Introducing a framework and a decision protocol to calibrated recommender systems

Diego Corrêa da Silva · Frederico Araújo Durão

Accepted: 29 April 2023 / Published online: 20 June 2023
© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2023

Abstract
Recommender Systems use the user’s profile to generate a recommendation list with unknown items to a target user. Although the primary goal of traditional recommendation systems is to deliver the most relevant items, such an effort unintentionally can cause collateral effects, including low diversity and unbalanced genres or categories, benefiting particular groups of categories. This paper proposes an approach to create recommendation lists with a calibrated balance of genres, avoiding disproportion between the user’s profile interests and the recommendation list. The calibrated recommendations consider the relevance and the divergence between the genre distributions extracted from the user’s preference and the recommendation list. The main claim is that calibration can contribute positively to generating fairer recommendations. In particular, we propose a new trade-off equation, which considers the users’ bias to provide a recommendation list that seeks the users’ tendencies. Moreover, we propose a conceptual framework and a decision protocol to generate more than one thousand combinations of calibrated systems to find the best combination. We compare our approach against state-of-the-art approaches using multiple domain datasets and evaluate using rank and calibration metrics. The results indicate that the trade-off, which considers the users’ bias, produces positive effects on precision and fairness, thus generating recommendation lists that respect the genre distribution. For the decision protocol, we also found the best system for each dataset.

Keywords Calibration · Fairness · Framework · Protocol · Recommender system

1 Introduction

Usually, recommender systems seek to recommend the most relevant items according to the user’s profile. The recommendation is provided as a list ordered by the most relevant items, i.e., the more relevant, the more on top. However, the focus on just recommending relevant items brings some drawbacks, such as low diversity of categories [1], unbalanced genres [2], group consensus [3, 4], popularity bias [5], or miscalibration [6]. Therefore, in this paper, we investigate the calibration recommendation context.

A calibrated recommender system seeks to create a recommendation list, following the proportion of each area (genres, classes, or tags) in the user’s preferences. To this end, the system needs to understand all user’s interests as a distribution. This distribution will be used as the target for the recommendation list, designed to be as little divergent as possible. When the recommendation list distribution is highly divergent from the user’s preferences distribution, the state-of-the-art calls it miscalibration [6–8]. The miscalibration could mean the system fails to provide accurate, personalized items and fair treatment of the classes in the user’s preferences [2, 8]. Thus, a calibrated system creates a recommendation list considering relevant items and a fair representation of the user’s interest classes. The state-of-the-art defines calibration as a way to provide some degree of fairness in which all user’s interests are respected [2, 6, 8].

As an example of calibration, a user’s preferences of 60% Samba, 30% Rock, and 10% K-Pop will create a recommendation list with these proportions in the song context.
However, some divergence is expected between the user’s preferences and recommendation list proportions. In the example, it can be 56% Samba, 31% Rock, and 13% K-Pop. As introduced, the lower the divergence, the better the calibration.

From an implementation point-of-view, the calibrated recommendations [6] are described as post-processing, which rearranges the recommendation list from the recommender algorithm. The state-of-the-art (Sect. 6) develops many implementations. Each work is focused on its proposals with workflows assigned by the system specialist. Based on that, our first research claim is a conceptual framework. We use the pre-processing, processing, and post-processing pattern, commonly used in the field [9], to decompose the calibrated system in these three steps and twelve components (Sect. 2). To test the framework reproducibility, we implement 1092 calibrated recommendation systems (Sect. 4). In this sense, our second proposal is based on implementing many calibrated systems, and deciding which is the best to implement is hard for the system specialist. To make the decision manually, the specialist needs to observe and analyze each implemented system’s degree of precision and calibration. So, we introduce a new tool, the decision protocol, and its coefficients, to decide automatically and indicate the best system for each dataset (Sect. 3). Our third claim is a new formula to control the degree of relevant items and divergence (between the user’s preferences and the user’s recommendation list).

From the claims, we will conduct the discussion following the research questions (Sect. 5):

- **RQ1** Do the calibrated systems perform similarly on the datasets? Are there differences in the system’s behavior?
- **RQ2** How to find the best-calibrated system for each domain?
- **RQ3** Is it possible to design a common pattern to explain (framework model) calibrated recommendation systems? What are the important components of these systems?
- **RQ4** When the user’s bias is considered in the calibration trade-off, is it possible to improve the system performance?

The objective of designing a conceptual framework is connected with RQ1 and RQ3. Following the framework, it is possible to create many recommenders systems. These systems achieve different results due to their formulation. So, are these results influenced by the dataset, or the implemented system can achieve the same results using other datasets? The objective of using the decision protocol as a tool for the system specialist is related to RQ2: finding the best-calibrated system in each domain. After a deep analysis of the results, its objective is presented, and then we use the decision protocol. The objective of implementing a new trade-off balance that considers the user’s bias is connected with the RQ4, which asks about the implementation performance compared with other implementations.

This paper is organized as follows: Sect. 2 introduces our framework proposal of calibrated recommendations and discusses the state-of-the-art contributions. Section 3 presents the decision protocol with its debate and equations. Section 4 illustrates the experimental setup, presenting the used datasets, algorithms, and methodology. Section 5 presents the results from the experiment, discussing the metrics connection, the decision protocol, and the research questions. Section 6 discusses the related work, comparing their implementation with ours. Finally, Sect. 7 ends this paper, presenting a conclusion and some future works.

## 2 Calibrated recommendations

This section covers the first part of our claims, the conceptual framework for calibrated recommendation systems. The framework is inspired by other studies [2, 6, 8, 10] that exploit singular points of the calibrated system. These works do not present a generalized description of the calibrated system. The lack of specifications to produce modular systems may lead the specialist to avoid exploring new possibilities and improving the system.

In addition to the conceptual framework, a new equation is proposed to balance the list of recommendations with relevant and fair items. As we proceed in the section, we describe the framework and introduce the equations used in our experiments.

### 2.1 The framework

As introduced, we claim a conceptual framework for calibrated recommendations. The framework is systematically divided into stages and components described in order.

#### 2.1.1 System steps

The calibrated recommendation systems can be described in three major stages: pre-processing, processing, and post-processing. Each one has its restricted function in the system. The final stage is the most important because that is where the fairer list of recommendations is created. Each stage is outlined as follows:

1. **The pre-processing** deals with data treatment. This first step receives a raw input data set and returns the one that the calibrated recommendation system can use. The usable data have the requirements to be used during the process stage. In Sect. 4.1, we present the two datasets, showing their numbers of users, items, and interactions before and after the pre-processing:
2. **Processing** is the stage at which the recommender algorithm is implemented. It is common for recommendation systems to produce a recommendation list with the recommender output. However, in calibrated recommendation systems, the recommendation algorithm’s output is the post-processing input. In this paper, we refer to the recommendation list created by the recommender algorithm as candidate items. In Sect. 4.2, we discuss the seven recommenders used in our experiments;

3. The **post-processing** is responsible for receiving the candidate items and using them to create the fair recommendation list. In particular, this stage is described in this section.

### 2.1.2 System components

The **pre-processing** step is divided into three components: cleaning, filtering, and modeling. Our framework defines three core components to the pre-processing due to the calibration requiring a class/genre in addition to the user interactions (rating/likes). The cleaning component removes data that the calibrated system, such as missing and incorrect information, cannot use. The filtering component blocks data that can provide noise to the system. In the end, the pre-processing models the data to be used in the processing step.

For instance, Table 1 provides an example of songs modeled under our conceptual framework. The information used to calibrate will be the genre of the songs. Table 2 presents three users and their information. We only use users’ preferences, i.e., the interactions with the recommendation system. It is possible to apply other users’ information depending on the recommendation technique. However, our framework is ready to work in collaborative filtering [6, 8] and content-based filtering [11]. Table 3 shows the preferences of three modeled users, with the user-item rating composing one tuple. We introduce two formal definitions in these examples: first, the set of classes $C$ and a single class $c$, and second is the rating (or score, weight, times played) represented as $w_{ui}$.

The **processing** step is composed only of the **recommender algorithm** component, which is responsible for executing the recommender algorithm and producing the candidate items. In this second step, any technique can be implemented as collaborative filtering, content-based filtering, hybrid technique, and its recommender algorithms. Most of the related works implement collaborative filtering. Thus, we implement this technique inspired by the state-of-the-art [2, 6, 8, 10]. In some of these works, the recommenders provide a rank prediction [6, 10], and others provide rating predictions [8]. Regardless of the recommender algorithm implementation, the output of this step is the user’s candidate items. Table 4 provides an example of three users and their candidate items. The formal representation of the predicted rating is $\hat{w}_{ui,j}$. For each unknown item by the user, the recommender algorithm predicts a possible weight (e.g., rating) that the user will attribute to the item.

**Post-processing** is the most critical step for the calibrated recommendation system. It receives a set of candidate items from the processing step and uses them to create a final
Introducing a framework and a decision protocol... 22047

recommending a framework and a decision protocol... 22047

recommendation list. This step aims to select a subset of items from the user’s candidate items that are relevant and fair to the user. To achieve that, many formulas are used to compose the workflow. Thus, the state-of-the-art [2, 6, 8, 10, 11] implements different systems. In this section, we investigate the basic component and some generalized equations.

The eight core components of the post-processing are described as follows: i) filtering, ii) modeling, iii) distributions, iv) calibration measure (aka divergence measure), v) relevance measure, vi) trade-off weight (\( \lambda \)), vii) trade-off balance and viii) select item algorithm. On the post-processing, the filtering component will filter the inputted candidate items. The state-of-the-art [6, 8, 10] uses this component to get the 100 most relevant candidate items or the 1000 top candidate items [12]. Sometimes, it is impossible to use all candidate items due to the time complexity, which is an NP-hard problem [6, 8]. Thus, inspired by the state-of-the-art, we filter the 100 top relevant items in the candidate items set, i.e., the 100 items with high value in \( \hat{w}_{u,i} \). The modeling component is responsible for modeling the candidate items as a usable entrance to the post-processing. A predicted weight \( \hat{w}_{u,i} \) is needed in each candidate item. Thus the component prepares the items with the requested information and structure. All other components will be presented in sequence. Figure 1 shows the conceptual framework with its steps and components described so far.

2.2 Notations

The mathematical notations used in the following sections are formally presented in Table 5. Along this section, all notations are contextualized.

2.3 Class distributions

As introduced, the calibration recommendation system aims to create a list of recommendations that tracks the distribution of user’s preferences, i.e., the system extracts from the user’s preferences items a target distribution that the recommendation list must achieve. Distributions are obtained from the item classes. Extracting is done in different ways by the state-of-the-art, in some cases implementing genres [6, 8], popularity [13], or sub-profiles [10].

2.3.1 Class approach

Inspired by the state-of-the-art [6, 8], we consider the genre probability of being chosen in an item as part of the distribution equation. \( p(g|i) \) receives a genre \( g \) and an item \( i \) and verifies if the genre composes the item. If the genre belongs to the item, the equation returns a value computed as [6, 8]:

\[
p(g|i) = \frac{1}{|\text{classes}\ln(i)|}.
\]

2.3.2 Target distribution

It is based on the user’s preferences \( I(u) \). \( p \) is the formal representation of the Target Distribution [14] and it is described as [6, 8]:

\[
p(g|u) = \frac{\sum_{i \in I(u)} 1(g \in i) w_{u,i} \cdot p(g|i)}{\sum_{i \in I(u)} 1(g \in i) w_{u,i}},
\]

Table 5 Formal representation of the notations used in this study

| Notation | Description |
|----------|-------------|
| \( U \)  | Set of all users |
| \( u \)  | A user |
| \( I \)  | Set of all items |
| \( i \)  | An item |
| \( G \)  | Set of all genres |
| \( g \)  | A genre |
| \( I(u) \) | Set of items in the user preference \( u \) |
| \( C I(u) \) | Set of candidate items to be recommended to user \( u \) |
| \( L \)  | A generic list with ordered items |
| \( W(i) \) | All feedback weight values over an item \( i \) |
| \( w_{u,i} \) | Feedback weight value given by \( u \) over \( i \) |
| \( \hat{w}_{u,i} \) | Predicted feedback weight value for \( u \) over \( i \) |
| \( N \)  | Length of the final recommended list |
| \( R^* \) | Final recommended list to a user |

Fig. 1 The conceptual framework with its steps and components
where \( w_{u,i} \) is the item relevance weight value (aka rating) from the user \( u \) over an item \( i \); \( p(g|i) \) is the genre probability value presented in (1); \( 1 (g \in i) \) indicates that if the genre \( g \) is present in an item \( i \), then the result returns 1, if it is not, the result returns 0.

### 2.3.3 Realized distribution

It represents the distribution from the recommendation list \( L \), which the system aims to produce in less divergence with \( p, q \) is the formal representation of the Realized Distribution \([14]\) as follows \([6, 8]\):

\[
q(g|u) = \frac{\sum_{i \in \mathcal{L}(u)} 1 (g \in i) \hat{w}_{u,i} \cdot p(g|i)}{\sum_{i \in \mathcal{C}(u)} 1 (g \in i) \hat{w}_{u,i}},
\]

where \( q \) is similar to \( p \). The difference between them is the weighing strategy \( \hat{w}_{u,i} \), which in our approach is the predicted relevance weight from the recommender algorithm.

Some works implement \( \tilde{q} \) instead of \( q \) to deal with values and it is presented as \([6, 8]\):

\[
\tilde{q}(g|u) = (1 - \alpha) \cdot q(g|u) + \alpha \cdot p(g|u).
\]

### 2.4 Calibration measure

Likewise \([8]\), to measure how much calibrated \( p \) and \( q \) are, we use three divergence calibration measures: 1) Kullback–Leibler (\( \text{FKL} \)), 2) Hellinger (\( \text{FHE} \)), and 3) Pearson Chi-Square (\( \text{FCHI} \)). All measures work in an interval between \([0, \infty) \) \( \in \mathbb{R} \). High values indicate miscalibration between \( p \) and \( q \), and low values indicate calibration. In some works \([12, 15]\), the divergence measure is called the calibration measure.

#### 2.4.1 Kullback–Leibler

The KL-divergence equation is presented as follows \([6, 8, 10, 16]\):

\[
\text{FKL}(p, q) = \sum_{g \in \mathcal{G}} p(g|u) \log_2 \frac{p(g|u)}{q(g|u)}.
\]

#### 2.4.2 Hellinger

Steck \([6]\) alerts that Hellinger can be used as a calibration measure. Consequently, \([8]\) implement it in their work as described below \([8, 16]\):

\[
\text{FHE}(p, q) = \sqrt{2 \cdot \sum_{g \in \mathcal{G}} \sqrt{p(g|u)} - \sqrt{q(g|u)}}^2.
\]

#### 2.4.3 Pearson Chi Square

da Silva et al. \([8]\) propose to use the Pearson Chi-Square. We applied that measure too as follows \([8, 16]\):

\[
\text{FCHI}(p, q) = \sum_{g \in \mathcal{G}} \frac{(p(g|u) - \bar{q}(g|u))^2}{\bar{q}(g|u)}.
\]

### 2.5 Relevance rank measure

We use an equation to obtain a relevance rank weight, inspired by the state-of-the-art as follows \([6, 8, 10]\):

\[
\text{Sim}(L) = \sum_{i \in \mathcal{L}(u)} w_{r(a,i)}.
\]

where the \( \text{Sim}(L) \) is the sum of the predicted relevance weight given by the recommender algorithm.

### 2.6 Personalized trade-off weight

Most related works \([6, 10, 15, 17]\) use constant trade-off weights \( \lambda \). \([8]\) propose two personalized ways to find the trade-off weight. As proposed by our conceptual framework, the trade-off weight is a component of the post-processing step, and such a component acts as a parameter optimization. The state-of-the-art shows that the \( \lambda \) produces changes on the recommendation list, affecting the system performance.

#### 2.6.1 Normalized Variance

The equation calculates the distance along all values in the distribution, each of that value is a genre/class representation, and the values can be between \([0, 1] \) \( \in \mathbb{R} \). It is described as \([8]\):

\[
\text{mp}(u) = \frac{\sum_{g \in \mathcal{G}} p(g|u)}{|\mathcal{G}|},
\]

where the result from \( \text{mp}(u) \) is an average of the values in each genre. That value represents a center point among all genres.

\[
\lambda_u = 1 - \frac{\sum_{g \in \mathcal{G}} |p(g|u) - \text{mp}(u)|^2}{|\mathcal{G}|},
\]

where \( \lambda_u \) is an average of the summation over all genres, computing the squared difference between a genre and the center point.
2.6.2 Genre count

A second equation to find the personalized trade-off weight is described [8]:

\[
\lambda_u = \frac{\sum_{g \in G} \mathbb{1}(g \in I(u))}{|G|},
\]

(11)

where \( \mathbb{1}(g \in I(u)) \) returns 1 if the genre \( g \) is in an item of the user preference, otherwise 0. After the summation, the average is computed, and that final value indicates the degree of calibration/fairness with the user’s preferences.

2.7 Trade-off balance

All equations presented so far are to compose the trade-off balance. Then, we introduced ways to achieve the relevant and fair portion of the trade-off balance, which brings all parties together to generate a fair list of recommendations. Finally, in our study, we implement [6] claims used by [8, 10] and propose a new trade-off balance.

The Maximum Marginal Relevance (MMR) [18] is commonly used to maximize the choice of items in the recommendation. It tends to get the optimal set of recommendations

\[
\text{arg max } \text{MMR}(L, p),
\]

(12)

where the \( \text{MMR}(\lambda_u, L, p) \) is any trade-off balance. \( L \) is a sub list from the candidate items. That is an intermediate list used when building the final recommendation list \( R^* \).

2.7.1 Linear

Steck [6] proposes a way to calibrate the recommendation list with a trade-off between relevance and calibration. In our study, we call it a Linear trade-off, and it is described as [6, 8]:

\[
\text{TradeLin}(\lambda_u, L, p) = (1 - \lambda_u) \cdot \text{Sim}(L) - \lambda_u \cdot F(p, q(L)),
\]

(13)

where: \( \lambda_u \) is any personalized trade-off weight from Sect. 2.6 (or constant weight to all users), Sim(L) is from Sect. 2.5 and \( F(p, q(L)) \) is any divergence/calibration measure from Sect. 2.4.

2.7.2 Logarithmic bias

We claim a new trade-off balance to calibrate the recommendation list. Our balance considers the users’ bias \( (\hat{b}_u) \) as a way to improve the calibration with the user’s preferences, leading to a more personalized distribution. It is presented as:

\[
\text{TradeLog}(\lambda_u, L, p) = \text{sign}(\text{TradeLin}) \cdot \log^{\text{abs}(\text{TradeLin}+1)} + \hat{b}_u(L),
\]

(14)

where \( \text{sign}(\text{TradeLin}) \) is a function that returns the signal from the value obtained in the \( \text{TradeLin} \) (13). The \( \log^{\text{abs}(\text{TradeLin}+1)} \) is a logarithm from the \( \text{TradeLin} \) absolute value (i.e., casting to positive) due to the log do not comprehend negative values, and the +1 is to avoid the log from returning zero. The proposal of log usage is based on the distribution function that considers any item relevance value from \( w_{u,i} \). These values are used by the calibration measures, providing high values of divergence. Thus, the log is used to consider its scenarios. The \( \hat{b}_u(L) \) is presented in sequence.

Koren and Bell [19] present user and item bias equations. The item bias is described as follows:

\[
b_i = \frac{\sum_{u \in W(i)}(w_{u,i} - \mu)}{\sigma + |W(i)|}.
\]

(15)

Our proposed users’ bias is based on the items with the predicted relevance weight \( w_{r(u,i)} \) in the place of relevance weight \( w_{u,i} \). This adaptation turns possible to apply the users’ bias in the post-processing step. Our claim on the user’s bias equation indicates how much the rating prediction by the recommender algorithm is close to the real ratings. The \( \mu \) is an average of the relevance weights among all users’ preferences. \( \alpha \) and \( \sigma \) values are to avoid division by zero and we adopt 0.01 to both. The \( \hat{b}_u(L) \) receives the list \( L \) and finds the users bias for that list. The formal representation of the users’ bias with the proposed adaptation is:

\[
\hat{b}_u(L) = \frac{\sum_{i \in L}(w_{r(u,i)} - \mu - b_i)}{\sigma + |L|}.
\]

(16)

2.8 Select items algorithm

The last component is the select items algorithm. Here many re-ranking or selection algorithms can be applied. Most related works implement the Surrogate Submodular [20]. Similar to [6, 8, 10], we also implement it. The Submodular algorithm workflow is:

- \( R^* \) starts with an empty list {};

\[ \text{Sim}(L) \]

\[ F(p, q(L)) \]
• for each $N - 1$ interaction with the candidate items, if an item $i$ tested as optimal in temporally list $L$ it is the one that maximizes $R^*$, then $i$ is appended into the list;
• in the end, the list has $N - 1$ items, and the last optimal item is appended, completing the top-n items $R^*$.

3 Decision protocol

The main contribution of our work is a protocol for deciding which system is the best to be implemented driven by the data domain. The decision protocol expects two or more calibrated systems as input and suggests the best one based on the coefficient results. These systems are combinations of calibrated ones based on the proposed framework.

3.1 Metrics

The protocol uses three well-known metrics: The Mean Average Precision (MAP), a traditional metric used by the Recommendation Systems community; The Mean Average Calibration Error (MACE); and the Mean Rank MisCalibration (MRMC). MACE and MRMC are proposed by [8]. Therefore, each calibrated system needs to be evaluated by these metrics.

The MAP is defined as the equation below, where the $Ave P(u)$ represents the average precision of the recommendation list to user $u$ and is done to all users $U$. $p@i$ represents the number of successful items until the position $i$.

\[
MAP = \frac{1}{|U|} \sum_u AveP(u),
\]

\[
AveP(u) = \frac{1}{N} \sum_{i=1}^{N} p@i,
\]

\[
p@i = \frac{r}{i}.
\]

The MACE is defined by [8] as the equations below:

\[
CE(u, p, q) = \frac{\sum_{g \in G} |p(\text{glu}) - q(\text{glu})|}{|G|},
\]

where it shows the calibration error to a user.

\[
ACE(u) = \sum_{j=1}^{N} CE(u, p, q(R^* @ j))
\]

where $N$ is the recommendation list size. The equation computes the average calibration error of a recommendation list, considering each rank position.

\[
MACE = \frac{\sum_{u \in U} ACE(u)}{|U|},
\]

where it gets the average over the users.

The MRMC is defined by [8] as:

\[
MC(p, q) = \frac{F(p, q)}{F(p, q(\{\}))},
\]

where the equation computes the normalized miscalibration.

\[
RM C(u) = \frac{\sum_{j=1}^{N} MC(p, q(R^* @ j))}{N},
\]

where it computes the rank miscalibration, considering each rank position and the insertion of each item on the list.

\[
MRMC = \frac{\sum_{u \in U} RM C(u)}{|U|},
\]

it compiles the rank miscalibration of all users in a single value.

3.2 Coefficients

Steck [6] affirms that calibration reduces precision. da Silva et al. [8] show that calibration increases precision. Thus, considering these findings, calibration can increase or decrease precision performance depending on the dataset and system implemented. We develop the coefficients as a tool to help the system specialist. Each coefficient comprises the degree of precision and calibration in a single value. This value indicates if the results can be acceptable to the specialist, if the system needs some intervention to provide more calibration or precision, or if it is possible to lose some precision to gain calibration. We propose two coefficients: i) the Coefficient of Calibration Error (CCE) and ii) the Coefficient of MisCalibration (CMC). Both coefficients are part of our approach and determine the protocol’s effectiveness.

The Coefficient of Calibration Error (CCE) can be described as follows:

\[
CCE = \frac{MACE}{MAP}.
\]

The Coefficient of MisCalibration (CMC) can be described as:

\[
CMC = \frac{MRMC}{MAP}.
\]

The proposal is a division between the MACE or MRMC [8] by the MAP. All metrics evaluate the results of each position in the recommended list. In (26) and (27), the numerator is the MACE or MRMC results average, and the denominator is the MAP results average. All results are obtained over the
same calibrated system. It is worth mentioning that when the MACE or MRMC is higher, and the MAP is lower, the obtained CCE or CMC value is higher. That means the system has more error or miscalibration than precision.

Conversely, when the MACE or MRMC is lower, and the MAP is higher, the obtained CCE or CMC value is lower, meaning the system has fewer errors (or miscalibration) and is more precise. The two coefficients indicate systems that want to balance precision and calibration simultaneously, considering losing some precision to obtain more calibration or vice-and-versa. Although if the system provides calibration and precision, the coefficients will show it. In (26) and (27), the variables $MACE$, $MRMC$, and $MAP$ are representations of each metric average computed over each system running. Figure 2 shows the four possible results the coefficients can indicate. (A) High calibration error and low precision, (B) low error and high precision (ideal case), (C) low error and low precision, and (D) high error and high precision.

### 3.3 The decision

From the coefficients, the decision protocol indicates which system combination is the best choice. At this point, we have two coefficient values for each calibrated system. Using the coefficients to decide which system will be selected allows considering the precision and calibration through the rank perspective, giving an attempt to the system that presents more positive and calibrated items on the top of the list. The main objective of the protocol is to simplify the process of deciding the best-calibrated recommender systems. When a set of calibrated recommendation systems are implemented, the context needs precision and calibration simultaneously. Therefore, the protocol uses the coefficients in addition:

$$s_i = CCE + CMC,$$

where the system $i$ has its performance ($s_i$) measured with the coefficients, representing its total error and miscalibration over the precision as previously debated.

The final decision is obtained by:

$$S = \min(s_1, s_2, s_3... s_n),$$

where the $n$ calibrated systems are judged, the system with the lower value of $s$ is chosen as the best-calibrated system $S$.

To explain the decision protocol functionality, we introduce the example in Table 6. As introduced, the CCE considers the absolute error, and the CMC considers the miscalibration over the list. The protocol helps the system specialists, showing them the system with the lowest degree of calibration error, miscalibration and the highest degree of precision. In the example, the system $s_4$ has the lowest CCE, while the system $s_3$ has the lowest CMC. The system specialist can consider both systems because of their high precision and lower calibration error and miscalibration. However, the system specialist needs to choose one system. The decision protocol indicates that the $s_4$ can be considered by the system specialist to be implemented. When the two coefficients are considered together, the system $s_3$ is penalized because it has much more absolute error.

In the following sections, we will describe the experimental setup and results. Our research implements more than one thousand calibrated systems (considering the parameter optimization) and suggests the best implementation based on the proposed decision protocol.

### 4 Experimental setup

In this section, we illustrate the experimental setup applied in our study. We describe the used datasets and methodology. Our setup is modeled to provide a better understanding of the system’s behavior. The datasets and methodology are inspired by related works in the state-of-the-art [6, 8, 10].
Table 7: Datasets before and after the pre-processing

| Datasets           | |U|   | |I|   | |W|   | |G|   |
|--------------------|---|---|---|---|---|---|---|---|
| Raw ML20M          | 138,493 | 27,278 | 20,000,263 | 19 |
| Used ML20M         | 80,672 | 15,400 | 9,013,655 | 19 |
| Raw Taste Profile  | 1,019,318 | 999,056 | 48,373,586 | 16 |
| Used Taste Profile | 94,611 | 98,305 | 4,807,615 | 16 |

4.1 Datasets

Inspired by [10], we use two public datasets for evaluating the proposal. Table 7 shows a data description from both datasets.

For the movie domain, we use the MovieLens 20M dataset (ML20M)\(^1\) [21], which is the same dataset used by [6, 8, 10]. Table 7 shows the numbers of users (|U|), items (|I|), feedback (|W|), and genres (|G|). As defined in the proposed framework, the pre-processing applies cleaning, filtering, and modeling to the data. Based on that and inspired by [6, 8, 10], we apply a rating cut by four and remove all lower ratings, removing movies without genre information and movies without user interactions. However, unlike [6, 8, 10], we dropped users with a preference set size smaller than 30 items. In addition, we dropped items with less than three interactions with users to avoid the item cold-start.

For the song domain, we use the Taste Profile\(^2\) dataset and the Tagtraum genre annotations\(^3\), which is the same dataset used by [10], though it is not the same genre annotation. Table 7 shows the numbers of users, items, feedback/transactions, and genres. For obtaining pre-processed data as indicated in the framework and similarly to [10], we applied: the elimination of songs without genre information. Opposite to [10], we dropped users with a preference set size smaller than 30 items, whose value was defined based on the size of the recommendation list, and a cut by the most played songs where it discards the ones played less than three times. In addition, we dropped items with less than three interactions with users to avoid the item cold-start.

4.2 Methodology

With the pre-processed data, we ran the Grid Search method. Each recommender algorithm used in our experimentation searches for the best hyperparameters with a 3-fold cross-validation methodology. To choose the best hyperparameters (HP), we use the Mean Average Error (MAE) as a decision metric, i.e., the HP combination with the smallest error is chosen. The MAE measures the absolute error between the real and predicted weight. Thus the smaller the MAE error, the smaller the error propagation for post-processing. The choice of MAE is directly related to the predicted user feedback $\hat{w}_{ui}$. Computing the distribution values in (3) requires a low error value on the prediction algorithm. Otherwise, the error will propagate from the processing to the post-processing step. Thus, the absolute difference computed by the MAE reduces the error propagation and allows the calibrated recommendation system to balance relevance and calibration during the post-processing. Steck [6] indicates that other values can be used in place of the predicted user’s feedback $\hat{w}_{ui}$, as the rank of the items. However, we follow the author and use the relevance value.

We select seven well-known recommender algorithms to understand the post-processing effects better and test our decision protocol and framework, as shown in the processing step. They are:

1. Basic UserKNN: User-based K Nearest Neighbors [22] is a recommender algorithm that finds users with a high degree of similarity feedback with $u$ and creates a recommendation list with top similar items in feedback based on the $u$ neighbors. It is a memory-based recommender. The best HP are: the number of neighbors $k = 30$ and the Mean Squared Difference similarity measure for both datasets;

2. Basic ItemKNN: Item-based K Nearest Neighbors [22] is a recommender algorithm that finds items with high similarity feedback with the items in $u$ preferences and creates a recommendation list based on the top similar feedback items. It is a memory-based recommender. The best HP are: the number of neighbors $k = 30$ and Person Correlation similarity measure for both datasets;

3. Slope One: [23] presents the algorithm with no HP. The recommender considers the deviation in the users’ feedback between each pair of items $u$ has and uses it to predict the rating of the users based on $u$ mean rating. It is a memory-based recommender;

4. NMF: Non-negative Matrix Factorization presented by [24]. The recommender does not accept negative values in the matrix, considering that the latent factors in the user’s feedback cannot be negative. It is a model-based recommender. The best HP are: i) for ML20M, the number of epochs $ne = 50$, number of latent factors $f = 50$, user and item learning rate $\gamma_u = 0.005$ and $\gamma_i = 0.005$, user and item regularization constant $\lambda_u = 0.005$ and $\lambda_i = 0.05$, and user and item regularization bias $\lambda_{bu} = 0.005$ and $\lambda_{bi} = 0.005$; ii) for Taste Profile the number of epochs $ne = 30$, number of latent factors $f = 50$, user and item learning rate $\gamma_u = 0.003$ and $\gamma_i = 0.005$, user and item regularization constant

---

\(^1\) https://www.kaggle.com/grouplens/movielens-20m-dataset.
\(^2\) http://millionsongdataset.com/tasteprofile/
\(^3\) http://www.tagtraum.com/genres/msd_tagtraum_cd2.cls.zip.
Introducing a framework and a decision protocol... 22053

\[ \lambda_u = 0.003 \text{ and } \lambda_i = 0.05, \text{ and user and item regularization bias } \lambda_{bu} = 0.003 \text{ and } \lambda_{bi} = 0.005; \]

5. **SVD**: Singular Value Decomposition proposed by [25] is a recommender algorithm based in matrix factorization. It is commonly used due to its positive performance in time and precision. The recommender was popularized during the Netflix Prize. It is a model-based recommender. The best HP are: i) for ML20M, the number of epochs \( ne = 50 \), number of latent factors \( f = 50 \), user and item learning rate \( \gamma_u = 0.005 \) and \( \gamma_i = 0.005 \), and user and item regularization \( \lambda_u = 0.01 \) and \( \lambda_i = 0.01 \); ii) for Taste Profile the number of epochs \( ne = 30 \), number of latent factors \( f = 150 \), user and item learning rate \( \gamma_u = 0.001 \) and \( \gamma_i = 0.001 \), and user and item regularization \( \lambda_u = 0.05 \) and \( \lambda_i = 0.05 \);

6. **SVD++**: An extension of SVD presented by [26] that considers the implicit feedback during its process. However, the recommender takes more time to finish its recommendation. It is a model-based recommender. The best HP are: for ML20M and Taste Profile, the number of epochs \( ne = 50 \), number of latent factors \( f = 50 \), user and item learning rate \( \gamma_u = 0.005 \) and \( \gamma_i = 0.005 \), and user and item regularization constant \( \lambda_u = 0.01 \) and \( \lambda_i = 0.01 \);

7. **Co Clustering**: Presented by [27], Co Clustering is a recommender based on clusterization. First, it considers users and items clusters separated and, after, creates co-clusters for users and items to produce the recommendation list. It is a model-based recommender. The best HP is: i) for ML20M the number of epochs \( ne = 50 \), number of user and item clusters \( C_u = 3 \) and \( C_i = 7 \); ii) for Taste Profile the number of epochs \( ne = 10 \), number of user and item clusters \( C_u = 3 \) and \( C_i = 3 \).

This study does not propose any change in the recommender algorithms workflow. Therefore, we use the Python Surprise library [28] implementation to perform our experiments. The selected collaborative filtering recommenders used in the experiment are presented in the literature by [2, 7, 8].

The datasets used in the experiments are randomly divided for each user into 70% of training data and 30% of test data. This validation methodology is used by the state-of-the-art [6, 8, 10]. All test data is mixed with all unknown items by the user, and the test items are expected to compose the final recommendation list. The training data are input to the recommender to learn about user preferences. Having the recommender trained, the algorithm predicts the rating for all unknown+test items for each user. After predicting all possible ratings to the users-item matrix, similarly to [6, 8, 10], we get the top 100 candidate items with the highest rating predicted by the recommender algorithm. The post-processing for each user starts with the top-100 candidate items, from which the system will select the final recommendation list.

In addition to the two personalized trade-off weights, we use constant values of \( \lambda \). These same weighing values are also used by [8, 10] and are \( \lambda \in \{0.0; 0.1; \ldots; 0.9; 1.0\} \).

All system combinations are generated considering two datasets, seven recommender algorithms, two trade-off balances, two personalized weights (VAR, CGR), eleven constant values (totaling 13 trade-off weights), and the three divergence measures. Each configuration combines a different way to generate the final recommendation list, so we have \( 2 \cdot 7 \cdot 2 \cdot 13 \cdot 3 = 1092 \) different combinations to be evaluated by each metric. Finally, we run each combination three times with different training and test data.

The final recommendation list \( R^* \) is generated with the top-10 fair items given by the post-processing. Finally, we evaluate the recommendation list in all \( \{1:10\} \in \mathbb{N} \) positions.

5 **Experimental results**

This section presents the experimental results based on the setup described previously. First, we initiate describing the metrics MAP (Sect. 5.1), MACE (Sect. 5.2), and MRM (Sect. 5.3) for both datasets. Then, in sequence, we present the cross-metric between MAP and MACE called CCE (Sect. 5.4), the crossed results between MAP and MRM called CMC (Sect. 5.5), and the protocol decision (Sect. 5.6). Next, Sect. 5.7 answers all the research questions presented in the paper introduction. Finally, Sect. 5.8 discusses the findings.

5.1 **MAP**

We start our analyses using MAP metric. Figure 3 presents the results using the Movielens, and Fig. 4 presents the results using the Taste Profile.

5.1.1 **Movielens**

Figure 3 shows the MAP results for the Movielens dataset. It is possible to observe that the recommender algorithms with the best precision are SVD++, NMF, and SVD. They were followed by Slope One, Co-Clustering, and User-KNN. The worst results are with the Item-KNN. For SVD++, SVD, and User-KNN, the Logarithmic trade-off achieve the best results. The Linear trade-off achieves the best results for NMF, Slope One, Clustering, and Item-KNN.

For SVD++ and SVD, the best combination is the Logarithmic trade-off and the Hellinger \( (LOG_HE) \). For the NMF, the Linear trade-off gets the best results. Analyzing only by MAP, the system specialist can consider the HELOG-SVD++. The weight values of 0.7 or 0.8 can be chosen to tune the system. These weights produce 0.032 against
the pure recommender recommendation list, achieving 0.020 precision. We can also observe that CHI-LOG-SVD++ achieves moderated precision results compared to the best and worst ones.

The results in Fig. 3 demonstrate that the calibration can produce positive effects on precision depending on the recommender algorithm. The results show that all recommenders can increase the precision. There are cases in which the precision is reduced. However, the best result is produced by the calibration.

5.1.2 Taste profile

Figure 4 shows the MAP results for the Taste Profile dataset. It is possible to observe the best precision performance with the recommender algorithms Item-KNN, SVD++, and SVD.
Introducing a framework and a decision protocol...

Fig. 4  Taste Profile - MAP - Results for all recommender algorithms

followed by NMF, User-KNN, and Slope One. The worst result is with the Co-Clustering. For most results, the Linear trade-off achieves the best results.

For Item-KNN, the best combination is the Linear trade-off and the Hellinger \( (LIN\_HE) \). For SVD are \( LIN\_HE \) and \( LIN\_CHI \). For the SVD++, the best results were achieved by the combination \( LIN\_KL \). The \( LIN\_CHI \) achieves the best result for some recommenders, such as User-KNN, Slope One, NMF, and Co-Clustering. Analyzing only by MAP, the system specialist can consider the HE-LIN-Item-KNN. To tune the system, weight values of 0.1 or 0.2 can be chosen.

The results in Fig. 4 demonstrate that the calibration can produce positive effects on precision depending on the recommender algorithm. For all recommenders, most combinations increase the precision. That reinforces the findings that calibration can improve precision.
5.2 MACE

In sequence, we evaluate the MACE individually. Figure 5 presents the results using the Movielens, and Fig. 6 presents the results using the Taste Profile.

5.2.1 Movielens

Figure 5 shows the MACE results for the Movielens dataset. The lower the value, the better the performance. The best recommender algorithms are SVD++, SVD, and Co Clustering.

Fig. 5 Movielens - MACE - Results for all recommender algorithms
Introducing a framework and a decision protocol... 22057

Fig. 6  Taste Profile - MACE - Results for all recommender algorithms

Item-KNN, NMF, and User-KNN followed them. The worst results are with Slope One. For SVD++, SVD, and Item-KNN, the Logarithmic trade-off with the Hellinger achieves the best results, followed by the Linear trade-off with the Hellinger. Which achieves the best results for all other algorithms, indicating that the calibration measure improves the algorithm performance and the Hellinger is a measure to reach it. Also, the figure shows that the LIN-KL achieves all the worst results. The Pearson Chi-Square measure achieves medium results. Analyzing only by MACE, the system specialist can consider the HE-LOG-SVD++ because it has the best MACE performance, using the trade-off weights 0.2 and 0.3. The specialist can also consider the SVD because it has similar results to the SVD++. 123
5.2.2 Taste profile

Figure 6 shows the MACE algorithms for the Taste Profile dataset. The best recommender algorithms are Item-KNN, NMF, and SVD. Followed by SVD++ and Co-Clustering. The worst results are with the User-KNN and the Slope One. For most results, the Linear trade-off achieves the best performance. Except for some cases as the HE-LOG-Item-KNN for instance.

The best combination for Item-KNN, SVD, SVD++, and NMF is the Linear trade-off and the Hellinger ($LIN_{HE}$), followed by the Pearson Chi-Square. Analyzing only by MACE, the system specialist can consider the HE-LIN-Item-KNN. The weight values of 0.1 to 0.7 can be chosen to tune the system. All recommenders are influenced by the trade-off weight 1.0, which can cause a miscalibration reduction or increase. However, it is observable that the influence can reduce half of the miscalibration.

5.3 MRMC

In the last individual analysis, we evaluate the MRMC. Figure 7 presents the results using the Movielens, and Fig. 8 presents the results using the Taste Profile.

5.3.1 Movielens

Figure 7 shows the MRMC results for the Movielens dataset. Similar to MACE, the lower the value, the better the performance. It is possible to observe that all recommender algorithms have similar results behavior. The combinations LOG-CHI and LIN-CHI achieve the best performance. In sequence, the combinations LIN-KL and LOG-CHI for all recommenders achieve all the second and third-best results. Differently from MAP and MACE, the Hellinger combinations achieve the worst results. We can affirm that the calibration measure influences the miscalibration control.

Observing the recommenders, we can see that to MRMC, the best recommenders are Slope One and User-KNN, followed by the SVD and SVD++. Analyzing only by MRMC, the system specialist can consider the CHI-LOG-SLOPE or CHI-LIN-SLOPE.

5.3.2 Taste profile

Figure 8 shows the MRMC results for the Taste Profile dataset. It is possible to observe common behaviors among the recommender algorithms results. The Co Clustering achieves the best calibration at weight 1.0 by the combination LIN-CHI, followed in sequence by LOG-CHI and LIN-KL. To all recommenders, the LIN-CHI and LIN-KL achieve the best results. The SVD and SVD++ show similar results, presenting a stable miscalibration reduction.

When we observe the MRMC for the Taste Profile, we see that the weight value 1.0 influences all recommenders, reducing the miscalibration. Analyzing only by MRMC, the system specialist can consider the Co-Clustering with the weight in 1.0. However, the SVD++ and SVD bring three times miscalibration reduction to all trade-off weights. That would lead the specialist to decide on a stalemate.

5.4 MAP x MACE

First, we analyze the results obtained from the MAP and MACE metrics. To do that, we combine the results of each system combination, where each dot in the lines is a $\lambda$. Figure 9 presents the cross-metric MAPxMACE for Movielens, and Table 8 presents the CCE values. Figure 10 presents the cross-metric for the Taste Profile, whereas Table 9 presents the CCE values.

5.4.1 Movielens

We start the dataset analysis with the system combinations that implement the KL-divergence. Figure 9a and b and Table 8 (Lines 1 and 2) show that the best trade-off balance is the $LOG$. KL-LOG-SVD++ combination achieves the best performance with $CCE = 3.05 (LOG)$. The second and third places are the combinations that use matrix factorization KL-LIN-SVD++ $CCE = 3.52$ and KL-LIN-NMF $CCE = 3.8$. To the $LIN$ trade-off (Fig. 9a), the recommenders change the positions to NMF and SVD sequentially. As indicated by Table 8 (Lines 1 and 2), the best performances in the KL-divergence are all matrix factorization approaches, especially to the SVD++ with $CCE = 3.05 (LOG)$ and $CCE = 3.52 (LIN)$. The worst results are held by the KNNs approaches considering the KL-divergence as a fixed divergence measure.

The results achieved by the Hellinger combinations are shown in Fig. 9c, in d and Table 8 (Lines 3 and 4). The results indicate that the $LOG$ trade-off (Fig. 9d), similar to the KL analysis, achieves the best performance, and such behavior remains in all matrix factorization approaches. It is possible to verify that the worst performance remains with the KNNs approaches. The Hellinger achieves lower MACE values than the KL-divergence, indicating that the HE produces a recommended list with low calibration error. The results in Table 8 (Lines 3 and 4) show that the SVD++, in the Hellinger, achieved the $CCE = 2.08$ to $LOG$ and $CCE = 2.38$ to $LIN$.

The last MAPxMACE results we analyze are the Pearson Chi Square ($\chi^2$) combinations. Figure 9e and f with Table 8 (Lines 5 and 6) show that the $\chi^2$ follows the same behavior found in the two previous measures analyzed. The best performances are all matrix factorization approaches, as the KNNs obtain the worst performance. The CHI-LIN-
Introducing a framework and a decision protocol...

Fig. 7 Movielens - MRMC - Results for all recommender algorithms

NMF obtains the best combination performance in the $\chi^2$ with a $CCE = 2.92$ (Table 8 Lines 5 and 6), followed by the SVD++ in second place and the SVD in third place. In the LOG trade-off, the places follow the sequence SVD++, SVD, and NMF. In the LIN trade-off, the places follow the sequence SVD++, NMF, and SVD.

The results indicate that if the system is focused on MAP and MACE and according to our experimental setup, the best combination to be implemented in the Movielens dataset is the HE-LOG-SVD++ with a $CCE = 2.08$, as presented in Table 8 Line 4. The second best performance based on the CCE belongs to the HE-LIN-SVD++ with $CCE = 2.38$, and
the third place belongs to HE-LIN-NMF with \( CCE = 2.51 \) (Table 8 Line 3). We also observe that the Hellinger divergence measure achieves all the best performances on the CCE and there is no correlation between the MAP and MACE. The CCE compiles the results of the individual metric observation from Figs. 3 and 5, which shows that the HE-LOG-SVD++ achieves the best performance.

5.4.2 Taste profile

We start our results analysis in the Taste Profile by the KL-divergence. Figure 10a and b and Table 9 (Lines 1 and 2) show that the best recommender performance in the KL-divergence belongs to the Item-KNN in LIN and LOG. Verifying that the Item-KNN results in the LIN have more
variability than in the LOG trade-off. The CCE obtained by the combination KL-LIN-Item-KNN is 25.73 against 26.68 from KL-LOG-Item-KNN (Table 9 Lines 1 and 2). The second and third places are the SVD++ and SVD, both matrix factorization approaches. The worst performance belongs to Co Clustering.

Figure 10c and d support our analysis of the divergence measure Hellinger in the Taste Profile, besides Table 9, in Lines 3 and 4, presents the CCE performances. Like KL-
Table 8: Movielens - CCE - Results from LIN and LOG trade-offs combined with the divergence measures KL, HE, and CHI for each recommender algorithm.

| Divergence | Trade-off | Co Clustering | Item KNN | NMF | Slope One | SVD | SVD++ | User KNN |
|------------|-----------|---------------|----------|-----|-----------|-----|-------|----------|
| KL         | LIN       | 60.44         | 90.73    | 3.8 | 63.52     | 4.95| 3.52  | 430.91   |
| KL         | LOG       | 62.94         | 110.46   | 5.76| 83.46     | 4.31| 3.05  | 130.58   |
| HE         | LIN       | 34.7          | 57.4     | 2.51| 40.28     | 3.68| 2.38  | 711.24   |
| HE         | LOG       | 57.84         | 157.32   | 8.31| 104.41    | 3.06| 2.08  | 60.73    |
| CHI        | LIN       | 38.26         | 63.33    | 4.83| 71.43     | 4.3 | 3.06  | 364.44   |
| CHI        | LOG       | 50.63         | 88.66    | 2.92| 47.41     | 4.29| 3.06  | 364.44   |

From the CCE results in the Taste Profile, we can observe that if the system is focused on MAP and MACE, the best performance refers to the combination HE-LIN-Item-KNN with a CCE = 14.39, followed by the combination HE-LOG-Item-KNN with a CCE = 21.32, and CHI-LIN-Item-KNN with CCE = 23.29. These results can be verified in Table 9 in lines 3 and 4. We remind that results are restricted to the CCE (MAP and MACE metric crossed). The CCE compiles, in its results, for Taste Profile, the individual metric observation from Figs. 4 and 6, which shows the HE-LIN-Item-KNN achieves the best performance.

5.5 MAP x MRMC

Another crossed-metric that we will analyze is the results from MAP and MRMC. Similar to the MAPxMACE, we want to understand the behavior of each system combination. Figure 11 presents the results from the Movielens, and Table 10 presents the CMC. Figure 12 presents the results from the Taste Profile, and Table 11 presents the CMC values.

5.5.1 Movielens

Similar to the MAPxMACE, we start our analysis with the KL-divergence system combinations. Figure 11a and b, along with Table 10 (Lines 1 and 2), show the results from MAP and MRMC. It is possible to verify that the matrix factorization approaches obtained the best performance in the measure (KL). Table 10 (Lines 1 and 2) shows that the KL-LOG-SVD++ obtained the best performance with a CMC = 3.05. In the LOG trade-off, the second and third place is the SVD (CMC = 4.31) and NMF (CMC = 5.76) sequentially. The SVD++ obtained the best performance, in the LIN, whereas the combination KL-LIN-SVD++ (CMC = 3.52) obtained the second-best performance when we observed all the KL-divergence combinations. So, we can verify that the MAPxMRMC and the MAPxMACE obtain the same recommender places to the KL-divergence.

For the Hellinger divergence measure, we will analyze Fig. 11c and d along with Table 10 (Lines 3 and 4). Like the KL-divergence, the matrix factorization approaches obtained all the best performances. However, the HE-LIN-NMF obtained the best performance to the measure with a CMC = 20.22, followed by the HE-LIN-SVD++ with a CMC = 20.56. To the LOG trade-off, the best performance is HE-LOG-SVD++ with a CMC = 20.95.

As the last analysis, we will observe the Pearson Chi Square ($\chi^2$) combinations taking into account Fig. 11e and f and Table 10 (Lines 5 and 6). Like the MAPxMACE results, and the previously presented measures, the matrix factorization approach obtained all the best performances in the $\chi^2$. However, in this measure, the first place belongs to the SVD++ in the LOG with a CMC = 9.08 and the LIN with a CMC = 9.39.

Based on the CMC results, we can observe that if the system is focused on MAP and MRMC and according to our experimental setup, the best combination to be implemented in the Movielens dataset is the KL-LOG-SVD++ with a CMC = 3.05 as presented in Table 10 Line 2. The
second-best performance belongs to KL-LIN-SVD++. When we observe Fig. 11, it is possible to evidence that the metrics are not directly correlated. We note that the LOG trade-off is part of this research proposal. The CMC compiles, for Movielens, the individual metric observation from Figs. 3 and 7. The CMC indicates the KL-LOG-SVD++ because it has the second best performance in MAP and one of the lower miscalibration. The coefficient does not indicate the CHI-LOG-Slope or the HE-LOG-SVD++ because the first gets a low precision, and the second gets the worst miscalibration. Thus, the CMC indicates a system that is balanced between both metrics.

Fig. 10  Taste Profile - MAP (x) and MACE (y) - Results from LIN and LOG trade-offs combined with the KL-divergence, Hellinger and Pearson Chi-Square
### 5.5.2 Taste profile

Similar to the Movielens analysis, in the Taste Profile, we will start our results by the KL-divergence. Figure 12a and b and Table 11 (Lines 1 and 2) show that the best recommender performance, in the KL-divergence, belongs to the Item-KNN both in LIN with a $CMC = 76.12$ and LOG with a $CMC = 115.76$, respectively. The second and third best recommender performances are the matrix factorization SVD++ and SVD. The Item-KNN system combinations obtain the best performance with a large difference to the second and third places. The worst performance in the KL belongs to the recommender Co Clustering.

Figure 12c and d and Table 11 (Lines 3 and 4) present the measure results for the Hellinger in the Taste Profile. Similar to the KL-divergence, the best performance of the Hellinger is the Item-KNN with the LIN trade-off, as well as the second and third places are the matrix factorization approaches. The Co Clustering approach achieves the worst performance.

As to the Pearson Chi Square presented in Fig. 12e and f and Table 11 (lines 5 and 6), similar to the previous measures and the MAPxMACE results of the Taste Profile, the Item-KNN obtained the best performance to the $\chi^2$. The achieved values to the LIN trade-off by the $CMC$ is 68.46, and to the LOG trade-off is $CMC = 86.13$. The second and third places are the matrix factorizations and the worst performance is with the Co Clustering approach.

Based on the results, the CHI-LIN-Item-KNN is the best combination to be implemented with the Taste Profile. This analysis is based on the $CMC = 68.46$. Observing the table makes it possible to verify that the second-best performance belongs to the KL-LIN-Item-KNN combination with $CMC = 76.12$, followed by the combination CHI-LOG-Item-KNN.

The CMC compiles, for Taste Profile, the individual metric observation from Figs. 4 and 8. The CMC indicates the CHI-LIN-Item-KNN because it has the best performance in MRMC, considering that its performance compensates for the MAP result in the Item-KNN. However, the MAP performance to CHI-LIN-Item-KNN is higher than that achieved by other algorithms.

### 5.6 The decision protocol

The results presented so far help understand the performance of each system combination. Nevertheless, only the CCE and CMC do not indicate the best combination to be implemented in a given domain. Based on this, we will apply the decision protocol to choose the best combination.

#### 5.6.1 Movielens

Table 12 presents the systems combinations performance values to the Movielens dataset. Based on the performance, we can observe that the best combination is CHI-LOG-SVD++ ($s = 12.14$). The Pearson Chi-Square divergence obtained the best results due to its moderate performance in the CCE and CMC. The LOG trade-off balance, a part of our proposal, obtained the best performance in many results in the CCE and CMC evaluation. The SVD++ remains in the first position in the majority of the results of the Movielens. Table 12 presents in bold the best performance for each recommender. The CHI-LIN-SVD++ achieves the second-best performance when all performances are compared. The worst performance is the KNNS.

#### 5.6.2 Taste profile

Table 13 presents the systems’ combinations performance values to the Taste Profile dataset. According to the results, we observe that the best combination is CHI-LIN-Item-KNN ($s = 91.75$). The Pearson Chi-Square divergence achieved the best results due to its best performance in the CMC and, the moderated performance in CCE. The LIN trade-off balance, original [6] proposals, obtained the best
Introducing a framework and a decision protocol...

Fig. 11  Movielens - MAP (x) and MRMC (y) - Results from LIN and LOG trade-offs combined with the KL-divergence, Hellinger and Pearson Chi-Square performance in all results (first and second place). The ItemKNN remains in the first position in all results of the Taste Profile. Like the Movielens tests, Table 13 presents in bold the best performance of each recommender and underlined the best performance. The worst performance belongs to the Co Clustering approach.

5.7 Answering research questions

In this section, we debate each research question presented in Sect. 1.

**RQ1**: Do the calibrated systems perform similarly on the datasets? Are there differences in the system’s behavior?
Table 10 Movielens - CMC - Results from LIN and LOG trade-offs combined with the KL, HE, and CHI divergence measure for each recommender algorithm

| Divergence | Trade-off | Co Clustering | Item KNN | NMF | Slope One | SVD | SVD++ | User KNN |
|------------|-----------|---------------|----------|-----|-----------|-----|-------|----------|
| KL         | LIN       | 60.44         | 90.73    | 3.8 | 63.52     | 4.95| **3.52**| 430.91   |
| KL         | LOG       | 62.94         | 110.46   | 5.76| 83.46     | 4.31| **3.05**| 130.58   |
| HE         | LIN       | 298.94        | 454.46   | 20.22| 278.07    | 31.17| 20.56 | 5319.06  |
| HE         | LOG       | 576.33        | 1654.34  | 83.42| 982.12    | 31.01| **20.95**| 616.98   |
| CHI        | LIN       | 156.53        | 228.18   | 10.52| 136.21    | 13.21| **9.39**| 1367.03  |
| CHI        | LOG       | 184.39        | 294.33   | 15.84| 181.31    | 12.8 | **9.08**| 391.45   |

Among the results, we can observe that each calibrated system has different behavior and performance on the datasets, i.e., there are no best-calibrated systems through the datasets. Each dataset has its system that performs better. This affirmation enhances the decision protocol’s necessity because each dataset presents different results.

**RQ2**: How to find the best-calibrated system for each domain? Evaluating by more than one metric brings a multi-interpretation of the results. It helps to understand how the calibrated recommendation system acts. Using coefficients such as CCE and CMC to comprise the results helps to find the calibrated recommendation system that achieves the best results over all considered metrics. Thus, the proposed decision protocol indicates which calibrated system is the best choice.

**RQ3**: Is it possible to design a common pattern to explain (framework model) calibrated recommendation systems? What are the important components of these systems? Yes, creating a pattern development in the calibrated system context it is possible. As we see in this research, the proposed framework can be reproducible and provides an N-calibrated system. The system components that are more important to the calibrated context are in the post-processing step. There is possibility to expand the framework with new components to help the task. However, the Components Distribution, Trade-off Balance, Trade-off Weight, Select Item Algorithm, Relevance, and Calibration are the fixed ones.

**RQ4**: When the user’s bias is considered in the calibration trade-off, is it possible to improve the system performance? Yes, considering the user’s bias can improve the system performance over all metrics. The Movielens and Taste Profile results show that the performance increases when we consider the user’s bias over the systems that do not use post-processing. This behavior is more observable in the Movielens results.

### 5.8 Discussion

The results show that the matrix factorization approaches are prominent recommenders when used with the calibration methods. In the Taste Profile, the Item-KNN obtains the best performance, followed by the matrix factorization. As the results show, the calibration method increases the precision and reduces the miscalibration.

The LOG trade-off approach shows that it is possible to consider the users’ bias in the equation and increase the overall performance. The framework can be easily applied and creates thousands of system combinations. The proposed protocol can choose the right combination to be implemented, thus facilitating the exploitation of optimal performances.

The claim made by [6, 10] about the drop in precision can be partially evidenced by our results. Other colleagues claim that Pearson achieves competitive results such as [8]. Likewise [8], our findings point towards the possibility of implementing and testing new divergence measures in the calibrated recommendations.

### 6 Related work

In recent years, the calibrated recommendation has attracted attention as a means of achieving fairness. This topic of research addresses different points of view. For instance, some papers in the state-of-the-art analyze the context of calibration [2, 7], others focused on creating a calibrated recommendation list in a user-centric view [6, 8, 10, 13], and others sought to provide more fairness in an item-centric view [17]. Our research focuses on the user-centered view, i.e., a C-fairness system [8], seeking to provide fairness for the consumer.

Most studies argue that fairness can be achieved through pre-, in-, or post-processing [29–31]. Wang et al. [29] divide fairness systems into six groups: Consistent, Calibrated, Envy-free, Counterfactual, Rawlsian Maximin, and Maximin-shared. They argue that calibrated recommendation is also known as merit-based fairness. Pitoura et al. [30] describe four post-processing methods to provide fairness: Generative Process, Constraint Optimization, Calibration, and Multiple Outputs. Calibration seeks to create a recommendation list based on the user’s preference distribution, respecting all the user interest fields. It is achieved by inserting a new step in the recommendation system design.
Introducing a framework and a decision protocol...

Fig. 12  Taste Profile - MAP (x) and MRMC (y) - Results from LIN and LOG trade-offs combined with the KL-divergence, Hellinger and Pearson Chi-Square

the post-processing step. In a sequence, we discuss several related works that generate calibrated recommendation lists by post-processing.

Steck [6] proposes the calibrated recommendations as a way to provide fairness, in which the recommendation list is generated to be the least divergent from the movie genres in the user’s profile. The author implements the collaborative filtering technique through a matrix factorization recommendation algorithm. This algorithm provides a list of post-processed recommendations to ensure a certain degree of fairness. Steck implements the calibrated system as a unique system divided into three steps. The results show the
importance of calibrating recommendations where the objective is to obtain a certain degree of fairness, but the accuracy is slightly reduced. Then, inspired by Steck [6], we implement the linear trade-off, the relevant sum, the distribution, the surrogate submodular, and the Kullback–Leibler. So Steck’s work is covered by our findings. In the results, the systems that implement Kullback–Leibler and the linear trade-off are based on the author’s work. Furthermore, the author suggests that the Hellinger divergence measure may be implemented as a fairness measure. Consequently, we follow Steck’s suggestion. The work uses only movie genres and does not cover other domains such as music or books. Steck [6] debates that calibrated recommendations can be applied in the context of group recommendation. To achieve it there is many possible algorithms and approaches. For instance, it is possible to use group consensus as presented by [3] and [4].

In a similar fashion to Steck [6] and expanding the study about calibrated recommendations, Kaya and Bridge [10] investigate if calibrated recommendations can improve diversification in the recommendation list. Beyond the calibrated systems, the authors implement the Intent-aware approach, seeking to compare the results. As the authors explain, Intent-aware is an approach initially from Information Retrieval (IR) and adapted to the Recommendation System. The approach works with ambiguous representation. For instance, in the requisition for “Jaguar”, the system can have multiple representations for the requisition as cats, cars, or operating systems. By that, the approach seeks to include the most significant number of representations possible in the list. Thus, the study [10] implements two Intent-aware adaptations to recommendation systems, Query Aspect Diversification (xQUAD) and Subprofile-Aware Diversification - (SPAD). The authors compare these two Intent-aware forms with Steck’s [6] calibrated system and extend the system, adding a new way to find the user profile distribution, and comparing four systems. The authors evaluate the results from three perspectives: precision, diversity, and miscalibration. As part of the methodology, the authors use the Movielens 20M and the Taste Profile as datasets, different trade-off weight values from Steck [6], and a fast alternative least-squares matrix factorization recommender. The results indicate a relation that intent-aware calibrates the recommendation list to some degree, and calibrated recommendations somehow imply a diverse recommendation. To the Taste Profile dataset, the calibrated system with the distribution based on subprofile increases the precision when the trade-off weight is high. As Steck [6], Kaya and Bridge [10] results indicate, the higher the trade-off value, the more calibrated the list. From Kaya and Bridge [10], we cover all implementations inspired by Steck [6]. We do not cover diversification in our study.

Abdollahpouri et al. [2] study a connection between popularity bias and calibration. The short paper is focused on analyzing the calibration aspect and its fairness. It is argued that the groups that receive popularity bias amplification will also receive a higher miscalibration degree. Beyond, it is argued that if the popularity bias is fixed, the disparate miscalibration can be fixed, too, turning the system fairer. To create the groups, the authors divide all users into ten groups

| Table 11 | Taste Profile - CMC - Results from LIN and LOG trade-offs combined with the KL, HE, and CHI divergence measure for each recommender algorithm |
|----------|-------------------------------------------------------------------------------------------------------------------------------------|
| Divergence | Trade-off | Co Clustering | Item KNN | NMF | Slope One | SVD | SVD++ | User KNN |
| KL       | LIN       | 25124.25 | 76.12 | 2299.61 | 6598.41 | 953.96 | 624.99 | 5331.51 |
| KL       | LOG       | 22937.04 | 115.76 | 2299.61 | 6598.41 | 953.96 | 624.99 | 5331.51 |
| HE       | LIN       | 42028.72 | 119.23 | 3775.94 | 11175.3 | 1635.21 | 1452.92 | 9181.55 |
| HE       | LOG       | 42531.68 | 239.19 | 3775.94 | 11175.3 | 1635.21 | 1452.92 | 9181.55 |
| CHI      | LIN       | 12640.08 | 68.46 | 1987.01 | 3937.01 | 798.85 | 687.41 | 2939.81 |
| CHI      | LOG       | 11985.59 | 86.13 | 1987.01 | 3937.01 | 798.85 | 687.41 | 2939.81 |

| Table 12 | Protocol performance values to each system combination using the Movielens dataset |
|----------|-------------------------------------------------------------------------------------------------------------------------------------|
| Divergence | Trade-off | Co Clustering | Item KNN | NMF | Slope One | SVD | SVD++ | User KNN |
| KL       | LIN       | 244.1   | 365.4  | 16.7 | 224.86 | 19.83 | 14.06 | 1682.22 |
| KL       | LOG       | 268.94  | 484.4  | 26.8 | 321.3  | 18.65 | 13.14 | 575    |
| HE       | LIN       | 333.64  | 511.86 | 22.73 | 318.35 | 35.46 | 22.94 | 6030.3 |
| HE       | LOG       | 634.17  | 1811.66| 91.73 | 1086.53| 34.07 | 23.03 | 677.71 |
| CHI      | LIN       | 194.79  | 291.51 | 13.44 | 183.62 | 17.5  | 12.45 | 1731.47 |
| CHI      | LOG       | 235.02  | 382.99 | 20.67 | 252.74 | 17.1  | 12.14 | 512.16 |
Table 13  Protocol performance values to each system combination using the Taste Profile dataset

| Divergence | Trade-off | Co Clustering | Item KNN | NMF     | Slope One | SVD     | SVD++   | User KNN |
|------------|-----------|---------------|----------|---------|-----------|---------|---------|----------|
| KL         | LIN       | 30217.86      | 101.85   | 2967.28 | 8069.44   | 1443.41 | 954.38  | 6507.13  |
| KL         | LOG       | 28136.9       | 142.44   | 2967.28 | 8069.44   | 1443.41 | 954.38  | 6507.13  |
| HE         | LIN       | 47101.47      | 133.62   | 4235.29 | 12581.75  | 1939.16 | 1731.9  | 10329.72 |
| HE         | LOG       | 47589.66      | 260.51   | 21338.53| 15365.32  | 5953.06 | 5281.65 | 11126.32 |
| CHI        | LIN       | 16948.48      | 91.75    | 2456.91 | 5375.44   | 1107.25 | 970.79  | 3987.65  |
| CHI        | LOG       | 16428.1       | 113.77   | 13507.54| 9956.75   | 3478.12 | 3228.84 | 7161.56  |

based on the degree of popular items each user likes. They do not implement a post-processing step; the connection is between the user’s preferences and five recommender algorithms and their recommendation list. Abdollahpouri et al. [2] use a different genre distribution formulation without probability and rank weight, as proposed in previous calibrated work. As a dataset, they use Movielens 1 M and core-10 Yahoo Movie and evaluate with the miscalibration metric proposed by Steck [6] and algorithmic popularity lift proposed by the authors in previous work. The results indicate that the users with less popular items are affected by the popularity bias and receive more miscalibrated recommendations.

Lin et al. [7] analyze several categories of profile characteristics over many factors contributing to miscalibration for some users but not others. These characteristics are provided by the users’ profiles and can be interpreted from the users’ preferences. As in our work, the category described by Lin et al. [7] is a generic word for class, tag, or genre. To evaluate the categories, authors use the miscalibration proposed by Steck [6] and the normalized Discounted Cumulative Gain (NDCG). As datasets, they use Movielens 1 M and Yelp.com. On the other hand, Abdollahpouri et al. [2] and Lin et al. [7] do not implement the post-processing step and use five well-known algorithms. The results show that the miscalibration is present in different degrees for all algorithms and both datasets. They analyze the characteristics depending on the miscalibration present at different degrees too.

Seymen et al. [12] propose a new trade-off balance called Calib-Opt. As a calibration measure, the authors use the proposed Weighted Total Variation. In the rank measure, they consider two new constraints. The selector algorithm used by Steck [6], surrogate submodular, is substituted by Gurobi branch&bound. The results indicate that the proposed Calib-Opt has better performance than Steck [6].

da Silva et al. [8] propose a set of calibrated recommendation systems and metrics. The authors use the divergence measure Kullback–Leibler similar to Steck and Kaya and Bridge [6, 10], and propose the Hellinger and Pearson Chi-Square. To evaluate the systems, da Silva et al. [8] propose two new metrics for the calibrated context, the Mean Average

Table 14 Most important related work and the comparison with our proposal

| Components        | Steck [6]          | Kaya and Bridge [10] | da Silva et al. [8] | Our Proposal |
|-------------------|--------------------|----------------------|---------------------|--------------|
| Distribution      | Genre distribution | Subprofile distribution | Genre distribution | Genre distribution |
| Relevance         | Relevance Summation| Relevance Summation   | Relevance Summation | Relevance Summation |
| Calibration       | Kullback–Leibler   | Kullback–Leibler     | Kullback–Leibler    | Kullback–Leibler |
| Trade-off Weight  | 0.0, 0.2, 0.5, 0.8, 0.9, 0.95, 0.99 | 0.0, 0.1, 0.2, 0.3, ..., 1.0 | 0.0, 0.1, 0.2, 0.3, ..., 1.0 | 0.0, 0.1, 0.2, 0.3, ..., 1.0 |
| Trade-off Balance | Linear             | Linear               | Linear              | Linear |
| Select Item       | Surrogate          | Surrogate            | Surrogate           | Surrogate |
| Algorithm         | Submodular         | Submodular           | Submodular          | Submodular |
| Metrics           | Miscalibration     | Diversity            | MAP                 | MAP |
|                   | Accuracy           | Precision            | MRR                 | MRR |
|                   |                    | NDCG                 | MACE                | MACE |
|                   |                    |                      | MRMC                | MRMC |
|                   |                    |                      | CCE                 | CCE * |
|                   |                    |                      | CMR                 | CMR * |
Calibration Error (MACE) and the Mean Rank MisCalibration (MRMC). To evaluate, the authors use six well-known collaborative filtering recommender algorithms. The results indicate that the Pearson Chi-Square can perform better than other divergence measures. From this work, we implement the Hellinger and Pearson Chi-Square. Thus, all systems that implement these two fairness measures are inspired by this work.

Pitoura et al. [9] conduct an overview of fairness in ranking and recommendations. The authors argue that calibrated recommendations are one model of fairness, with six in total: demographic parity, conditional parity, equalized odds, fairness through awareness, counterfactual fairness, and calibration-based fairness. In the overview, the authors define five viewpoints on fairness: fairness for the recommended items, fairness for the users, fairness for groups of users, fairness for the item providers, and the recommendation platform. The study considers that post-processing is the core for fairness systems due to the step that ensures fairness, modifying the recommender algorithm output.

Naghiaeia et al. [32] explain that the calibrated systems must consider the profile size. They argue that the calibrated recommendations do not include confidence in the equation. To solve it, the authors propose a new trade-off balance called \textit{CCL} which considers the item diversity besides the relevance and calibration. Two datasets were used for the experiment: MovieLens Small and MovieLens 1M. The authors implement four recommender algorithms as the baseline. Besides, the authors implement Seymen et al. [12] system. The results indicate that their proposal increase precision, recall, NDCG, diversity, and catalog coverage. The miscalibration achieves the best values when compared with the baselines. However, the authors’ proposal does not beat the Seymen system in reducing the miscalibration.

Nazari et al. [33] propose understanding the podcast recommendation system using two types of data play times and subscriptions. They implement the Multi-Layer Perceptron (MLP) to produce the candidate items. To test the proposals, they apply offline and online experiments, using the Spotify Podcast database to select 800k users. The metrics Precision, NDCG, and Coverage were used to evaluate. The authors found that calibration for both data types improves the precision and the item coverage. The results show that some podcast genres gain more attention on the recommendation, and the most popular ones reduce their exposure, indicating a calibrated recommendation.

Abdollahpouri et al. [34] investigate the calibrated recommendations as a Minimum-Cost Flow (MCF) problem. The authors re-write the Linear trade-off proposed by [6], changing the relevance equation to the Maximum Weight Assignment and the calibration to Penalized Miscalibration Score. For the experiment, two datasets were used, the MovieLens 20M and Last.fm 1B. To evaluate the proposals, the authors implement five metrics Relevance, Precision, Recall, NDCG, and Miscalibration. As a result, Abdollahpouri et al. [34] found that as higher the list relevance, the higher the miscalibration. The MCF achieves the best precision, recall, and NDCG. The Gurobi achieved the lowest miscalibration.

We highlight the most important works that inspire our work from the related works presented in this section. We call attention to our trade-off contribution and coefficients in Table 14. In addition, this work is the one that most implement calibrated recommendation systems compared with the related works. The \* represents our implementation proposals, conceptual framework, and decision protocol. As the results show, our proposed approach LOG is an outstanding advantage compared to the state-of-the-art due to considering the user’s bias in the calibration process. Analyzing the coefficients in that proposal can help the system specialist understand the degree of precision and calibration in a single value.

7 Final remarks and future work

This research exploited three related fronts of calibrated recommendations in a user-centric view. First, we show that the state-of-the-art needs to consider the users’ bias in the trade-off balance. Based on this premise, we propose the LOG trade-off to increase MAP, MACE, and MRMC performance. We also showed that the state-of-the-art explores different numbers of implementations without an automatic way to decide the best implementation. To tackle this problem, we propose two coefficients and a decision protocol that indicates the best implementation to the dataset. In addition, we showed the possibilities of implementing the calibrated system following the framework division in components to provide an easier way to be implemented and tested.

In future work, we plan to investigate other calibration measures and compare the effect of post-processing in collaborative filtering and content-based filtering. Another extension is to explore different types of distribution. For instance, to compare the normalized against the non-normalized ones. In addition, we will investigate other measures that can be used on the Relevance Component. Most studies in calibrated recommendations recommend items to a single user. Thus, in future work, we plan to apply calibrated recommendations to a group recommendation context.

Acknowledgements This study was financed by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 88887.502736 / 2020-00. The authors acknowledge the National Laboratory for Scientific Computing (LNCC/MCTI, Brazil) for providing HPC resources of the SDumont supercomputer, which has contributed to the research results reported within this paper. URL: http://sdumont.lncc.br.
Introducing a framework and a decision protocol... 22071

Author Contributions Diego Corrêa da Silva: Conceptualization, Methodology, Software, Formal Analysis, Investigation, Data Curation, Writing - Original Draft, Funding Acquisition. Frederico Araújo Durão: Conceptualization, Methodology, Writing - Review & Editing, Supervision, Project Administration, Funding Acquisition.

Funding This study was financed by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 88887.502736 / 2020-00.

Data Availability The MovieLens and Taste Profile are public datasets

Code Availability https://github.com/DiegoCorrea/masters-dissertation

Declarations

Ethics approval Not Applicable.

Conflicts of interest The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Kaminskas M, Bridge D (2016) Diversity, serendipity, novelty, and coverage: A survey and empirical analysis of beyond-accuracy objectives in recommender systems. ACM Trans Interact Intell Syst 7(1). https://doi.org/10.1145/2926720
2. Abdollahpouri H, Mansoury M, Burke R, Mobasher B (2020) The connection between popularity bias, calibration, and fairness in recommendation. In: Fourteenth ACM Conference on Recommender Systems. RecSys ’20, pp. 726–731. Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/3383313.3418487
3. Ji F, Wu J, Chichlana F, Wang S, Fujita H, Herrera-Viedma E (2023) The overlapping community driven feedback mechanism to support consensus in social network group decision making. IEEE Trans Fuzzy Syst 1–15. https://doi.org/10.1109/TFUZZ.2023.3241062
4. Xing Y, Wu J, Chichlana F, Yu G, Cao M, Herrera-Viedma E (2023) A bargaining game based feedback mechanism to support consensus in dynamic social network group decision making. Information Fusion 93:363–382. https://doi.org/10.1016/j.inffus.2023.01.004
5. Abdollahpouri H, Burke R, Mobasher B (2017) Controlling popularity bias in learning-to-rank recommendation. In: Proceedings of the Eleventh ACM Conference on Recommender Systems. RecSys ’17, pp. 42–46. ACM, New York, NY, USA. https://doi.org/10.1145/3109859.3109912
6. Steck H (2018) Calibrated recommendations. In: 12th ACM Conference on Recommender Systems. RecSys ’18, pp. 154–162. ACM, New York, NY, USA. https://doi.org/10.1145/3240323.3240372
7. Lin K, Sonboli N, Mobasher B, Burke R (2020) Calibration in collaborative filtering recommender systems: A user-centered analysis. In: 31st ACM Conference on Hypertext and Social Media. HT ’20, pp. 197–206. Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/3372923.3404793
8. da Silva DC, Manzato MG, Durão FA (2021) Exploiting personalized calibration and metrics for fairness recommendation. Expert Systems with Applications 181, 115112. https://doi.org/10.1016/j.eswa.2021.115112
9. Pitoura E, Stefanidis K, Koutrika G (2021) Fairness in rankings and recommendations: an overview. The VLDB Journal. https://doi.org/10.1007/s00778-021-00697-y
10. Kaya M, Bridge D (2019) A comparison of calibrated and intent-aware recommendations. In: 13th ACM Conference on Recommender Systems. RecSys ’19, pp. 151–159. Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/3298689.3347045
11. Starychfojtu J, Peska L (2020) Smartreceipes: Towards cooking and food shopping integration via mobile recipes recommender system. In: Proceedings of the 22nd International Conference on Information Integration and Web-Based Applications; Services. iiWAS ’20, pp. 144–148. Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/3428737.3429096
12. Seymen S, Abdollahpouri H, Malthouse EC (2021) A constrained optimization approach for calibrated recommendations. In: Fifteenth ACM Conference on Recommender Systems. RecSys ’21, pp. 607–612. Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/3460231.3478857
13. Abdollahpouri H, Mansoury M, Burke R, Mobasher B, Malthouse E (2021) User-Centered Evaluation of Popularity Bias in Recommender Systems, pp. 119–129. Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/3450613.3456821
14. Sacharidis D, Mouratidis K, Klefogiannis D (2019) A common approach for consumer and provider fairness in recommendations. In: 13th ACM Conference on Recommender Systems. Rec Sys ’19. ACM, New York, NY, USA. http://ceur-ws.org/Vol-2431/paper1.pdf
15. Zhao X, Zhu Z, Caveree J (2021) Rabbit Holes and Taste Distortion: Distribution-Aware Recommendation with Evolving Interests, pp. 888–899. Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/3442381.3450099
16. Cha S-H (2007) Comprehensive survey on distance/similarity measures between probability density functions. International Journal of Mathematical Models and Methods in Applied Sciences 1(4):300–307
17. Zhao X, Zhu Z, Alifii M, Caveree J (2020) Addressing the target customer distortion problem in recommender systems. In: The Web Conference 2020. WWW ’20, pp. 2969–2975. Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/3366423.3380065
18. Carbonell J, Goldstein J (1998) The use of mmr, diversity-based reranking for reordering documents and producing summaries. In: 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. SIGIR ’98, pp. 335–336, ACM, New York, NY, USA. https://doi.org/10.1145/299041.291025
19. Koren Y, Bell R (2015) Advances in collaborative filtering. In: Ricci, F., Rokach, L., Shapira, B. (eds.) Recommender Systems Handbook, pp. 77–118. Springer, Boston, MA. https://doi.org/10.1007/978-1-4899-7637-6_3
20. Nemhauser GL, Wolsey LA, Fisher ML (1978) An analysis of approximations for maximizing submodular set functions-I. Math Programming 14(1):265–294. https://doi.org/10.1007/BF01588971
21. Harper FM, Konstan JA (2015) The movielens datasets: History and context. ACM Trans. Interact Intell Syst 5(4). https://doi.org/10.1145/2827972
22. Koren Y (2010) Factor in the neighbors: Scalable and accurate collaborative filtering. ACM Trans Knowl Discov Data 4(1):1–1124. https://doi.org/10.1145/1644873.1644874
23. Lemire D, Maclachlan A (2007) Slope one predictors for online rating-based collaborative filtering, 2005 SIAM International Conference on Data Mining, SDM 2005 5. https://doi.org/10.1137/1.9781611972757.43
24. Luo X, Zhou M, Xia Y, Zhu Q (2014) An efficient non-negative matrix-factorization-based approach to collaborative filtering for recommender systems. IEEE Trans Ind Inf 10(2):1273–1284. https://doi.org/10.1109/TII.2014.2308433

25. Koren Y, Bell R, Volinsky C (2009) Matrix factorization techniques for recommender systems. Computer 42(8):30–37. https://doi.org/10.1109/MC.2009.263

26. Koren Y (2008) Factorization meets the neighborhood: A multifaceted collaborative filtering model. In: 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD ’08, pp. 426–434. ACM, New York, NY, USA. https://doi.org/10.1145/1401890.1401944

27. George T, Merugu S (2005) A scalable collaborative filtering framework based on co-clustering. In: Fifth IEEE International Conference on Data Mining (ICDM’05). ICDM ’05, pp. 625–628. IEEE Computer Society, USA. https://doi.org/10.1109/ICDM.2005.14

28. Hug N (2020) Surprise: A python library for recommender systems. Journal of Open Source Software 5, 2174. https://doi.org/10.21105/joss.02174

29. Wang Y, Ma W, Zhang M, Liu Y, Ma S (2022) A survey on the fairness of recommender systems. ACM Trans Inf Syst. https://doi.org/10.1145/3547333. Just Accepted

30. Piotoura E, Stefanidis K, Koutrika G (2021) Fairness in rankings and recommendations: An overview. The VLDB Journal 31(3):431–458. https://doi.org/10.1007/s00778-021-00697-y

31. Wang Y, Ma W, Zhang M, Liu Y, Ma S (2023) A survey on the fairness of recommender systems. ACM Trans Inf Syst 41(3). https://doi.org/10.1145/3547333

32. Naghiaei M, Rahmani HA, Aliannejadi M, Sonboli N (2022) Towards confidence-aware calibrated recommendation. In: Proceedings of the 31st ACM International Conference on Information and Knowledge Management, CIKM ’22, pp. 4344–4348. Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/3511808.3557713

33. Nazari Z, Chandar P, Fazelnia G, Edwards CM, Carterette B, Lalmas M (2022) Choice of implicit signal matters: Accounting for user aspirations in podcast recommendations. In: Proceedings of the ACM Web Conference 2022. WWW ’22, pp. 2433–2441. Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/3485447.3512115

34. Abdollahpouri H, Nazari Z, Gain A, Gibson C, Dimakopoulou M, Anderson J, Carterette B, Lalmas M, Jehara T (2023) Calibrated recommendations as a minimum-cost flow problem. In: Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining, WSDM ’23, pp. 571–579. Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/3539597.3570402

Publisher’s Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Diego Corrêa da Silva received his B.Sc and M.Sc in Computer Science from the Federal University of Bahia, Brazil 2018 and 2021. At the time of this paper, he is a D.Sc candidate in Computer Science at the Federal University of Bahia. His main research interests include Recommendation Systems and Machine Learning.

Frederico Araújo Durão received his B.Sc in Computer Science from Faculdade Ruy Barbosa, M.Sc in Computer Science from Federal University of Pernambuco, Brazil, in 2004 and 2008, and his D.Sc in Computer Science from Aalborg University, Denmark, 2012. At the time of this paper, he is a permanent professor at the Federal University of Bahia. His main research interests include Recommender Systems, Information Retrieval, and the Web of Data.