution, making it ideal for detecting clusters in epidemiology, identifying pollution hotspots in environmental monitoring, and finding influential subgroups in social network analysis. This method optimizes a score function over subsets of data and focuses on identifying regions of interest efficiently, making it scalable and adaptable to irregularly shaped clusters.

However, slice-based evaluation involves examining model behavior across specific dimensions or “slices” of the data, allowing us to assess model behavior across different groups or specific scenarios. This enhances model interpretability by revealing how it performs across various subgroups, such as understanding if a classifier works equally well for all demographic groups. Slicing data is also scalable for model evaluation, especially when assessing performance across multiple dimensions.

While both methods involve examining subsets of data, their goals and applications diverge. Subset Scanning is specialized for event detection, whereas slice-based evaluation is a broader machine learning model assessment technique. Our proposal remains close in theory to Subset Scanning techniques and slice-based evaluation but remains specific to the recommender system features and applications context. As shown in the previous section, Group-based includes many elements from slice-based validation but can, in fact, be framed as a problem of Subset Scanning, especially in the following domains:

- Tackling data that exhibits significant deviations from an expected distribution;
- Pattern detection in subsets.

2.3 The Simpson’s Paradox in Recommender Systems

While not directly related to the evaluation of slices, Simpson’s paradox in the offline evaluation of recommendation systems, which is covered by Jadidinejad et al. [2], introduces a phenomenon that describes a very interesting phenomenon on granular evaluation. The research in Reference [2] shows that the typical offline evaluation of recommender systems suffers from the phenomenon termed Simpson’s paradox. Simpson’s paradox is when a significant trend appears in several
different sub-populations of observational data but disappears or is even reversed when these
sub-populations are combined.

Although the definition of Simpson’s paradox might theoretically be linked to our proposal, the
authors in Reference [2] conducted different experiments. They proposed an approach that tackles
a marginally different issue while utilizing recommender systems datasets. The experiments are
based on “stratified sampling” and reveal that a tiny minority of items that are frequently exposed
by a deployed system (such as the system that helped generate the open-source datasets, like
MovieLens [28], that are used in most of the studies on recommender systems) plays a confounding
factor in the offline evaluation of recommendation systems. So, the study investigates the issue of
an initial recommender system, called a confounder, that influences the rating elicitation process
of the users. This concept is the main difference between the study in Simpson’s paradox and the
group validation mechanism we are proposing, where the latter only tackles the issue with the
performance of the recommender on smaller groups of users.

2.4 Evaluation Benchmarking

A missing idea in the benchmarking proposals is the differentiation between different recom-
mender algorithms and their general goal inside a particular application. In our proposal, the
group-based mechanism evaluates a recommender based on the optimization of the model, which
is the recommended way of approaching performance assessment [7]. As touched on in the intro-
duction of our work, with the increase in the number of recommender algorithms proposed and
different approaches to enhance them, a critical issue presents itself. A standardized approach to
evaluating algorithmic enhancements as recommender proposals evolve does not exist. Some of
the recent studies on evaluation focus on the benchmarking method to create a framework for
the most accurate evaluation and comparison. In one of the proposals by Sun et al. [8], the au-
thors aimed to conduct rigorous (i.e., reproducible and fair) evaluation for implicit-feedback-based
top-N recommendation algorithms. They reviewed several recent proposals and analyzed the dif-
f erent approaches for evaluating recommender systems. As expected, several inconsistencies were
in what was utilized for evaluation, leading to inconsistent results when judging if a new proposal
is better than its predecessor. Accordingly, the authors created benchmarks with standardized pro-
dcedures and provided the performance of seven well-tuned models across six metrics on six widely
used datasets.

In a similar and more recent study, Ovaisi et al. [3] proposed a toolkit for evaluation to assess
recommender models’ robustness. In this research, the authors mentioned the necessity for evalu-
ating the model on different data slices, such as gender subgroups. In one particular example of the
study, a sub-population of the test set consisted of users who were grouped by gender. The system
performed much worse for females than males across all the models used in the experiment. This is
proof of the importance of studying the performance of data groups versus evaluating the whole
data altogether, where the negative performance tends to average out. Unfortunately, most rec-
ommenders’ data lacks essential features like user information. That is why forming meaningful
slices, such as those proposed in References [21, 22], work in theory but can be extremely difficult
to apply to the standard datasets for interactive systems. Our method overcomes this for the first
most common datasets in such systems: the rating-based datasets.

In general, most of the benchmarking toolkit proposals do not include studies about the quality
of the data used for the recommender application nor provide detailed coverage of the concept
related to analyzing the performance of smaller groups. The group validation method proposed in
this study fills this gap and provides a new layer of evaluation that better tracks the performance
of models across different groups.
3 GROUP VALIDATION IN RECOMMENDER SYSTEMS

As previously indicated, data slicing for evaluation was formulated in the ML community [19] and further adapted by Chung et al. [21] to design an automated mechanism for slice identification on a general binary classification use-case. In our proposal, we re-formulate and re-structure this automatic slicing mechanism to develop an effective cluster-based evaluation process in a personalized recommender system setting. The goal is to create a tool capable of identifying performance issues on smaller clusters in the data, especially in cases where the standard evaluation methods cannot recognize such performance drops. In the following sections, we present the methodology behind our proposed framework and the adaptations we applied to the slicing evaluation study done in Reference [21].

3.1 Data Slicing and Evaluation in Machine Learning

Following the mechanism of slicing-based evaluation in Reference [21], consider a binary classification model \( h \), a general training dataset \( D \) with some available features \( F \), and a value for every feature \( v \) (which could be numerical or non-numerical). A slice \( S \) is a subset of the data records in \( D \) and could be expressed in the following manner: \( \cap_j (F_j \text{ op } v_j) \), where \( \text{op} \) could be any comparison operator and \( F_j \) a certain distinct value. One example of a random slice \( S \) could be: \( \text{(country} = \text{DE}) \cap \text{(gender} = \text{Female)} \). The rationale behind the data slicing evaluation method in References [20–22] is a trained Decision Tree (DT) coupled with a loss function that acts as the basis for which a slice’s evaluation performance is measured. As the use-case implemented in the test only included a binary classification problem, the loss function used was the general logarithmic loss equation defined as:

\[
\psi(h(x^i_F), y^i) = -\frac{1}{n} \sum_{(x^i_F, y^i) \in S} [y^i \ln h(x^i_F) + (1 - y^i) \ln (1 - h(x^i_F))] 
\]

(1)

where \( h(x^i_F) \) is the model’s prediction, \( y^i \) is the true label, and \( n \) is the total number of data points in \( S \). The authors automated the slice identification process by setting up the DT model that utilizes a breadth-first traversal. This model travels down through the data starting at the root slice, which comprises the whole dataset and evaluates every possible easy-to-understand slice, i.e., a slice with the maximum number of feature combinations after which it could not be easily understandable [21]. Ultimately, the model identifies a list of top-\( k \) slices where a test model performs the worst by evaluating every slice as a stand-alone dataset using Equation (1). A slice is considered critical and should be reported as part of the top-\( k \) when the following primary condition is satisfied:

\[
S_{\text{critical}} : \psi(S) - \psi(S') > 0 \quad \Rightarrow \quad C_{\text{critical}} = \text{True} 
\]

(2)

where \( S' \) corresponds to the rest of the examples in the dataset \( D \) and can be calculated as \( S' = D - S \). Afterward, the resulting slices of the first part of Equation (2) will have to satisfy a supplemental two-aspect condition, which we call in our work here \( C_{\text{critical}} \), to determine if \( S \) indeed has a significantly higher loss than \( S' \). The first condition is Welch’s t-test [29], which measures the existence of an effect by determining if the difference in loss is statistically significant. The second aspect is the effect size [30], which complements the statistical significance and measures the magnitude of the effect. The combined usage of those two tests is adopted from the study of the effect size [31]. The two-aspect condition equations are summarized as follows:

\[
C_{\text{critical}} = \begin{cases} 
\text{t-test} & H_0 : \psi(S) \leq \psi(S') \\
& H_A : \psi(S) > \psi(S') \\
\text{effect-test} & \phi = \sqrt{2} \times \frac{\psi(S) - \psi(S')}{\sqrt{\sigma_S^2 + \sigma_{S'}^2}} ; \phi \geq 0.8
\end{cases} 
\]

(3)
where $\sigma^2_S$ and $\sigma^2_{S'}$ are the variances of the individual example losses in $S$ and $S'$, respectively. Equation (3) would result in True for a given slice $S$ only if both tests succeed, i.e., the Welch's t-test leads to the alternative hypothesis $H_a$ passing, and the significance test equates to a large significance value on Cohen's scale [32]—which is typically a value greater than or equal to 0.8.

### 3.2 Group Validation in Recommenders

To adapt the concept of slicing-based evaluation to recommenders in our group validation framework, we examine several aspects of the automated data slicing mechanism. First, in the recommender domain, the datasets are of different shapes and forms compared to data from other ML problems that tend to be more general. Ordinarily, recommender datasets are naturally sparse and lack rich features found in datasets like the one used for the experiments of the data slicing mechanism of Reference [21]. Typically, a recommender system model, $M$, produces a list of ranked item suggestions $\mathbf{i} = (i_1, i_2, \ldots, i_n)$ selected from the whole set of items not previously seen by the user in his profile $I_u$. The items are usually associated with aspects $A = \{a_1, a_2, \ldots, a_c\}$, which could be any categorical classification of items such as genres for a particular movie in a recommender movie dataset like MovieLens [28] ($a_{\text{action}}, a_{\text{comedy}}, a_{\text{drama}} \ldots$). Second, the process of rating elicitation is highly subjective [18, 33] in the recommender domain. Typically, such feedback is translated into a 5-point Likert scale: $r_{u,i} \in \{1, 2, \ldots, 5\}$, where $r_{u,i}$ represents the feedback user $u$ gives on item $i$. Although implicit feedback is becoming very popular lately, many systems are still utilizing the rating-based data format, as evidenced by recent publications on significant issues such as noise and attacks [18], personalization enhancement [34], model evaluation and effectiveness [2, 14], and model variations [35]. A field experiment by Zhao et al. [36] shows that blending explicit and implicit user feedback through an online learning algorithm can benefit user engagement and mitigate the increased browsing effort cost. In other words, this means that the browsing experience is generally improved, and there is much less effort on the user’s side to reach the intended content.

#### 3.2.1 Data Slicing

Clustering has been implemented as a baseline mechanism in the binary classification use-case experiment in the study of automated data slicing for evaluation [21]. The authors argue that clustering would not be an optimal approach in the binary classification problem (which could also be a regression problem), because the clusters would be complex to interpret unless a manual investigation is conducted. Further, they add that the user has to specify the number of clusters beforehand, presenting a different problem affecting the ability to automate the process.

However, things are different in a personalized recommender context where we will utilize a clustering method as the primary step in the group validation framework. First, if we apply the same slicing method of Reference [21] to recommender datasets, then the slices would not make sense, because the features of both datasets are different, as indicated at the beginning of this section. Recommender datasets are typically sparse; other vital features such as user biodata and specific extended information about items could be nonexistent. It is very challenging to effortlessly form a slice like this: $(\text{country} = \text{DE}) \cap (\text{gender} = \text{Female})$ from a recommender dataset when we do not have enough data on such features. Additionally, clustering in recommender systems is very effective due to the personalized nature of the data and the way feedback is collected and used as the ground truth in training. It has been heavily leveraged to address several issues in models, such as balancing diversity, consistency, and reliability of ranked recommendations, leveling the data sparsity of user-preference matrices, and accounting for changes in user preferences over time [34, 37, 38].
As a result, in the context of group validation in recommenders, slices will be referred to as groups where $G$, the counterpart of $S$, now represents a subset of the recommender dataset $D$. The clustering technique we will use to form those dataset groups is the unsupervised learning algorithm k-means [39, 40]. The k-means algorithm commonly works by grouping data together in $n$ clusters with equal variance. It does this by minimizing a parameter called the inertia or within-cluster sum-of-squares. It is essential to specify the number of clusters required for this algorithm. K-means is a popular algorithm, because it performs well with many samples. However, since the group validation framework is modular, any grouping could be used in this phase, such as KNN with a Pearson Correlation Coefficient (PCC) method [41] to form groups of similar users, a simple yet effective CF approach. For the k-means method utilized in our experiments, we use the aspects $A$ of the dataset to form a matrix with all functional aspects as features in an $n$-dimensional space where $n$ is the total number of available features in a chosen dataset. Recall that $A$ is the set of item genres used in our dataset experiments: $A = \{a_1, a_2, \ldots, a_c\}$.

### 3.2.2 Group Validation

To validate the resulting groups and determine which performs worst, we follow the general methodology of Reference [21] regarding statistical significance assessment on the group level. The first step is to replace Equation (1) with a suitable evaluation metric relevant to the recommender context. Since the group-validation framework is modular, we implement several ranking-based metrics: $nDCG$, $Precision$, $Recall$, and $\alpha\beta-nDCG$. The $\alpha\beta-nDCG$ is a unified assessment metric for measuring accuracy and diversity proposed by Farpar et al. [14]. It was recently adapted from an information retrieval evaluation metric called $\alpha-n\text{ndcg}$ [42–44] to fit into the recommender ecosystem (just like the adaptation of our group validation proposal). The metric addresses the dilemma of having to choose what types of evaluation metrics (primitive accuracy, classical ranking [6], diversity and novelty for highly related properties [13, 45, 46]) to optimize on, an issue initially presented by Herlocker et al. [17]. Adding to that, there are abundant measures to choose from and many optimization procedures for every chosen method, like simple refinements in this work to one CF approach [47] (an information filtering process using techniques involving collaboration among multiple data forms). Further, this new metric satisfied all the essential axioms of modern evaluation [48], proving to be a significant upgrade from its predecessor $\alpha-n\text{ndcg}$ [43] in the search field. Through both parameters, $\alpha$ and $\beta$, $\alpha\beta-nDCG$ is good at detecting non-optimal item order, aspect distribution and ranking, and topical redundancy accumulation. Additionally, it has been proven in the experiments conducted on the MovieLens 20M dataset [28] to behave very well in terms of discriminative power and robustness to incompleteness. A very brief description of the metric formulation is provided in the following paragraph, where the full details can be reviewed in Reference [14] or the source code [49]. As the authors validated the effectiveness on the same datasets we experimented with, we kept the same tuning achieved for $\alpha$ and $\beta$.

As seen in the above equation, there are two new adapted parameters, $\alpha$ and $\beta$. The $\alpha$ parameter accounts for the possibility of the user being wrong in judgment. Alternatively, the $\beta$ factor defines the confidence in a user’s judgment value, represented by the authors as a smoothing factor accounting for user rating uncertainty. It also exhibits a secondary role that models the user’s eagerness to look at items lower in the ranking (higher $\beta$ implies more relevant items match the user’s interests). First, the probability of an item $i$ contributing to satisfying the user’s interest in an aspect $a_\phi$ is defined as:

$$P(a_\phi|u,i) = \begin{cases} 0 & a_\phi \notin i \\ \alpha(u,i) & \exists r_{u,i} and a_\phi \in i \\ \beta(u,r_{u,i}) & \exists r_{u,i} and a_\phi \in i \end{cases}$$

(4)
The authors then introduce redundancy and novelty by estimating whether or not the user is interested in position $k$ in more items capturing a given aspect after having been shown earlier items in ranking $S = i[0, \ldots, k - 1]$. This is defined as the gain value at position $k$ $G[k]$: 

$$P(a_{\phi}|S) = P(a_{\phi}|u)\Pi_{i\in S}(1 - P(a_{\phi}|u, i))$$ (5)

The cumulative gain computation can be done by the following equation:

$$CG[k] = \sum_{j=1}^{k} G[j]$$ (6)

Following the original nDCG equation that applies a discount factor to penalize documents lower in ranking, the authors then define:

$$DCG[k] = \sum_{j=1}^{k} \frac{G[j]}{\log_2(1 + j)}$$ (7)

Returning to our adaptation and the usage of the evaluation metric $\alpha\beta$-nDCG, Equation (1) will now be adapted differently within the group validation framework and will correspond to the following rank-based metrics nDCG and $\alpha\beta$-nDCG:

$$\psi@k = \frac{DCG[k]}{IDCG[k]}$$ (8)

After that, to identify the groups that a recommender model performs poorly on, we use the mathematical model of Equation (2) with some slight variations to include the new metric of Equation (8):

$$G_{critical} : \psi@k(G) - \psi@k(G') < 0 \ni C'_{critical} = True$$ (9)

Finally, we apply the same tests as in 2 to identify a critical cluster in a recommender dataset as part of the group validation process:

$$C'_{critical} = \begin{cases} \text{t-test} & H_0 : \psi(G) \leq \psi(G') \\ \text{H_1 : } \psi(G) > \psi(G') \\ \text{effect-test} & \phi = \sqrt{2} \times \frac{\psi(G) - \psi(G')}{\sqrt{\sigma^2_G + \sigma^2_{G'}}}; \phi \geq 0.8 \end{cases}$$ (10)

### 3.2.3 Group Weights - A Theoretical Model.

In a real-world scenario, the group validation framework procedure could also work if weights are to be assigned to groups. Specifically, certain groups in a system might be more crucial for performance monitoring due to a specific financial aspect or other significant correlation to certain outlooks. We can expand on the above system (Equations (9) and (10)) and define a threshold beyond which the recommender would be performing poorly on important weight-correlated data groups. This can be represented in the following manner:

$$\sum_{g=1}^{CG} \psi(C_g) \times w_g \leq \lambda = 0.5$$ (11)

In this model, $C_g$ represents one critical group out of the total identified critical groups $CG$, while $w_g$ is a special weight assigned to group $g$. $C'_g$ corresponds to the group’s equivalent metric value similar to $\psi@k(G')$ in Equation (9). The equation’s threshold $\lambda$, currently set to 0.5 as an example, can be further tweaked depending on the system’s defined group weights and the strictness of the validation framework on important groups.
Table 1. Datasets Used in the Experiments of the Group Validation Mechanism

| Dataset       | Total users | Total items | Total ratings | Sparsity |
|---------------|-------------|-------------|---------------|----------|
| ml-latest-small | 610         | 9,742       | 100,836       | 0.983    |
| ml-1m         | 6,040       | 3,900       | 1,000,209     | 0.957    |
| personality   | 1,820       | 35,196      | 1,028,751     | 0.983    |
| ml-25m        | 283,228     | 58,098      | 27,753,444    | 0.998    |
| ml-25m*       | 10,000      | 15,316      | 250,000       | 0.995    |

4 GROUP VALIDATION EXPERIMENTATION METHODOLOGY

In this section, we discuss the application methodology of the group validation framework and explain the experiment setup and procedures. We first cover the algorithms and the data used to conduct the different experiments and their primary objectives. After that, we present three types of data perturbations used to simulate potentially harmful behaviors in the dataset, and finally, we cover the mode of operation of the group validation framework in correlation with the experiments designed to showcase the validation process.

4.1 Data and Algorithms

Generally, recommender systems can be categorized into two broad classes: algorithms optimized for accurate predictions and others optimized for better rankings. Since ranking is more suitable for most use-cases of deployed recommender systems compared to the less popular prediction-focused algorithms [17], and the framework we introduced has ranking-based metrics for the group validation part (recall Equation (9)), we mainly focus on models that are optimized for generating optimal user rankings.

The recommenders chosen for showcasing group validation are **Bayesian Personalized Ranking (BPR)** [50] and **Singular Value Decomposition (SVD)** [51]. BPR uses item pairs $i,j$ and optimizes for the correct ranking given the preference of a user $u$ by maximizing the posterior probability. For the model’s parameters, we use the generic tuning done on the same datasets here [7] with minimal optimization and set $k$ to 400 (dimension of the latent space), $max$-iter to 100 (the number of iterations of the SGD procedure), learning-rate to 0.01 (step size $\alpha$ in the gradient update rules), and lambda-reg to 0.001, which controls the L2-regularization $\lambda$ in the objective function. The final objective function of the maximum posterior estimator (a probabilistic framework for solving the problem of density estimation) is $J = \sum (u, i, j) \in D_{\text{test}} \ln p(x_{uij}) - \lambda \theta ||\theta||^2$. Unlike BPR, the SVD algorithms model the user and item biases from users and items and use **Stochastic Gradient Descent (SGD)** as an optimization technique.

In SVD, when baselines are not used, this is equivalent to Probabilistic Matrix Factorization [52, 53]. All implementation details can be found in this study [53] (SVD module), however, we give a quick review here for clarity:

The prediction $\hat{r}_{u,i}$ is set as:

$$\hat{r}_{u,i} = \mu + b_u + b_i + q_i^T p_u$$

(12)

If the user $u$ is unknown, then the bias $b_u$ and the factors $p_u$ are assumed to be zero. The same applies to item $i$ with $b_i$ and $q_i$. The unknowns are estimated with a regularized squared error:

$$\sum_{r_{ui} \in R_{\text{train}}} (r_{ui} - \hat{r}_{ui})^2 + \lambda (b_i^2 + b_u^2 + ||q_i||^2 + ||p_u||^2)$$

(13)
The minimization is then performed by an SGD:

\[
\begin{align*}
    b_u & \leftarrow b_u + \gamma (e_{ui} - \lambda b_u) \\
    b_i & \leftarrow b_i + \gamma (e_{ui} - \lambda b_i) \\
    p_u & \leftarrow p_u + \gamma (e_{ui} \cdot q_i - \lambda p_u) \\
    q_i & \leftarrow q_i + \gamma (e_{ui} \cdot p_u - \lambda q_i)
\end{align*}
\]

(14)

The formula \( e_{ui} = r_{ui} - \hat{r}_{ui} \) is used to calculate the difference between the actual rating and the predicted rating. To train the model, the steps are performed repeatedly over all the ratings in the training set for a specified number of epochs, denoted as \( n_{epochs} \). At the beginning of the training process, baselines are set to zero. The user and item factors are randomly initialized using a normal distribution, with the mean and standard deviation of the normal distribution being adjustable through the \( \text{init\_mean} \) and \( \text{init\_std\_dev} \) parameters. We can also control the learning rate \( \gamma \) and the regularization term \( \lambda \). We used the default settings of 0.005 and 0.02, respectively.

As for the datasets used in our study, the experiments are conducted on several famous rating-based (explicit feedback) datasets of different sizes, which are summarized in Table 1: MovieLens [28] and personality [54], where smaller counterpart versions used are marked with a (*). The data attributes used from those datasets are the user ratings for training the models and the ratings with the item attributes (such as genres) to perform the grouping.

4.2 Data Perturbations

In this section, we introduce the data modification mechanism and schemes used in the experiments of the group validation framework. First, the data alteration process is described, followed by the types of perturbations applied and how they relate to existing methods covered in the introduction of this work.

4.2.1 Mechanism. Given that the introduced framework tracks the performance of smaller data groups and shows possible clusters of interest that might suffer performance degradation, we try to simulate a case of old and future dataset versions and apply both the normal and the group validation methods to both versions in an attempt to compare and showcase both performances side-by-side. We establish the initial dataset version as the “old state” and a future version with newer data introduced as a “future state” where a recommender model will be refreshed. This is a customary process in live settings where recommender systems must be periodically re-trained to refresh the model weights with new user observations. There are even new proposed efficient ways to optimize this process regarding memory and speed and use advanced ways to transfer knowledge from previously trained-on data. This aids in using much bigger dataset sizes and helps avoid shrinking older data in future train runs to increase performance, as this would cause overfitting and historical forgetting issues [55].

Additional data perturbations applied on the future dataset version are introduced on targeted dataset groups, for example, on the 20th out of the 50 available groups (after the clustering process has been applied). There is no particular reasoning behind a target group selection in the scope of this study; however, it is crucial to note that the general idea of the results was consistent with different chosen groups across several runs [49]. This point is further affirmed in the other experiment parts in the following sections, as we vary the target groups for the synthetic data introduction and lower potential biases by having a small 2% standard data progression, i.e., the data added to a system as users interact with it over time.
4.2.2 Types and Parameters. Data perturbations represent the feedback variations that will be applied to selected user profiles to showcase the possibilities of the group validation framework and position it as an additional layer beside the general evaluation metrics. The two main types of perturbations utilized are arbitrary feedback and aspect reversing, followed by one baseline type, which we refer to as standard feedback. Figure 1 represents the methods of applying arbitrary feedback and aspect-reversing. The general idea behind every type is summarized in the following points:

- Standard: A simple process of holding out a few portions of user data in the initial dataset version and re-applying them as a form of “future” feedback.
- Arbitrary: On the first side, Schematic 1 depicts how arbitrary feedback corresponds to the mechanism of randomly/arbitrarily applying new future ratings for $N$ users, i.e., not following a certain profile’s historical patterns and rating inclinations, which could be framed as an arbitrary form of feedback.
- Aspect reversing: As opposed to the arbitrary scheme, aspect reversing (right side in the diagram), a target aspect is initially selected (such as comedy in the MovieLens dataset cases), and the new future ratings for $N$ users will be introduced in a fashion where their least favorite and most favorite item aspects will be reversed. This would be possible after analyzing the overall per-user item aspect space.

As reviewed in the introduction and the state-of-the-art sections, such feedback patterns like the arbitrary and aspect reversing types exist in the real world and can maintain different forms. For example, they could appear as natural noise, a process described in one of the significant studies of natural noise ratings [56]. This type inspired the arbitrary feedback perturbations and is mainly due to errors in the rating elicitation that create erratic user review data not aligned with their general feedback profile, even when serendipity is accounted for with relaxed parameters [57]. This type of behavior is usually challenging to identify and manage, since it does not have a specific pattern and consequently cannot be modeled [18, 56]. Another real-world form that inspired the aspect reversing is opt-out, sometimes referred to as obfuscation [58, 59]. Users leverage this
Table 2. Group Validation Framework’s Mode of Operation: Data Perturbation Methods and Percentages of the Magnitudes Applied to the Datasets

| Feedback Method     | Min Change | Max Change | Affected Users | Notes               |
|---------------------|------------|------------|----------------|---------------------|
| standard*           | 2%         | 2%         | -              | Normal Progression  |
| arbitrary           | 2%         | 25%        | 10/G           | Arbitrary Progression|
| aspect reversing    | 2%         | 25%        | 10/G           | Reversed Aspects    |

The 2% of standard data progression is implemented on all users, while the arbitrary and aspect reversing methods are only applied on a portion of the specific target group G.

mechanism when an opt-out option is not provided in a system, meaning they are not provided with an option to remove their data from the system’s databases and revert the consent for processing it in machine learning applications. This type is usually prevalent in masking profiles when users want to hide their identity, which could be revealed from registered interactions. Similar data perturbations, in the form of aspect reversing, were used for testing the effectiveness of the $\alpha\beta$-nDCG evaluation metric in Reference [14].

For the arbitrary ratings and the aspect reversing data schemes, we have to set a reasonable limit in the experiments that show how group validation works on evaluating smaller data clusters and how it compares with current evaluation methods. Adding significant percentages can cause unwanted biases that might overshadow the point of showing the mode of operation of validating groups. Generally, any data modification like the ones introduced above should yield worse recommender performances. As long as the percentage of the affected data is high enough, sensitive ranking metrics like the $\alpha\beta$-nDCG (or even nDCG and prediction in some cases [14]) should be able to spot a degradation in performance. With group validation, the framework studies smaller data clusters, so any minor and potentially malicious change could negatively affect the group level (recall Section 3). Since group validation is intended to localize the effect of such small-scale variations, especially their potential malicious effect on different user groups, we select the behavioral synthetic data portions to be smaller in scale than those used to test new metrics like the $\alpha\beta$-nDCG in Reference [14]. The magnitudes applied are summarized in Table 2. The first four magnitudes range between 2% and 15% of added feedback of the total ratings in a target cluster. This is quite similar to what was done in Reference [14] for the metric tests and also in Reference [18], where a similar range was used to introduce data anomalies that exhibit the same characteristics as the arbitrary rating scheme of our experiment. Therefore, we try to utilize a similar structure. In our process, the percentage change is less, since the percentage of added feedback is based on the group size (affecting around 10 users per group—10/G), a subset of the dataset.

4.3 Group Validation Experiment Process

In this section, we cover the mode of operation of the group validation process and go over the experiments presented in the following section. Algorithm 1 [49] represents the high-level code structure of the proposal’s procedure. First, the primary process of the algorithm repeats twice as we target one typical real-world scenario where the data is refreshed with a newer version for offline re-training and batch inferencing (recall Section 4.2.1). Therefore, in this case, the dataset’s future state change would be the addition of the data perturbation schemes alongside the small percentage of standard held-out data. The small held-out data corresponds to the standard rating method introduced in the previous section. It aids in establishing a baseline and better representing a real-world scenario where newer data feedback usually occurs more visibly across a more significant portion of the dataset. Following this method, we can measure the effect different data variations (Section 4.2) might have on the system using standard evaluation mechanisms while on the clusters using the group validation process.
The below process is repeated twice, before and after the change in data $(D$ and $D')$, as described in Section 4

Initialization

$$G_{\text{critical}}, G_{\text{non-critical}} = [[]]$$

$$\lambda = 0.5$$

clusters = apply_clustering($D, R, A$) // all groups and their respective users

train_data, test_data = test_split($D$)

predictions = fit_score($M$, train_data)

for $G$ in clusters do

$$G' = D - G$$ // get the equivalent of $G$

condition_1 = metric($G$) − metric($G'$) // metric is ndcg@$k$ or $\alpha\beta$-ndcg

while condition_1 do

group_status = aspects_test($g, g'$) // apply the tests of Equation (10)

if group_status is True then

$$G_{\text{critical}} \ append(g)$$ // The identified critical groups. The algorithm is performing poorly here

else

$$G_{\text{non-critical}} \ append(g)$$

end

end

$$P_D = evaluate(predictions, test_data)$$

$$P_G = group_evaluate(G_{\text{critical}}, G_{\text{non-critical}}, \lambda)$$

The other parts of the pseudocode in Algorithm 1 mainly follow the theoretical model of Section 4. The initial step is establishing the dataset groups using the k-means method and performing a typical train-test split on a target dataset. The number of clusters for each dataset has been determined in a standard way using the elbow method, resulting in around 50 clusters for smaller datasets and 100 for the bigger ones. General evaluation metrics are obtained using typical evaluation with predicted outcomes and held-out test sets with the evaluation metrics Precision, Recall, nDCG, and so on. At the same time, group validation commences with studying each group and applying the procedures explained in Section 3.2.2.

Recall that the primary aim of the group validation method is to identify which data clusters would be negatively affected by potentially harmful behavioral patterns (in the form of data perturbations) that are typically undetectable by standard evaluation methods. We conducted three experiments to position group validation in the evaluation procedures of recommenders. Initially, we would like to establish whether the data perturbations (representing the previously discussed feedback scenarios) affect only some data groups but not the whole system. This can help us validate the first point introduced in the introduction:

1. Clusters can be more sensitive to certain types of feedback, allowing the framework to detect adverse effects, something not possible solely with generic evaluation.
Fig. 2. Group validation results using nDCG on ml-latest-small with no data perturbations—just standard data (left), arbitrary data perturbations (center), and aspect reversing (right).

The group validation process should be able to identify which groups are negatively impacted by changes in the data, rendering the negative effect more localized and the clusters sensitive to small, potentially malicious changes. Standard metric evaluation on the whole test set might be unaffected by those schemes. To verify this, we conduct two additional experiments to generalize the results further and place both the standard validation methods alongside the group validation process as we test the different data perturbation schemes. Potentially, this can help us affirm the following second main motivation of the work:

(2) Certain user feedback can evolve in a way that negatively affects other users. Cluster evaluation like that in the group validation framework can help localize this effect.

5 RESULTS AND ANALYSIS

In this section, we present a walk-through and further analysis of the experiments and findings of the group-based validation framework following the theoretical model and framework guidelines introduced in Section 3 and the experiment methodology presented in Section 4.

5.1 Critical Groups Due to Data Perturbations and the Simpson’s Paradox

The first experiment is split into three parts, and for brevity, we present the results on one of the test datasets. In the first trial, we check the regular evaluation and group validation framework on standard data review future values (devoid of any perturbations) to establish a baseline result and verify whether or not we will encounter critical groups. This permits us to show the mode of operation of group validation against other typical evaluation methods when users provide standard data feedback consistent with their profile predilections without any potentially harmful data between the two dataset versions. The second and third tests will have the data perturbations in two forms: arbitrary feedback and profile swapping. As explained in the methodology, we have the standard data scheme of 2% regular data feedback in the three tests as a form of natural data progression between the current and future dataset versions. For the arbitrary and aspect reversing, in the last two tests of this experiment, we use a 10% rating feedback (of a target cluster’s total item feedback) as a data perturbation magnitude on a random target cluster.

Figure 2 shows the group validation framework’s outcome and mode of operation. The first test results are displayed in the left plot. The almost straight line represents the values of the core metric of the group validation of all the equivalent examples of every group $G'$ and is denoted by group-nDCG-eq. In contrast, the scattered points in blue represent n-DCG values of the groups of interest $G$ and are labeled as group-nDCG. The plot shows a generic distribution of the groups in an acceptable zone of ranked item performance. This means that not even the first condition of the group validation model in Equation (9) was met. Only two groups slightly appear below their $G'$ counterpart value, thus validating the first condition of Equation (9), but the second condition has not been met, and therefore they are still considered in the acceptable zone with a
non-critical-status. The overall system metric registered an nDCG value of approximately 0.338, coinciding with the group-nDCG-eq values of each respective $G'$. We can imply that this establishes a stable state of the dataset with all groups having acceptable performance results in a case where none of the two primary perturbation schemes are applied.

In this experiment’s second and third tests, displayed in the middle and the right plots of Figure 2, we introduce arbitrary rating and aspect-swapping data perturbations, respectively. As described, this is done on a target cluster to create a new future dataset version where the model will be trained and re-evaluated. Similar percentages are utilized: 2% standard progression for all clusters to simulate the normal data evolution at two points in time and a 10% data perturbation on another cluster number (which was group number 20 this time). It is clear from the plot how group validation identifies critical groups (presented as squares) following the theoretical model in Equations (9) and (10). Those groups are, therefore, negatively affected by the small-magnitude perturbations introduced in only one of the clusters. The results of the third test are not very different: The system metric maintained almost the same value, 0.336 versus 0.333, while several critical groups were reported. Combining the results in both cases of the groups results in the overall nDCG system value. It will diminish the negative effect spotted on the groups shown in the plots—which the group validation framework translated into critical groups. This renders a Simpson’s paradox scenario in effect (recall Section 2.3). For instance, a drop of 0.9% in nDCG in a re-train/re-evaluation scenario such as this one evaluation will not be alarming if a model is regularly re-trained for offline batch-inferencing.

Recall that this experiment aims to prove that there will be negatively affected users in a dataset whenever there is undetectable and malicious behavioral data. The above tests show how group validation with nDCG as the core evaluation metric of Equation (9) was able to identify critical groups with a 10% data perturbation magnitude. This validates the first goal (Goal 1 in Section 4.3) outlined in the introduction of this work. Some groups are sensitive to specific changes in the dataset, and the group evaluation process detected the effect.

Additionally, reiterating the connection with the concept of Simpson’s paradox, we proved how the paradox explained the results obtained in our experiments where the group validation mechanism was able to spot and report smaller versions of the data (in the form of clusters/groups) where the model’s performance is negatively affected. This negative performance will not be noticeable when we evaluate the system with legacy evaluation methods.

5.2 Critical Groups versus Normal Evaluation

In our second experiment, we study the effect of the chosen data perturbations on the group validation method alongside the normal model evaluation using the evaluation metrics nDCG, precision, recall, and $\alpha\beta$-nDCG. This second test expands on the previous results and tests the difference between measuring the system’s performance and the group’s performance as the data evolves. For this experiment, we select two datasets, the ml-latest-small and the ml-1m, with the two algorithms, BPR and SVD. Data perturbation aspect reversing is used, and the method follows the same strategy as in the first experiment for the 2% normal data progression between the two dataset versions on which the tests will be applied. However, in this case, we vary the magnitude intensity and define five levels ranging from 2% to 25% of the group’s total ratings. The target group for the data perturbation has also been randomly selected: 25th (out of the entire 50 groups) for the ml-latest-small and 70th (out of 150) for the ml-1m. This test aims to validate that the malicious data perturbations with various percentages will not significantly affect metric results contrary to the group validation method. Similarly, the group validation result is used for the evaluation results with nDCG and precision as the core evaluation metrics for the group validation process of Equation (9) in the first test and nDCG for the second and upcoming ones.
Fig. 3. The percentage of critical groups alongside normal metric scores as data perturbation percentage (aspect reversing scheme) increases on ml-latest-small (left) and ml-1m (right). With increased perturbation magnitudes, modest metric effects are observed while critical groups increase, as spotted by the group validation process.

Fig. 4. The percentage of critical groups alongside normal metric scores as aspect reversing intensity increases on ml-latest-small (left) and ml-1m (right) using the SVD algorithm.

Figures 3 and 4 display the outcome of the metrics on the 5-magnitude data perturbations for the BPR and the SVD algorithms, respectively. We first notice that the $\alpha\beta$-nDCG metric scored slightly higher in both tests than its counterpart nDCG. This indicates a somewhat better metric value on the rankings achieved by the BPR model with the likelihood and aspect weights (recall Section 3.2.2). As the intensity of perturbations increases from 2% to 25%, we notice a modest decrease in the ranking metrics, slightly sharper for the $\alpha\beta$-nDCG, especially in the ml-1m case. For ml-latest-small, until the 15% data perturbation magnitude, the decrease in $\alpha\beta$-nDCG registered a mere 5%, while for nDCG and precision, the score is a 3% decrease only. However, the recall metric values maintained a consistent outcome of around 0.065 for ml-latest-small and 0.12 for ml-1m. Analyzing the group validation outcome, we notice that critical groups significantly increase in both cases with increased perturbation magnitudes. This indicates that even though our system metrics still show acceptable results, a significant number of affected users in the dataset are overshadowed by the effect reversal once the results are combined to generate the system metric. Even with the 15% data perturbations, we can see that ml-latest-small still registered around 18%-24% (precision and nDCG, respectively) from the 50 total groups. The same appears for ml-1m but with a slightly lower intensity where 10% of the dataset groups are now in a critical state and will potentially experience degraded recommender ranking performance.

Figure 4 shows the same results but uses the SVD model as a recommender. The results appear to be not very different from the BPR model except for generally lower system metric values. The
Fig. 5. The effect of different data perturbation magnitudes on the percentage of critical group formation and metric scores. The schemes used are random (left column) and aspect reversing (right column) on the three datasets: ml-latest-small, ml-25M, and personality.

$\alpha \beta$-nDCG still scores slightly higher for the same reason, while metrics exhibit a modest decrease in the ml-1m case. It is marginally different in the ml-latest-small, where we can see an increase in metrics performance as the perturbation magnitudes increased before a decrease on the last more-intensified level of 25%. Critical groups expanded from around 5% of the total groups to 10% for the highest perturbation magnitude applied. The group validation, however, still reported critical groups that increased with the increase of the perturbation magnitude, with results very close to those achieved with the BPR model on the ml-1m dataset. The same conclusions can be drawn for this test: Even though our system metrics still show acceptable results, a significant number of affected users in the dataset are overshadowed by the reversal of the effect once the results are combined to generate the system metric.

5.2.1 Extended Analysis - Possible Group Validation Break-point. We extend the experiments and test the group validation method alongside the same metrics on different datasets to further validate the previous test’s results and better represent the group validation versus the typical evaluation methods. Figure 5 depicts the outcome of this experiment on ml-latest-small, ml-25M, and personality (top to bottom). The outcome of the first two datasets is consistent with the previous results and theory. The critical groups increase as the magnitude of data perturbations increases from 2% to 25%. The random perturbations result in slightly lower percentages of critical
groups, most likely due to the random variability of the feedback where some random items might be relevant to a user’s profile. The aspect reversing would still be a more targeted data change compared to the random feedback process and is guaranteed to result in a high level of “non-relevance” for the targeted user [14]. Conversely, metric results showed humble decreases until the 15% magnitude of data perturbations, where the metrics start to indicate a slightly higher decline in performance. At this point, critical groups registered the highest number for all three datasets and reached around 30% for the first two. Personality is a richer dataset with many more attributes and reliable feedback. We can notice from the figure that the performance is significantly higher for the metrics compared to the other datasets with the same BPR model and parameter tuning used in the second experiment. The critical group percentage is slightly lower, which is equally consistent with the metric results; however, group validation can still identify affected clusters. Analyzing the overall graph patterns, defining a breaking point at the 4th level representing a 15% change (marked with a vertical line in the figure’s plots) of data perturbations would seem possible. At this stage, potentially malicious change of up to 20% of the target cluster’s total feedback would not be reflected in the system’s normal evaluation metrics (recall the data operations defined in Table 2).

Given the above-obtained results, we can see how the group validation tests helped us localize the adverse effects of potentially harmful behavioral feedback in the form of critical groups, thus affirming the motivation set in the second goal (Goal 2 in Section 4.3).

5.2.2 General Conclusions - Group Validation versus Normal Evaluation. The second section of the experiments better presented the mode of operation of the group validation process side-by-side with normal evaluation metrics in a scenario where data evolves from one state to the other in several datasets. The evolution is obtained with a slight uniform data progression combined with the aspect reversing data perturbation with an increased magnitude from 2% of rating data of a target cluster to 25% of its feedback data. The results show how particular user feedback can evolve in a way that negatively impacts users of specific clusters more than others, affirming the second motivation behind this proposal. Cluster evaluation like that in the group validation framework helps localize this effect and reports where the performance specifically degrades. In contrast, the normal system evaluation generally exhibits Simpson’s paradox effect. Negative results are balanced out by other higher performance results that might be due to biases resulting from the newly added information. Additionally, the small perturbations that affect one group could degrade the performance of the target group and other groups in the dataset. This further affirms the second point of the motivation behind this method, where the feedback can evolve in a way that negatively impacts other users.

Applying the method’s validation procedures to monitor different groups’ performance evolution of a dataset is not a replacement for the general evaluation procedure. Returning to the introduction of this work, we aim to have this mechanism run in parallel with the typical evaluation techniques, as it can apply evaluation procedures on recommenders as the data evolves from one to the other from a different vantage point. It complements the general evaluation results and helps create a more robust filtering process. We further review this method’s possible applications and extensions in Section 6.

5.3 Varying Group Sizes
In this final test, we analyze the dataset’s effect on different group options. As mentioned earlier, the grouping method we apply is an example procedure that could be adapted to a different mechanism in a distinct setting, depending on the features available, the metadata type, and the recommender’s primary goal. However, we can still vary the number of groups with the k-means
method we selected and study how different cluster values can affect the critical groups’ outcome of the group validation method. The results are summarized in Figure 6, which shows the percentage of critical groups as we vary the total number of clusters on the ml-latest-small dataset in 9 unique runs. For this test, we utilized a constant amount of 15% random data perturbations in every run to maintain a coherent scenario across the runs. With a small number of groups (such as 10), we can notice a higher percentage of critical groups. This phenomenon implies that group validation can identify a negative effect in one of the dataset groups; however, since the group number might be small, the negative effect is not very well localized and points to a group containing a relatively significant number of users. When the number of groups increases, we can notice that the critical clusters start to decrease. We conclude from this that we can spot the negative impact of the data perturbations in a more localized cluster in the dataset. As the number of clusters decreases, we are approaching the case of the normal system metrics, i.e., evaluating the effect on the whole dataset. The correct amount of groups is subjective and should be solely based on the dataset attributes and the recommender’s goal. We provided a small theoretical adaptation of the group validation method if weights are to be assigned to groups in Section 3.2.3.

6 GROUP VALIDATION LIMITATIONS AND POSSIBLE APPLICATIONS

In this section, we highlight some of the limitations of our work and the possible applications enabled by the new group validation framework in the recommender system ecosystem.

6.1 Limitations

The data clustering in our experiments was done based on the available item attributes of the datasets at hand. It would be interesting to investigate how the group-based evaluation approach would behave when the clustering is conducted based on a variation of different features, such as user behavior in the system. It is crucial to assess the behaviors using different cluster forms to generalize the approach to various scenarios where recommender systems are used. As different data types and features lead to the usage of different techniques in recommender systems [37, 60], this can be extended to form different strategies of group formations based on rating types (e.g., implicit or explicit) and features.

6.2 Applications

The different parts of this framework, such as the core metric used to identify the critical groups and the means of generating clusters, are interchangeable. This renders the foundation a little
Fig. 7. This schematic shows a hypothetical hybrid model setup of recommenders systems that leverages the group validation framework.

more flexible and open to further experimentation. Some of the potential applications that could be implemented on top of the group validation framework are listed below:

— Model Evolution and Fairness. With the recent vital importance of fair recommender systems [23], it is crucial to report and analyze the performance of a specific group of users. Group validation spots a localized form of negative effects that could result from potentially harmful behavioral data. This can be an initial step in forming a metric better optimized to increase fairness in a system and monitor it across different model generations.

— Noise/Fraud Detection. The detection of fraud (which sometimes can be referred to as noise in a simpler form) involves identifying malicious behaviors that form special patterns. Popular methods for their detection involve Graph-based anomaly detection (GBAD) [61], primarily used to analyze connectivity patterns in communication networks and identify suspicious behaviors. The proposed group validation framework can be further tweaked to provide a new layer for anomaly detection in recommender systems. The granular evaluation outlook could be sophisticated enough to detect weaknesses in performance and spot malicious behavior that forms a unique connection between different groups. A similar concept has also been debated by Al Jurdi et al. [18] in the study of natural noise, where it was shown that certain malicious patterns could affect parts of the system.

— Hybrid Model Decision System. Building on the evaluation method introduced in this work, the architecture could also be extended to include multiple models deployed in a production environment where each would be tuned for different groups based on the results from the group validation framework. Figure 7 shows a flow diagram of this hypothetical application. The critical clusters could be transferred to a grouping mechanism that arranges them based on the metrics portfolio similar to that employed in the experiments of Section 5. This potentially useful application aids in limiting degrading performance results in small groups.

7 CONCLUSION

Evaluating the effectiveness of recommendation systems and testing their performance as the underlying data evolves with time remains a remarkably tough challenge yet to be tackled. Model evaluation in the current state of the art has reported some improvement in the benchmarking field, where researchers created several evaluation frameworks to standardize the assessment process. However, the recommender ecosystem has not covered the concept of reporting the performance of models on dataset subgroups and the potential of detecting specific kinds of potentially malicious data behaviors. This chapter described a new evaluation strategy for recommenders: group validation framework in recommenders. This method can be employed as an assessment tool to track the performance of a recommender on certain important clusters/groups of data. The results showed how recommenders performed differently in various groups as unique data perturbations were introduced. The most fundamental aspect is that we could spot negatively
affected sections of the dataset due to added synthetic data that was not visible with normal evaluation techniques. Group validation helped localize the errors in the data and provided a means to identify the effect of behavioral data changes in the system.

In the future, this proposed framework can be extended to cover other scenarios where dataset grouping could be effectively applied to several recommender dataset types (such as implicit feedback) and embed different recommender goals. Further, a data-oriented approach can be introduced to have a more intelligent grouping mechanism. It can automatically treat potential performance degradation in specific clusters and measure the effect on serendipity as a factor of such changes.

REFERENCES

[1] Alejandro Bellogín and Alan Said. 2021. Improving accountability in recommender systems research through reproducibility. User Model User-Adap Inter 31 (2021), 941–977.
[2] Craig Macdonald. 2021. The Simpson’s Paradox in the offline evaluation of recommendation systems. ACM Transactions on Information Systems 40, 1 (2021).
[3] Zohreh Ovaisi, Shelby Heinecke, Jia Li, Yongfeng Zhang, Elena Zheleva, and Caiming Xiong. 2022. RGRRecSys: A toolkit for robustness evaluation of recommender systems. arXiv preprint arXiv:2201.04399 (2022).
[4] Di Han, Yifan Huang, Xiaotian Jing, and Junmin Liu. 2021. AND: Effective coupling of accuracy, novelty and diversity in the recommender system. In 17th International Conference on Mobility, Sensing and Networking (MSN’21). IEEE, 772–777.
[5] N. Ferro, Norbert Fuhr, Gregory Grefenstette, Joseph A. Konstan, Pablo Castells, Elizabeth M. Daly, Thierry Declerck, Michael D. Ekstrand, Werner Geyer, Julio Gonzalez, Tsiv Kuflik, Krister Lindén, Bernardo Magnini, Jian-Yun Nie, R. Perego, Bracha Shapira, Ian Soboroff, Nava Tintarev, Karin M. Verspoor, Martijn C. Willemsen, and Justin Zobel. 2018. The Dagstuhl Perspectives Workshop on performance modeling and prediction. In ACM SIGIR Forum, Vol. 52. ACM New York, NY, 91–101.
[6] Daniel Valcarce, Alejandro Bellogín, Javier Parapar, and Pablo Castells. 2018. On the robustness and discriminative power of information retrieval metrics for top-N recommendation. In 12th ACM Conference on Recommender Systems. 260–268.
[7] Andreas Argyriou, Miguel González-Fierro, and Le Zhang. 2020. Microsoft recommenders: Best practices for production-ready recommendation systems. In the Web Conference. 50–51.
[8] Zhu Sun, Di Yu, Hui Fang, Jie Yang, Xinghua Qu, Jie Zhang, and Cong Geng. 2020. Are we evaluating rigorously? Benchmarking recommendation for reproducible evaluation and fair comparison. In 14th ACM Conference on Recommender Systems. 23–32.
[9] Vito Walter Anelli, Alejandro Bellogín, Antonio Ferrara, Daniele Malitesta, Felice Antonio Merra, Claudio Pomo, Francesco Maria Donini, and Tommaso Di Noia. 2021. Elliot: A comprehensive and rigorous framework for reproducible recommender systems evaluation. In 44th International ACM SIGIR Conference on Research and Development in Information Retrieval. 2405–2414.
[10] Wayne Xin Zhao, Shanlei Mu, Yupeng Hou, Zihan Lin, Yushuo Chen, Xingyu Pan, KIayuan Li, YuJie Lu, Hui Wang, Changxin Tian, Yingqian Min, Zhichao Feng, Xinyan Fan, Xu Chen, Pengfei Wang, Wendi Ji, Yaliang Li, XiaoQing Wang, and Ji-Rong Wen. 2021. RecBole: Towards a unified, comprehensive and efficient framework for recommendation algorithms. In 30th ACM International Conference on Information & Knowledge Management. 4653–4664.
[11] Jin Yao Chin, Yile Chen, and Gao Cong. 2022. The datasets dilemma: How much do we really know about recommendation datasets? In 15th ACM International Conference on Web Search and Data Mining. 141–149.
[12] Yan-Martin Tamm, Rinchin Damdinov, and Alexey Vasilev. 2021. Quality metrics in recommender systems: Do we calculate metrics consistently? In 15th ACM Conference on Recommender Systems. 708–713.
[13] Pablo Castells, Neil Hurley, and Saul Vargas. 2022. Novelty and diversity in recommender systems. In Recommender Systems Handbook. Springer, 603–646.
[14] Javier Parapar and Filip Radlinski. 2021. Towards unified metrics for accuracy and diversity for recommender systems. In 15th ACM Conference on Recommender Systems. 75–84.
[15] Marina Drosou and Evaggelia Pitoura. 2010. Search result diversification. ACM SIGMOD Rec. 39, 1 (2010), 41–47.
[16] Hui Fang and ChengXiang Zhai. 2005. An exploration of axiomatic approaches to information retrieval. In 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. 480–487.
[17] Jonathan L. Herlocker, Joseph A. Konstan, Loren G. Terveen, and John T. Riedl. 2004. Evaluating collaborative filtering recommender systems. ACM Trans. Inf. Syst. 22, 1 (2004), 5–53.
[18] Wissam Al Jurdi, Jacques Bou Abdo, Jacques Demerjian, and Abdallah Makhoul. 2021. Critique on natural noise in recommender systems. ACM Trans. Knowl. Discov. Data 15, 5 (2021), 1–30.
[19] H. Brendan McMahan, Gary Holt, D. Sculley, Michael Young, Dietmar Ebner, Julian Grady, Ian Nie, Todd Phillips, Eugene Davydov, Daniel Golovin, Sharat Chikkerur, Dan Liu, Martin Wattenberg, Arnar Mar Hafinnelsson, Tom Boulos, and Jeremy Kubica. 2013. Ad click prediction: A view from the trenches. In 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 1222–1230.

[20] Yeounoh Chung, Tim Kraska, Steven Euijung Whang, and Neoklis Polyzotis. 2018. Slice finder: Automated data slicing for model interpretability. In SysML Conference.

[21] Yeounoh Chung, Tim Kraska, Neoklis Polyzotis, Ki Hyun Tae, and Steven Euijung Whang. 2019. Automated data slicing for model validation: A big data-AI integration approach. IEEE Trans. Knowl. Data Eng. 32, 12 (2019), 2284–2296.

[22] Yeounoh Chung, Tim Kraska, Neoklis Polyzotis, Ki Hyun Tae, and Steven Euijung Whang. 2019. Slice finder: Automated data slicing for model validation. In IEEE 35th International Conference on Data Engineering (ICDE’19). IEEE, 1550–1553.

[23] Ashudeep Singh and Thorsten Joachims. 2018. Fairness of exposure in rankings. In 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. 2219–2228.

[24] Thomas Cover and Peter Hart. 1967. Nearest neighbor pattern classification. IEEE Trans. Inf. Theor. 13, 1 (1967), 21–27.

[25] Daniel B. Neill. 2015. Subset Scanning for Event and Pattern Detection. Springer, International Publishing, Cham, 1–10.

[26] Daniel B. Neill. 2012. Fast subset scan for spatial pattern detection. J. R. Stat. Soc. Series B: Stat. Methodol. 74, 2 (2012), 337–360.

[27] F. Maxwell Harper and Joseph A. Konstan. 2015. The MovieLens datasets: History and context. ACM Trans. Interact. Intell. Syst. 5, 4 (2015), 1–19.

[28] Wikipedia contributors. 2022. Welch's t-test—Wikipedia, The Free Encyclopedia. Retrieved from: https://en.wikipedia.org/w/index.php?title=Welch%27s_t-test&oldid=1091289937

[29] Wikipedia contributors. 2022. Effect Size — Wikipedia, The Free Encyclopedia. Retrieved from: https://en.wikipedia.org/w/index.php?title=Effect_size&oldid=1102056368

[30] Gail M. Sullivan and Richard Feinn. 2012. Using effect size—or why the P value is not enough. J. Grad. Medic. Educ. 4, 3 (2012), 279–282.

[31] Jacob Cohen. 2013. Statistical Power Analysis for the Behavioral Sciences. Routledge.

[32] Bart P. Knijnenburg, Martijn C. Willemsen, Zeno Gantner, Hakan Soncu, and Chris Newell. 2012. Explaining the user experience of recommender systems. User Model. User-adapt. Interact. 22, 4 (2012), 441–504.

[33] R. Logesh, V. Subramaniyaswamy, D. Malathi, N. Sivaramakrishnan, and Varadarajan Vijayakumar. 2020. Enhancing recommendation stability of collaborative filtering recommender system through bio-inspired clustering ensemble method. Neural Comput. Appl. 32, 7 (2020), 2141–2164.

[34] Rishabh Ahuja, Arun Solanki, and Anand Nayyar. 2019. Movie recommender system using K-Means clustering and K-Nearest Neighbor. In 9th International Conference on Cloud Computing, Data Science & Engineering (Confluence’19). IEEE, 263–268.

[35] Qian Zhao, F. Maxwell Harper, Gediminas Adomavicius, and Joseph A. Konstan. 2018. Explicit or implicit feedback? Engagement or satisfaction? A field experiment on machine-learning-based recommender systems. In 33rd Annual ACM Symposium on Applied Computing. 1331–1340.

[36] Saúl Vargas. 2015. Novelty and diversity evaluation and enhancement in recommender systems and information retrieval evaluation. In 31st Annual International ACM SIGIR Conference on Research & Development in Information Retrieval. 659–666.

[37] Saúl Vargas. 2014. Novelty and diversity enhancement and evaluation in recommender systems and information retrieval. In 37th International ACM SIGIR Conference on Research & Development in Information Retrieval. 1281–1281.

[38] Saúl Vargas. 2015. Novelty and Diversity Evaluation and Enhancement in Recommender Systems. PhD dissertation, Universidad Autónoma de Madrid, Spain.
[45] Li Chen, Yonghua Yang, Ningxia Wang, Keping Yang, and Quan Yuan. 2019. How serendipity improves user satisfaction with recommendations? A large-scale user evaluation. In the World Wide Web Conference. 240–250.

[46] Miriam El Khoury Badran, Jacques Bou Abdo, Wissam Al Jurdi, and Jacques Demerjian. 2019. Adaptive serendipity for recommender systems: Let it find you. In International Conference on Agents and Artificial Intelligence (ICAART’19). 739–745.

[47] Wissam Al Jurdi, Chady Abou Jaoude, Miriam El Khoury Badran, Jacques Bou Abdo, Jacques Demerjian, and Abdallah Makhoul. 2019. SCCF parameter and similarity measure optimization and evaluation. In International Conference on Knowledge Science, Engineering and Management. Springer, 118–127.

[48] Enrique Amigó, Damiano Spina, and Jorge Carrillo-de Albornoz. 2018. An axiomatic analysis of diversity evaluation metrics: Introducing the rank-biased utility metric. In 41st International ACM SIGIR Conference on Research & Development in Information Retrieval. 625–634.

[49] Wissam Al Jurdi. 2022. Group Validation in Recommender Systems: Framework for Multi-layer Performance Evaluation. Retrieved from: https://github.com/wissamjur/group-validation

[50] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2012. BPR: Bayesian personalized ranking from implicit feedback. arXiv preprint arXiv:1205.2618 (2012).

[51] Simon Funk. 2006. Netflix Update: Try This at Home. Retrieved from: https://sifter.org/~simon/journal/20061211.html

[52] Andriy Mnih and Russ R. Salakhutdinov. 2007. Probabilistic matrix factorization. Adv. Neural Inf. Process. Syst. 20 (2007).

[53] Nicolas Hug. 2020. Surprise: A Python library for recommender systems. J. Open Source Softw. 5, 52 (2020), 2174. DOI: http://dx.doi.org/10.21105/joss.02174

[54] Tien T. Nguyen, F. Maxwell Harper, Loren Terveen, and Joseph A. Konstan. 2018. User personality and user satisfaction with recommender systems. Inf. Syst. Front. 20, 6 (2018), 1173–1189.

[55] Yang Zhang, Fuli Feng, Chenxu Wang, Xiangnan He, Meng Wang, Yan Li, and Yongdong Zhang. 2020. How to retrain recommender system? A sequential meta-learning method. In 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval. 1479–1488.

[56] Raciel Yera Toledo, Yailé Caballero Mota, and Luis Martinez. 2015. Correcting noisy ratings in collaborative recommender systems. Knowl.-based Syst. 76 (2015), 96–108.

[57] Wissam Al Jurdi, Miriam El Khoury Badran, Chady Abou Jaoude, Jacques Bou Abdo, Jacques Demerjian, and Abdallah Makhoul. 2018. Serendipity-aware noise detection system for recommender systems. In Proceedings of the Information and Knowledge Engineering 1 (2019), 107–114.

[58] Finn Brunton and Helen Nissenbaum. 2015. Obfuscation: A User’s Guide for Privacy and Protest. MIT Press.

[59] Wissam Al Jurdi, Jacques Bou Abdo, Jacques Demerjian, and Abdallah Makhoul. 2022. Strategic attacks on recommender systems: An obfuscation scenario. In IEEE/ACS 19th International Conference on Computer Systems and Applications (AICCSA'22). IEEE, 1–8.

[60] Deepiyoti Roy and Mala Dutta. 2022. A systematic review and research perspective on recommender systems. J. Big Data 9, 1 (2022), 59.

[61] Tahereh Pourhabibi, Kok-Leong Ong, Booi H. Kam, and Yee Ling Boo. 2020. Fraud detection: A systematic literature review of graph-based anomaly detection approaches. Decis. Supp. Syst. 133 (2020), 113303. DOI: http://dx.doi.org/10.1016/j.dss.2020.113303

Received 1 December 2022; revised 12 November 2023; accepted 22 December 2023