Les Cahiers du GERAD ISSN: 0711–2440

Estimating serendipity in content-based recommender systems

N. Tuval, A. Hertz, T. Kuflik

G–2023–35

August 2023

Citation suggérée : N. Tuval, A. Hertz, T. Kuflik (Août 2023). Estimating serendipity in content-based recommender systems, Rapport technique, Les Cahiers du GERAD G– 2023–35, GERAD, HEC Montréal, Canada.

Avant de citer ce rapport technique, veuillez visiter notre site Web (https://www.gerad.ca/fr/papers/G-2023-35) afin de mettre à jour vos données de référence, s'il a été publié dans une revue scientifique.

Suggested citation: N. Tuval, A. Hertz, T. Kuflik (August 2023). Estimating serendipity in content-based recommender systems, Technical report, Les Cahiers du GERAD G–2023–35, GERAD, HEC Montréal, Canada.

Before citing this technical report, please visit our website (https://www.gerad.ca/en/papers/G-2023-35) to update your reference data, if it has been published in a scientific journal.

La publication de ces rapports de recherche est rendue possible grâce au soutien de HEC Montréal, Polytechnique Montréal, Université McGill, Université du Québec à Montréal, ainsi que du Fonds de recherche du Québec – Nature et technologies.

Dépôt légal – Bibliothèque et Archives nationales du Québec, 2023
– Bibliothèque et Archives Canada, 2023

The publication of these research reports is made possible thanks to the support of HEC Montréal, Polytechnique Montréal, McGill University, Université du Québec à Montréal, as well as the Fonds de recherche du Québec – Nature et technologies.

Legal deposit – Bibliothèque et Archives nationales du Québec, 2023
– Library and Archives Canada, 2023

GERAD HEC Montréal
3000, chemin de la Côte-Sainte-Catherine
Montréal (Québec) Canada H3T 2A7

Tél. : 514 340-6053
Téléc. : 514 340-5665
info@gerad.ca
www.gerad.ca

Estimating serendipity in content-based recommender systems

Noa Tuval [a]
Alain Hertz [b, c]
Tsvika Kuflik [a]

[a] *Information Systems Department, The University of Haifa, Haifa, Israel, 3498838*

[b] *Department of Mathematics and Industrial Engineering, Polytechnique Montréal, Montréal (Qc), Canada, H3T 1J4*

[c] *GERAD, Montréal (Qc), Canada, H3T 1J4*

alain.hertz@gerad.ca

August 2023
Les Cahiers du GERAD
G–2023–35

Abstract : Recommender systems provide personalized recommendations to their users for items and services. They do that using a model that is tailored to each user to infer their preferences based on their characteristics and previous interactions they have made with the system. Recent research suggests that users of a recommender system may like to receive suggestions that provide a pleasant surprise. In other words, a recommendation may be unexpected to the user, but it must be useful. This concept, called serendipity, is one of the aspects that have been proposed to meet user expectations for the recommendations they receive. Introducing serendipity means going beyond the 'more of the same' aspect that past recommender systems are criticized for. In this article, we first show how to estimate user preferences based on ratings they have done in the past in a content-based recommender system. This estimation allows us to measure the relevance of a recommendation. We then determine the item attributes that play an important role in the relevance measure. Experiments in the movie domain show that the greater the relevance of a recommendation, the more the users seem willing to discover items having attributes with which they are not familiar, as long as these do not play an important role in their ratings.

Keywords : Recommender systems, serendipity, estimating user preferences

1 Introduction

The main purpose of a recommender system is to provide users with relevant and useful recommendations [19]. In order to produce personalized recommendations, a system collects information about user preferences and builds user models that allows to offer recommendations for items that suit their tastes. Traditionally, recommender systems are evaluated focusing primarily on the accuracy of predictions of user ratings [3]. As a result, the recommended items tend to be similar to items to which the user has given a high rating, which means that the users are not exposed to items that deviate from what they have already rated, even though some of those items may be of interest to them. This leads to the *serendipity problem* which can be defined as over-specialization in recommendations [6, 18]. Users receive recommendations for similar items to those they liked in the past, whereas they may also be interested in recommendations for surprising and unexpected items [1, 10, 14, 23]. For example, a user of a recommender system who gives high ratings to Mexican restaurants will receive additional recommendations for Mexican restaurants, which limits the possibility of discovering new tastes and experimenting with restaurants that are different from those the user is familiar with. On the other hand, not every item that is different from the items the user is familiar with can be a *pleasant surprise* for the user. For example, it is unlikely to surprise a user who likes to watch romantic dramas by recommending a horror movie. Hence, the challenge is to produce un-obvious recommendations that are also relevant for the user [1].

Serendipity is one of the beyond-accuracy metrics proposed for measuring the quality of recommendations [8, 17]. Although the meaning of serendipity is rather vague and has not yet been precisely defined, its emotional aspect can be interpreted as surprise, unexpectedness and novelty [10].

In this paper, we first show how to estimate the user preferences in a content-based recommender system, on the basis of their previous ratings. This is done by solving a mixed integer linear program. The information necessary for generating these estimations can be kept locally and does not require the knowledge of the ratings of other users and does not require sharing it with any service provider. This means that we can maintain user privacy and the proposed approach can be implemented within the user's controlled area (e.g., a private space on the cloud or a personal device).

For each item attribute, we predict whether the user likes it, does not care about it, or dislikes it. This *user profile* is determined by solving a mixed integer linear program (MILP) which is an adaptation of the model presented in [4]. We can thus estimate the relevance of a recommendation by defining a distance between the recommended item and the user profile. We then determine which item attributes seem important to the user, in that they greatly influence their ratings.

Unexpectedness and novelty can be measured using a distance between the recommended item and the items that the user has rated in the past. Roughly speaking, such a distance corresponds to the similarity between the recommended item with those to which the user was exposed in the past [21].

Experiments performed in the movie domain show that there is a strong relation between the relevance of a serendipitous recommendation and its similarity with items that the user has rated in the past. Kotkov et al. [10] found that high expected ratings can be a predictor for serendipity. Nevertheless, Wang and Chen [22] showed that users of the movie domain focus more on the unexpectedness component of serendipity than on the relevance component when choosing movies. We extend these research results by showing that the greater the relevance of a recommendation, the more the users seem willing to be surprised, as long as the unexpected item's attributes are not those that play an important role in their ratings.

In summary, our contributions extend past research and can be summarized as follows.

- We propose a mixed integer linear program to estimate user preferences in content-based recommender systems on the basis of their previous ratings; it is an improved version of the model proposed in [4].

- We determine which item attributes seem to have the biggest impact on the user ratings.
- We show how to estimate the serendipity of a recommendation by comparing its relevance with its similarity to previous rated items. Serendipitous recommendations with a high relevance are more distant to items that the user was exposed to, but the unexpected item attributes are not those that have a high impact on the user's ratings.
- The proposed approach maintains user privacy and can be implemented within the user's controlled area (e.g., a private space on the cloud or a personal device).

The paper is structured as follows: A brief literature review is given in the next section. The mixed integer linear program that builds user profiles is described in Section 3. Section 4 is devoted to similarity and relevance measures. Computational experiments are reported in Section 5 with an analysis of the results. Concluding remarks, limitations of our approach and future work are mentioned in Section 6.

2 Brief literature review

The literature dealing with serendipity in recommender systems is not very abundant. The survey written by Ziarani et al. [24] gives a very detailed review on the subject and we invite interested readers to consult this excellent article if they want to have more details than what we are going to give now.

According to our understanding, the prevailing approach to improve serendipity in recommender systems is to first define the characteristics of a serendipitous recommendation and then determine metrics that apply these characteristics to the recommender system and thus allow it to produce serendipitous recommendations. Evaluating a recommender system for its ability to produce serendipitous recommendations is not a trivial task since the definition of serendipity is not unique, hence allowing for different interpretations of the concept. For example, according to [1, 2, 16], serendipity is related to unexpectedness and relevance, while in [6], novelty is included in the definition of serendipity. In [7] a recommendation is considered serendipitous if it is unexpected and interesting for the user, while in [15], it must also be pleasant.

In many studies the interpretation of serendipity is based on a similarity measure between a candidate item and a set of items. For example Vargas and Castells [21] define a the distance between the recommended item and those rated by the user in the past, while Adamopoulos and Tuzhilin [1] consider a distance from the recommended item to the items that are expected to be recommended to the user. A random-walk algorithm is proposed in [2] where recommendations are produced by moving in a graph having the items as vertex set and where the probability of moving from a vertex to another depends on the similarity between the two corresponding items. This method uses external sources like Wikipedia to reveal hidden links between items and thus weight the edges linking them, the aim being to increase the likelihood that users will view recommendations made to them as non-obvious. As noted in [12] and [20], this method generates recommendations that are not biased towards popular items, but does not consider user preferences.

Kito et al.[9] have analyzed serendipity in the music domain using acoustics and metadata such as the artist's name, the title and the release year. They have tried to find relations between serendipitous music recommendations, metadata similarities and an acoustic-based distance from the music that the user likes. Their conclusion is that a serendipitous music recommendation does not necessarily have both a low metadata similarity and a low acoustic distance to the user's preferred music.

Kotkov et al. [10] collected data from users of the MovieLens web-based recommender system (`https://grouplens.org/datasets/serendipity-2018/`) regarding the serendipity of the recommendations they received. The users were asked to rate their agreement with six statements regarding two important aspects that define serendipity, namely novelty and unexpectedness. The six statements, denoted $s_1, \ldots, s_6$, are as follows:

- Novelty components of serendipity:
  - s_1: The first time I heard of this movie was when MovieLens suggested it to me.
  - s_2: MovieLens influenced my decision to watch this movie.
- Unexpectedness components of serendipity:
  - s_3: I expected to enjoy this movie before watching it for the first time.
  - s_4: This is the type of movie I would not normally discover on my own; I need a recommender system like MovieLens to find movies like this one.
  - s_5: This movie is different (e.g., in style, genre, topic) from the movies I usually watch.
  - s_6: I was (or, would have been) surprised that MovieLens picked this movie to recommend to me.

The above-mentioned dataset contains responses of 475 participants regarding 2146 movies. It is shown in [10] that movie recommendations for which the users have given a positive answer to s_1, s_2, s_4, s_5 or s_6 broaden their preferences more than movies users found non-novel and non-unexpected. However, a positive answer to statement s_3 seems to hurt user satisfaction. The same authors have coupled s_1 or s_2 with s_4, s_5 or s_6 to create six definitions of serendipity, and have shown that movies that are serendipitous according to at least one of the six definitions have higher predicted ratings than corresponding non-serendipitous movies.

In [11], a serendipity-oriented algorithm is proposed which improves serendipity of recommendations through attribute diversification and helps overcome the overspecialization problem. This algorithm diversifies a top-n list of recommendations generated by an accuracy-oriented algorithm by adding items with a high serendipity score that is defined as a linear combination of four parameters important for serendipity, namely, relevance, diversity, dissimilarity of an item with the user profile, and unpopularity. Experiments on the above-mentioned dataset show that in terms of accuracy, this algorithm outperforms other serendipity-oriented algorithms [10], but underperforms accuracy-oriented algorithms with diversification such as the technique proposed in [25].

In [22], additional experiments on the same dataset show that novelty aspects (i.e., s_1 and s_2) do not engender significant effect on the serendipity, while unexpectedness aspects derived from s_4, s_5 or s_6 can lead users to broaden their preferences. Also, they suggest to measure serendipity by combining the answers to s_4, s_5 and s_6.

Our work is in line with the studies in [10, 11, 22] and is described in the next sections. We reveal a strong correlation between the estimated relevance of a serendipitous recommendation and a similarity measure between a recommended item and the items that the user has evaluated in the past. This seems to be especially true when serendipity is defined on the basis of positive responses to statements s_1, s_4, s_5 and s_6.

3 A mixed integer linear program to estimate user profiles

The recommendations in the above-mentioned movie database on which the users had to give their opinion to indicate whether they considered them to be unexpected and presenting aspects of novelty all have a known level of relevance for the users since we know the ratings which they attributed to these recommendations. This relevance level was used, for example, to skew the database by keeping only recommendations for which users gave a score of at least 7 stars out of 10. However, in real situations, when recommendations are made to users, these relevance levels remain unknown until the users decide to evaluate what has been suggested to them. We propose here to estimate the relevance of recommendations by creating user profiles that fit the ratings the users have given in the past. While the proposed technique is conceptually similar to what is done in [4], we describe here an improved model that is computationally more efficient.

Let A be an ordered set of n Boolean attributes, and let I be a set of items of a recommender system. Let Q_n be the n-dimensional hypercube with vertex set $\{0,1\}^n$, and where two vertices $\boldsymbol{x}$ and $\boldsymbol{y}$ are linked with an edge if and only if the Hamming distance $d(\boldsymbol{x},\boldsymbol{y})$ between $\boldsymbol{x}$ and $\boldsymbol{y}$ (i.e., the number of attributes $j \in A$ such that $x_j \neq y_j$) equals 1. All items in I are represented as vertices in Q_n. More precisely, a vertex $\boldsymbol{v}^i = (v^i_1, \ldots, v^i_n)$ of Q_n is associated with every item $i \in I$ so that $v^i_j = 1$ if i has the jth attribute in A, and $v^i_j = 0$ otherwise. Note that two items with the same attributes are associated with the same vertex in Q_n. We can therefore consider every vertex of Q_n as an *item type*.

Every user u of the recommender system is represented by a vector $\boldsymbol{w}^u$ in $\{-1,0,1\}^n$ so that $w^u_j = -1$ if u does not like the jth attribute, $w^u_j = 0$ if u does not care about it, and $w^u_j = 1$ if u likes it. If $w^u_j = 0$, the ratings of u do not depend on the value of the jth attribute. To take this into account, we consider the distance $d : \{0,1\}^n \times \{-1,0,1\}^n \to \{0,\ldots,n\}$ which, given a vertex $\boldsymbol{v} \in \{0,1\}^n$ and a vertex $\boldsymbol{x} \in \{-1,0,1\}^n$, counts the number of components j with $v_j = 1$ and $x_j = -1$ (i.e., the item has attribute j which the user does not like), or $v_j = 0$ and $x_j = 1$ (i.e., the item does not have the attribute j that the user likes).

Let $I' \subseteq I$ be a subset of m items rated by u according to an s-star scale, where the highest score of s stars is given by u to items that perfectly match the user preferences, and the lowest score of 1 star when the user did not like any of the attribute values of the rated item. So let r_i be the rating given by u to an item $i \in I'$, using an s-star scale. This rating can be translated into a distance δ_i, called d-rating using function $\tau : \{1,\ldots,s\} \longrightarrow [0,n]$ which is defined as follows:

$$\delta_i = \tau\left(r_i\right) = n - \frac{n\left(r_i - 1\right)}{s-1}.$$

If u likes all attributes of an item $i \in I'$, the user's rating r_i should be equal to s stars, which we translate into $\delta_i = 0$. On the contrary, if u does not like any of the attributes of an item $i \in I'$, then r_i should be equal to 1 star, which we translate into $\delta_i = n$.

In order to estimate the vector $\boldsymbol{w}^u$ that models the preferences of user u, we determine a vertex $\boldsymbol{x}$ in $\{-1,0,1\}^n$ that fits the user's ratings. More precisely, we estimate that the d-rating δ_i of user u for item i will be the distance $d(\boldsymbol{v^i},\boldsymbol{x})$. If u likes all the attributes present in an item i and dislikes all the others, then $d'(\boldsymbol{v^i},\boldsymbol{x}) = 0$ which means that r_i will probably be equal to s, which corresponds to a d-rating $\delta_i = 0$. Conversely, if u dislikes all the attribute present in i and likes all the others, then $d(\boldsymbol{v^i},\boldsymbol{x}) = n$ and r_i will probably be equal to 1, which corresponds to a d-rating $\delta_i = n$. The cumulative error with all these estimations is

$$f_(x) = \sum_{i \in I'} |d(\boldsymbol{v}^i,\boldsymbol{x}) - \delta_i|.$$

In order to minimize this value, we solve a mixed integer linear program (MILP) which we now define. The MILP proposed in [4] for performing the above task has a variable y_{ij} for every item $i \in I'$ and every attribute $j \in A$ with the following meaning:

$$y_{ij} = \begin{cases} 1 & \text{if } v_j = 1 \text{ and } x_j = -1 \text{ or } v_j = 0 \text{ and } x_j = 1 \\ 0 & \text{otherwise} \end{cases}$$

In words, $y_{ij} = 1$ if item i has attribute j which the user does not like, or item i does not have attribute j that the user likes; otherwise, $y_{ij} = 0$. Then

$$d(\boldsymbol{v}^i,\boldsymbol{x}) = \sum_{j \in A} y_{ij}$$

As a result, the model has $O(nm)$ variables and constraints. Tests have shown that when the number n of attributes is large and users have evaluated a large number m of items, solving the MILP can take a few hours. We propose here a modified version of this MILP which has only $O(n+m)$ variables and constraints. The solution generated by the new model is therefore the same as that produced by the technique presented in [4], but the computing times are much shorter since with a few thousands attributes and a few hundreds of evaluated items, it only takes a few seconds to determine an estimate of a user profile. To do this, we define two vectors $\boldsymbol{x}^{like}$ and $\boldsymbol{x}^{dislike}$ in $\{0,1\}^n$ as follows :

$$x_j^{like} = \begin{cases} 1 & \text{if } x_i = 1 \\ 0 & \text{if } x_i = -1 \text{ or } 0, \end{cases}$$

$$x_j^{dislike} = \begin{cases} 1 & \text{if } x_i = -1 \\ 0 & \text{if } x_i = 0 \text{ or } 1. \end{cases}$$

Then $d(\boldsymbol{v}^i, \boldsymbol{x})$ is the number of components j with $v_j = 0$ and $x_j^{like} = 1$, or $v_j = 1$ and $x_j^{dislike} = 1$. Hence,

$$d'(\boldsymbol{v}^i, \boldsymbol{x}) = \sum_{j|v_j=0} x_j^{like} + \sum_{j|v_j=1} x_j^{dislike}.$$

So let $z_i = |d'(\boldsymbol{v}^i, \boldsymbol{x}) - \delta_i|$. We minimize $\sum_{i \in I'} z_i$ and impose

$$z_i \geq d'(\boldsymbol{v}^i, \boldsymbol{x}) - \delta_i \qquad \text{for all } i \in I' \tag{1}$$

$$z_i \geq \delta_i - d'(\boldsymbol{v}^i, \boldsymbol{x}) \qquad \text{for all } i \in I' \tag{2}$$

which is equivalent to impose

$$\sum_{j|v_j^i=0} x_j^{like} + \sum_{j|v_j^i=1} x_j^{dislike} - z_i \leq \delta_i \qquad \text{for all } i \in I' \tag{3}$$

$$\sum_{j|v_j^i=0} x_j^{like} + \sum_{j|v_j^i=1} x_j^{dislike} + z_i \geq \delta_i \qquad \text{for all } i \in I' \tag{4}$$

Vectors $\boldsymbol{x}^{like}$ and $\boldsymbol{x}^{dislike}$ can be defined with the following constraints:

$$x_j + 2x_j^{dislike} \leq 1 \qquad \text{for all } j \in A \tag{5}$$

$$x_j + x_j^{dislike} \geq 0 \qquad \text{for all } j \in A \tag{6}$$

$$x_j - x_j^{like} \leq 0 \qquad \text{for all } j \in A \tag{7}$$

$$x_j - 2x_j^{like} \geq -1 \qquad \text{for all } j \in A \tag{8}$$

$$x_j^{like}, x_j^{dislike} \in \{0,1\};\ x_j \in \{-1,0,1\} \qquad \text{for all } j \in A \tag{9}$$

In summary, by minimizing $\sum_{i \in I} z_i$ under constraints 3–9, we get a vector $\boldsymbol{x} = (x_1, \ldots, x_n)$ that corresponds to our estimate of the profile of user u based on the ratings that u has given to the m items in I'. This MILP has $3n$ integer variables, m continuous ones (the z_i's) and $4n+2m$ constraints.

4 Measures of relevance and similarity

In this section we define several measures. The first one is an estimation of the relevance of a recommendation for a user. The other measures estimate the similarity between a recommended item and those already rated by the considered user.

4.1 Relevance of a recommendation

In order to estimate the relevance of a recommendation $i \in I$ for a user u, we compute the distance $d(\boldsymbol{v}^i, \boldsymbol{x})$ between the vector associated with item i and our estimate $\boldsymbol{x}$ of the profile of user u.

While function τ translates an s-star rating in $\{1,\ldots,s\}$ into a d-rating in $[0,n]$, we consider the following inverse function $\tau^{-1}:[0,n]\rightarrow\{1,\ldots,s\}$:

$$\tau^{-1}(\delta)=s-\frac{\delta(s-1)}{n}.$$

For example, for $n=20$ and $s=5$, a d-rating $\delta=7$ is transformed into a 5-star rating of $\tau^{-1}(7)=5-\frac{28}{20}=3.6$.

The normalized expected rating, denoted e_i, that a user u will assign to a recommendation $i\in I$ is then defined as the estimated s-star rating of u for i divided by the number s of stars in the rating scale. More precisely,

$$e_i=\frac{1}{s}\tau^{-1}(d'(\boldsymbol{v}^i,\boldsymbol{x}))$$

where $\boldsymbol{x}$ is the estimate of the user profile produced by the MILP defined in Section 3.

4.2 Similarity measures between a recommended item and already rated ones

We define four measures to evaluate the similarity of a recommendation $i\in I$ with the items that a user u has already rated. The first one, denoted M_{min} is the normalized minimum Hamming distance between $\boldsymbol{v}^i$ and the vectors $\boldsymbol{v}^{i'}$ with $i'\in I'$, where $I'\subseteq I$ is the subset of items already rated by u. More precisely, we compute the minimum Hamming distance and divide it by the number of attributes for which the user seems to care about. For this purpose, we consider $N=\sum_{j=1}^{n}|x_j|$ (where $\boldsymbol{x}=(x_1,\ldots,x_n)$ is the estimate of the user profile produced by the MILP defined in Section 3) which is the number of attributes that the user seems to like or dislike, meaning that he cares about them. We then define

$$M_{min}(i)=\frac{1}{N}\min_{i'\in I'}d(\boldsymbol{v}^i,\boldsymbol{v}^{i'})=\frac{1}{N}\min_{i'\in I'}\sum_{j=1}^{n}|v^i_j-v^{i'}_j|.$$

The second measure, denoted M_{av} is the normalized average Hamming distance between $\boldsymbol{v}^i$ and the vectors $\boldsymbol{v}^{i'}$ with $i'\in I'$:

$$M_{av}(i)=\frac{1}{N|I'|}\sum_{i'\in I'}d(\boldsymbol{v}^i,\boldsymbol{v}^{i'})=\frac{1}{N|I'|}\sum_{i'\in I'}\sum_{j=1}^{n}|v^i_j-v^{i'}_j|.$$

The third and fourth measures are similar to the first ones except that the attributes are weighted according to their importance for the user. More precisely, to define the weight of attribute j, we compute the absolute value of the difference between the average rating that u has given to items having attribute j and the average rating that u has given to items not having attribute j. We thus get a number in $[0,s-1]$ which we divide by $s-1$ to get a normalized weight that is independent of the rating scale s. So let I''_j be the subset of items in I' having attribute j. Also, let r_i be the rating given by user u to $i\in I'$. The weight ω_j of attribute j is defined as

$$\omega_j=\frac{1}{s-1}\left|\frac{1}{|I''_j|}\sum_{i''\in I''_j}r_{i''}-\frac{1}{(|I'|-|I''_j|}\sum_{i''\in I'-I''_j}r_{i''}\right|$$

Measures $M^{weighted}_{min}$ and $M^{weighted}_{av}$ are then defined as follows:

$$M^{weghted}_{min}(i)=\frac{1}{N}\min_{i'\in I'}\sum_{j=1}^{n}\omega_j|v^i_j-v^{i'}_j|;$$

$$M^{weighted}_{av}(i)=\frac{1}{N|I'|}\sum_{i'\in I'}\sum_{j=1}^{n}\omega_j|v^i_j-v^{i'}_j|.$$

5 Computational experiment

We first describe the dataset we used in our experiments and then perform tests and analyze the results.

5.1 Description of the dataset used in our experiments

For our experiments, we use the MovieLens Serendipity 2018 database which was generated by Kotkov et al. [10]. It contains responses of 475 users of the the MovieLens website who agreed to answer questions $s_1, \ldots, s_6$ (see Section 2) regarding the serendipity of the recommendations they received. In total, there are 2146 items in the database for which we can estimate their serendipity for the concerned users. All movies in the database were rated by the users at most three months before they had to answer to questions $s_1, \ldots, s_6$. Moreover, these movies have received very few ratings from other users, which means that there are not popular items. Also, the movies that the users had to comment on are considered highly relevant for them since they got a rating of at least 7 stars on a scale of 10.

Due to some incomplete movie descriptions which do not specify some feature values, we used data from 444 of the 475 users, which gives a set D of 1797 movies for which, thanks to the answers to questions $s_1, \ldots, s_6$, we can measure the novelty and unexpectedness aspects of the recommendations offered to the users.

For each of the 444 users, we have extracted the set of movies they have rated in the past as well as the attribute values of all these movies. We have thus been able to use the MILP described in Section 3 to generate the profile of each of these users. The number of previously rated items varies from 52 to 6213, with an average of 606, depending of which user is considered. All ratings are on a 10-star scale.

Each movie is characterized by 1128 attributes which exhibit particular properties like movie's genre (action, crime, drama etc.), movie's theme (political corruption, midlife crisis, racism, memory loss etc.), awards given to the movie (Oscar awards in different categories, Saturn award for best special effects etc.), famous directors (Spielberg, Tarantino etc.), and other characteristics (true story, allegory, twist ending, thought provoking etc.) as well as viewers personal impressions (too long, unfunny, scary etc.). The dataset corresponding to a specific user is characterized by a subset of the full attribute list where, an attribute is included in the dataset only if it exists in at least one of the movies rated by the user. As a result, considering the 444 users, the number of attributes varies from 703 to 1124.

The answers given by the 444 users of the database to questions $s_1, \ldots, s_6$ are numbers in $\{0, 1, 2, 3, 4, 5\}$, where 5 corresponds to 'strongly agree', 4 to 'agree', 3 to 'neither agree nor disagree', 2 to 'disagree', 1 to 'strongly disagree', and 0 to 'don't remember'.

5.2 Experimental results

As mentioned in Section 2, serendipity can be defined in various ways. Kotkov et al. [10] suggest to combine novelty (s_1 or s_2) with unexpectedness (s_4, s_5 or s_6) and declare that s_3 should not be taken into account for defining serendipity. For our experiments, we have followed this recommendation and have thus considered the 32 possible combinations of s_1, s_2, s_4, s_5 and s_6. More precisely, given a subset c of $\{1, 2, 4, 5, 6\}$, we say that a recommendation for user u is *serendipitous according to* c if u has given an answer that belongs to $\{4, 5\}$ to all statements s_i with $i \in c$, which means that the user agrees or strongly agrees with these statements. Note that the experiements reported in [10, 11, 22] also consider that an answer to a statement in $\{s_1, s_2, s_4, s_5, s_6\}$ is positive if it belongs to $\{4, 5\}$.

Given $c \subseteq \{1, \ldots, 6\}$, we consider the subset D_c of recommendations in D which are serendipitous according to c for the concerned users. Note that $D_\emptyset = D$ since with $c = \emptyset$, there is no restriction for a recommendation to be serendipitous.

To every movie $i \in D$ we associate points $(e_i, m_i) \in \mathbb{R}^2$, where e_i is the normalized expected rating defined in Section 4.1 that the user will give to the recommended movie, while m_i is one of the four similarity measures defined in Section 4.2, i.e., $m_i = M_{min}(i)$, $M_{av}(i)$, $M_{min}^{weighted}(i)$ or $M_{av}^{weighted}(i)$.

Given a similarity measure in $\{M_{min}, M_{av}, M_{min}^{weighted}, M_{av}^{weighted}\}$ and a subset c of $\{1, \ldots, 6\}$, we thus get a set of points associated with the recommendations in D_c. For each such set of points, we remove outliers by only considering those points having a z-score smaller than 2, and we then determine a linear regression for the remaining points, which gives us a linear equation that indicates how the considered similarity measure for the points in D_c is linearly related to the normalized expected rating of these points.

For illustration, the linear regressions for $c = \emptyset$ (plain line) and $c = \{1, 4, 5, 6\}$ (dotted line) with $m_i = M_{min}(i)$ is depicted in Figure 1. We observe that for the points in D (i.e., with $c = \emptyset$), the minimum distance $M_{min}(i)$ from the recommended item i to the items that the user has already rated is approximately equal to $0.095e_i - 0.001$, while for the points in $D_{1,4,5,6}$, we have $M_{min}(i)$ being linearly related to the normalized expected rating as $0.183e_i - 0.049$. Hence, the slope of the linear regression is almost doubled for serendipitous items according to $\{1, 4, 5, 6\}$ when compared to the slope associated with all points in D. On a 10-star scale, this means that for recommending serendipitous items to a user, we should increase the minimum distance to the items rated in the past by about 2 additional percents for each additional star we expect from the user's rating. For example, while the minimum distance from a serendipitous recommendation to already rated items is approximately 8% of the number of attributes for an expected rating of 7 stars (i.e., $e_i = 0.7$), it raises to about 10% for an expected rating of 8 stars.

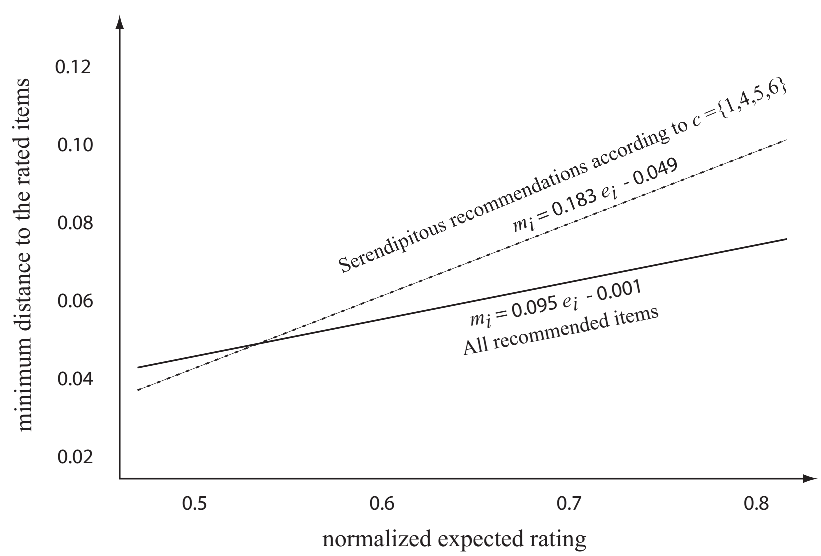

Figure 1: Linear regression that links the similarity measure $m_i = M_{min}(i)$ with the normalized expected rating e_i for $c = \emptyset$ (plain line) and $c = \{1, 4, 5, 6\}$ (dotted line).

We have determined these linear regressions for the 32 subsets of $\{1, 2, 4, 5, 6\}$. Wang and Chen [22] have suggested to combine s_4, s_5 and s_6 to define serendipity, while Kotkov et al. [10] suggest to include a novelty aspect. We show in Table 1 the slopes of the linear regressions for some definitions of serendipity, including those we just mentioned. More precisely, we consider every singleton $\{s_i\}$, all pairs of statements chosen in $\{s_4, s_5, s_6\}$, the set $\{s_4, s_5, s_6\}$ itself (as suggested in [22]), the latter with s_1 or s_2 (as suggested in [10]), and $\{s_1, s_2, s_4, s_5, s_6\}$. We also indicate the slope obtained with $c = \emptyset$. We observe that the biggest difference when compared singletons to $c = \emptyset$ is obtained with $c = \{4\}$ which indicates that statements s_4 seems to be the most influencial for serendipity. But the combination given by $c = \{1, 4, 5, 6\}$ is even more influential, whatever the considered similarity measure.

Table 1: Slopes of the linear regressions that indicate how similarity measures are linked to the normalized expected rating, for various definitions of serendipity.

	Slopes			
	Unweighted distance		Weighted distance	
c	M_{min}	M_{av}	$M_{min}^{weighted}$	$M_{av}^{weighted}$
$\{1\}$	0.106	0.152	0.001	0.001
$\{2\}$	0.104	0.147	0.002	0.001
$\{4\}$	0.131	0.167	0.001	-0.005
$\{5\}$	0.108	0.151	0.000	-0.006
$\{6\}$	0.113	0.152	0.000	-0.005
$\{4,5\}$	0.127	0.154	0.001	-0.004
$\{4,6\}$	0.148	0.174	0.000	-0.006
$\{5,6\}$	0.122	0.160	0.000	-0.004
$\{4,5,6\}$	0.157	0.177	0.001	-0.003
$\{1,4,5,6\}$	0.183	0.208	-0.009	-0.009
$\{2,4,5,6\}$	0.161	0.184	0.002	0.006
$\{1,2,4,5,6\}$	0.178	0.204	-0.006	-0.002
$\emptyset$	0.095	0.143	0.001	-0.004

Interestingly, the slope for $c = \{1,4,5,6\}$ and $m_i = M_{min}^{weighted}$ (as well as other slopes with $M_{av}^{weighted}$) has a negative value. The linear regression of this case is shown in Figure 2. We observe that the points in $D_{\{1,4,5,6\}}$ have $M_{min}^{weighted}(i)$ approximately equal to $-0.009e_i + 0.010$. This means that if we want to recommend serendipitous items to users, the attributes of the recommended items that are important for them (i.e., they have a large weight) must fit their preferences, and this is all the more true if the recommendation is relevant to them.

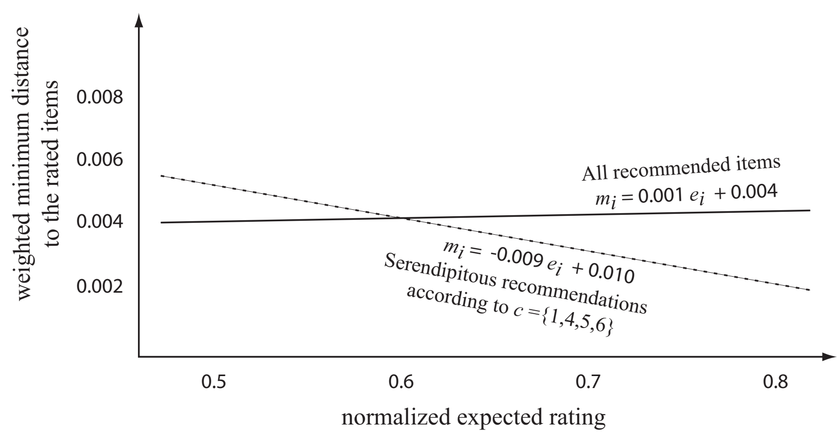

Figure 2: Linear regression that links the similarity measure $m_i = M_{min}^{weighted}(i)$ with the normalized expected rating e_i for $c = \emptyset$ (plain line) and $c = \{1,4,5,6\}$ (dotted line).

Note that the normalized expected rating shown in Figures 1 and 2 are relatively high, larger than 0.45, which means that we expect the user to rate the recommendations with at least 4.5 stars on a 10-star scale. This is due to the fact that the dataset D collected by Kotkov et al. [10] is skewed so that it contains movies for which the users have assigned a rating of at least 7 on a 10-star scale.

To summarize our findings, we have observed that the largest absolute value of a slope of the linear regression that links the normalized expected rating with a measure of the distance from the recommended item to the items that the user has rated in the past is obtained by defining serendipity with $c = \{1,4,5,6\}$. In other words, a recommendation is considered as serendipitous if the user has answered 'agree' or 'strongly agree' to statement s_1 (which concerns novelty) as well as to statements s_4, s_5, s_6 (which concern unexpectedness). With such a definition, it seems that for a recommendation to be considered serendipitous to a user, the distance between the recommended item and those that the user has rated in the past must increase with the relevance of the recommendation. In other words, to recommend serendipitous items, we should not hesitate to deviate from what a user has seen in the past, and this is all the more true if we expect a high relevance of the recommended item.

We have also observed that for a recommendation to be considered as serendipitous, the number of attribute values that do not correspond to the user profile can amount to 12% of the total number of attributes when the normalized expected rating is at least 0.8 (i.e., 8 stars on a 10-star scale) and should not be less than 4%, assuming that the system does not recommend items with normalized expected rating lower than 0.5. In addition, the attribute values of the user profile that can be modified to create unexpectedness should be those which are not too important for the user, where the importance of an attribute j is measured with weight ω_j. In other words, deviating from the user profile is more dangerous for attributes with large weights, which means that an item has a better chance of being considered serendipitous when its predicted rating is medium-high and its attributes with the largest weights match the user profile. Also, the higher the expected rating of the recommendation, the more we are allowed to deviate from the user profile.

6 Conclusions, limitations and future work

In this study we showed how we can assess whether an item has a high probability of being considered serendipitous based only on the previous ratings of the user. Our results show that the unexpectedness and novelty components that best reflect serendipity of items are statements s_1, s_4, s_5 and s_6. Following Kotkov et al. [10] who found that high expected ratings can be a predictor for serendipity, we showed that the more a serendipitous recommendation seems to be relevant to a user (i.e., it has a high normalized expected rating), the greater its distance from the items previously rated by the user can be, provided that its important features are preserved.

Several measures can be used in order to predict whether an item has a high probability of being serendipitous for a user. These measures are:

1. The distance between the recommended item i and the user profile: this measure reflects the normalized expecting rating e_i that the user will assign to i. To have a higher probability of being serendipitous, e_i should not be smaller than 0.5.
2. The distance between the recommended item and the items previously rated by the user. This distance should not be too small (because items with similar attributes values as those of already rated items are too obvious for the user) neither too large (because items that are very different from what the user was exposed to in the past, will probably not match the user preferences).

Based on these two measures we can estimate the likelihood of a certain item to be serendipitous. The weights of the attributes also play an important role. Changing an attribute value to create unexpectedness is more dangerous for attributes with large weights. Thus, an item has a better chance of being serendipitous for a user when its attributes with the largest weights have the same value as in the user profile while the attributes with the smallest weights are allowed to be different from what the user was exposed to in the past.

A limitation of the research described in this paper is that the data for the experiments was extracted from the movie domain only. Although the movie domain is commonly used for evaluating recommender systems algorithms [24], we would like to expand the experiments to other domains.

Preserving user privacy is also a challenge that needs to be addressed since datasets may include sensitive and vulnerable information [5]. The proposed recommendation method allows preserving the confidential data of the users because the information necessary to estimate the preferences of each user can be kept locally and does not require the knowledge of the ratings of other users. In future research we intend to implement our recommendation technique on a laptop or a smartphone while using a mechanism for preserving user privacy, as proposed in [13].

References

[1] Panagiotis Adamopoulos and Alexander Tuzhilin. On unexpectedness in recommender systems: Or how to better expect the unexpected. ACM Trans. Intell. Syst. Technol., 5(4), dec 2014.

[2] Marco de Gemmis, Pasquale Lops, Giovanni Semeraro, and Cataldo Musto. An investigation on the serendipity problem in recommender systems. Information Processing & Management, 51(5):695–717, 2015.

[3] Mouzhi Ge, Carla Delgado-Battenfeld, and Dietmar Jannach. Beyond accuracy: Evaluating recommender systems by coverage and serendipity. In Proceedings of the Fourth ACM Conference on Recommender Systems, RecSys '10, pages 257–260, New York, NY, USA, 2010. Association for Computing Machinery.

[4] Alain Hertz, Tsvi Kuflik, and Noa Tuval. Resolving sets and integer programs for recommender systems. Journal of Global Optimization, 81:1–26, 09 2021.

[5] Yassine Himeur, Shahab Saquib Sohail, Faycal Bensaali, Abbes Amira, and Mamoun Alazab. Latest trends of security and privacy in recommender systems: A comprehensive review and future perspectives. Computers & Security, 118:102746, 2022.

[6] Leo Iaquinta, Marco de Gemmis, Pasquale Lops, Giovanni Semeraro, Michele Filannino, and Piero Molino. Introducing serendipity in a content-based recommender system. In 2008 Eighth International Conference on Hybrid Intelligent Systems, pages 168–173, 2008.

[7] M. Jenders, T. Lindhauer, G. Kasneci, R. Krestel, and F. Naumann. A serendipity model for news recommendation. In KI 2015: Advances in Artificial Intelligence, pages 111–123, Cham, 2015. Springer International Publishing.

[8] Marius Kaminskas and Derek Bridge. Diversity, serendipity, novelty, and coverage: A survey and empirical analysis of beyond-accuracy objectives in recommender systems. ACM Trans. Interact. Intell. Syst., 7(1), dec 2016.

[9] Naoki Kito, Kenta Oku, and Kyoji Kawagoe. Correlation analysis among the metadata-based similarity, acoustic-based distance, and serendipity of music. In Proceedings of the 19th International Database Engineering & Applications Symposium, IDEAS '15, pages 198–199, New York, NY, USA, 2015. Association for Computing Machinery.

[10] Denis Kotkov, Joseph A. Konstan, Qian Zhao, and Jari Veijalainen. Investigating serendipity in recommender systems based on real user feedback. In Proceedings of the 33rd Annual ACM Symposium on Applied Computing, SAC '18, pages 1341–1350, New York, NY, USA, 2018. Association for Computing Machinery.

[11] Denis Kotkov, Jari Veijalainen, and Shuaiqiang Wang. How does serendipity affect diversity in recommender systems? a serendipity-oriented greedy algorithm. Computing, 102(2):393–411, feb 2020.

[12] Denis Kotkov, Shuaiqiang Wang, and Jari Veijalainen. A survey of serendipity in recommender systems. Knowledge-Based Systems, 111:180–192, 2016.

[13] Tsvi Kuflik and Katerina Poteriaykina. User model on a key. In Proceedings of the 20th ACM Conference on Hypertext and Hypermedia, HT '09, pages 371–372, 2009.

[14] Pasquale Lops, Marco de Gemmis, and Giovanni Semeraro. Content-based Recommender Systems: State of the Art and Trends, pages 73–105. Springer US, Boston, MA, 2011.

[15] Valentina Maccatrozzo, Manon Terstall, Lora Aroyo, and Guus Schreiber. Sirup: Serendipity in recommendations via user perceptions. In Proceedings of the 22nd International Conference on Intelligent User Interfaces, IUI '17, pages 35–44, New York, NY, USA, 2017. Association for Computing Machinery.

[16] Andrii Maksai, Florent Garcin, and Boi Faltings. Predicting online performance of news recommender systems through richer evaluation metrics. In Proceedings of the 9th ACM Conference on Recommender Systems, RecSys '15, pages 179–186, New York, NY, USA, 2015. Association for Computing Machinery.

[17] Sean M. McNee, John Riedl, and Joseph A. Konstan. Being accurate is not enough: How accuracy metrics have hurt recommender systems. In CHI '06 Extended Abstracts on Human Factors in Computing Systems, CHI EA '06, pages 1097–1101, New York, NY, USA, 2006. Association for Computing Machinery.

[18] Cataldo Musto, Marco de Gemmis, Pasquale Lops, Fedelucio Narducci, and Giovanni Semeraro. Semantics and Content-Based Recommendations, pages 251–298. Springer US, New York, NY, 2022.

[19] Francesco Ricci, Lior Rokach, and Bracha Shapira. Recommender Systems: Techniques, Applications, and Challenges, pages 1–35. Springer US, New York, NY, 2022.

[20] Nur Izyan Yasmin Saat, Shahrul Azman Mohd Noah, and Masnizah Mohd. Towards serendipity for content-based recommender systems. International Journal on Advanced Science, Engineering and Information Technology, 8(4-2):1762–1769, 2018.

[21] Saúl Vargas and Pablo Castells. Rank and relevance in novelty and diversity metrics for recommender systems. In Proceedings of the Fifth ACM Conference on Recommender Systems, RecSys '11, pages 109–116, New York, NY, USA, 2011. Association for Computing Machinery.

[22] Ningxia Wang and Li Chen. How do item features and user characteristics affect users' perceptions of recommendation serendipity? a cross-domain analysis. User Modeling and User-Adapted Interaction, 33:1–39, 12 2022.

[23] Qianru Zheng, Chi-Kong Chan, and Horace H. S. Ip. An unexpectedness-augmented utility model for making serendipitous recommendation. In Petra Perner, editor, Advances in Data Mining: Applications and Theoretical Aspects, pages 216–230, Cham, 2015. Springer International Publishing.

[24] Reza Ziarani and Reza Ravanmehr. Serendipity in recommender systems: A systematic literature review. Journal of Computer Science and Technology, 36:375–396, 04 2021.

[25] Cai-Nicolas Ziegler, Sean M. McNee, Joseph A. Konstan, and Georg Lausen. Improving recommendation lists through topic diversification. WWW '05, pages 22–32, New York, NY, USA, 2005. Association for Computing Machinery.