www.editada.org

International Journal of Combinatorial Optimization Problems and Informatics, 16(4), Sep-Dec 2025, 233-249. ISSN: 2007-1558.
https://doi.org/10.61467/2007.1558.2025.v16i4.937

Using the Zipf Distribution to Mitigate the Matthew Effect and Improve Fairness in Bias-SVD Algorithm Recommendations

Cupertino Lucero Álvarez [1,2], *Perfecto Malaquías Quintero Flores* [2], *David Eduardo Pinto Avendaño* [3], *Brian Manuel González Contreras* [2], *María del Rocío Ochoa Montiel* [2], *Ever Juárez Guerra* [2]
[1] Universidad Tecnológica de Izúcar de Matamoros, Tecnologías de la Información
[2] Universidad Autónoma de Tlaxcala, Facultad de Ciencias Básicas Ingeniería y Tecnología
[3] Benemérita Universidad Autónoma de Puebla, Facultad de Ciencias de la Computación
clucero@utim.edu.mx, perfectomalaquias.quintero.f@uatx.mx, dpinto@cs.buap.mx, brianmanuel.gonzalez@uatx.mx, mariadelrocio.ochoa.m@uatx.mx, ever.juarez.g@uatx.mx

Abstract. Recommender Systems are useful tools for helping users find items of interest within a universe of options in the Big Data 3.0 era. Singular Value Decomposition models have proven useful in e-commerce. However, these models do not consider popularity biases arising from the Matthew effect present in the data structure, which leads to unfair recommendations. To address this problem, strategies that compensate for long-tail items to increase their recommendation probability have been proposed, as well as approaches that use the Zipf distribution to generate predictions without prior knowledge of the data. However, these proposals have not been widely accepted because they do not consider user-item interactions in the training process. In this paper, we present a strategy that uses the Zipf distribution in a Matrix Factorization model based on Singular Value Decomposition that considers user and item biases in personalized recommendation tasks, to incorporate popularity biases and improve the fairness of recommendations. Experimental results demonstrate the validity of this strategy.
Keywords: Recommender Systems, Matthew Effect, Fairness.

Article Info
Received February 25, 2025
Accepted July 6, 2025

1 Introduction

In this era of Big Data 3.0, where digital information is generated from multiple sources within e-commerce, social networks, and IoT technologies, it is increasingly important to create computer tools that help to lighten the large volumes of information that are generated. Users need valuable information from this large amount of data, so building computational mechanisms through which to achieve this goal is a real challenge (Lee, 2017). In this context, Recommender Systems (RS) aim to provide valuable recommendations to users by suggesting items that best match their profiles. RS of Collaborative Filtering (CF) have been widely used in the e-commerce industry; in this RS the list of recommendations generated depends on the ratings that neighbors have given to items that the active user is unaware of (Lucero-Alvarez et al., 2021). Because the possibilities for user-item interactions are so large, many RS use Matrix Factorization (MF) techniques to reduce the dimensions of the data. Singular Value Decomposition (SVD) is an MF method in which predictions are made by taking the dot product of the vectors representing users and items. Driven by the Netflix prize in 2009, CF-based RS began their golden age, giving rise to classic models such as Regularized-SVD, Bias-SVD, SVD++ and their variants. These models basically follow the original proposal of Simon Funk to approximate the ratings matrix M_r by the product of two matrices $U \cdot V^T$ in such a way that the dimensions of the original system are reduced (Koren, 2008; Koren et al., 2009; Sharifi et al., 2013; Gower, 2014; Paterek, 2007; Kumar et al., 2014; Lucero-Alvarez et al., 2023). This approach is based on the mathematical idea that $M_r = U \Sigma V^T$ where Σ is considered multiplied proportionally in both U and V. The idea is to start U and V with random values and approximate M_r at each training stage by minimizing the Mean Square Error (MSE) using Stochastic Gradient Descent (SGD). The Bias-SVD model has gained acceptance due to its ease of implementation and the accuracy shown. Bias-SVD variants have to do with the incorporation of implicit feedback and with the way of dealing with different biases in the training stage, such as biases referring to the deviation of each user-item interaction with respect to the global mean, biases due to context, biases due to temporality, and demographic biases (Koren, 2008; Koren et al., 2009; Chen et al., 2023). However, these models and variants do not guarantee fairness in their recommendations, because they were designed with the focus on minimizing the error in the loss function, and consequently they recommend popular items, leaving aside unpopular items that are also relevant to the active user. These models do not address popularity biases since they do not consider during training that the original structure of the data is biased towards elements that have historically received greater exposure and coverage, which is why they end up recommending biased elements, overexposing them more, and at the same time generating greater bias. This phenomenon is known as the snowball effect, preferential attachment, or Matthew effect (Perc, 2014). However, researchers know that recommending popular items is contrary to the fundamental goal of RS, which is to recommend based on user profiles. To address this lack of fairness, researchers have proposed graph-based models, such as in the work of Wei et al. (2021), where popularity biases are explored using a strategy that models a causal graph to describe cause-effect relationships in the recommendation process. The objective was to investigate how the popularity of each element affects each interaction through its cause-effect model, with encouraging results. Along the same line, cause-effect approaches have been proposed that seek to counteract the importance of items with overexposure biases by increasing the recommendation probability of items in the long tail of the distribution in the training process and that make use of implicit information from the interaction of users with the system to estimate unknown ratings on rarely evaluated elements (Liang et al., 2016; Wang et al., 2019). Although these methods work, they do not explain how the biases of each rating interact with respect to compensation; that is, they blindly privilege the elements of the long tail without considering whether they are within the user's profile. Other approaches focused on user-item interaction use neural networks, with good results, but they also do not explain how popularity biases affect interactions (He et al., 2017; Zheng et al., 2016). Approaches that make use of Zipf's law have also been proposed: Models such as Zipf Matrix Factorization (ZMF) and ZeroMat are among the best known; they use Zipf's law without historical neighborhood information to make predictions, but they have proven to be effective in reducing the Matthew effect and improving equity (Wang, 2021a; Wang, 2021b).

This article presents research inspired by previous work in Information Retrieval (IR) within Natural Language Processing (NLP), which were focused on the automatic indexing of documents using Goffman's Transition Point (TP) and Zipf's law. The objective was to explore whether, by incorporating popularity biases derived from Zipf's law, it is possible to provide classical SVD models with a mechanism that improves the fairness of recommendations, and if so, propose variants that address the problem of inequity while maintaining the benefits of the base framework. For this purpose, three variants are proposed, and a comparison is made with the Bias-SVD model. Two MovieLens datasets with different distributions are used in the experiments, and they are often used as the gold standard of evaluation in many investigations in the area, namely MovieLens Small and MovieLens 100K (Harper & Constan, 2016).

2 Related Work

This section presents some work related to our research. Sections 2.1 and 2.2 introduce the concepts of Zipf's law and the Goffman transition point in NLP. Section 2.3 presents research in RS that uses Zipf's law in their learning models to address problems such as data sparsity, cold start, and recommendation unfairness. Section 2.4 presents some proposed strategies to address the problem of unfairness resulting from popularity biases in RS. Finally, in Section 2.5, we present the significance of our proposal with respect to the others.

2.1 Zipf Law

Zipf's law has its origins in linguistics and is a discrete power distribution that many phenomena approximate. Zipf's law was published by the American linguist and philosopher George Kingsley Zipf in 1949 in his famous book "Human behavior and the principle of least effort" (Zipf, 2016; Zhu et al., 2018). Natural or artificial phenomena that approximate this type of distribution show that there are few elements that have high frequencies and many elements that have low frequencies. This was also observed by Pareto in his studies on the population of cities and is also known as the 80-20 proportion or "Pareto principle" (Newman, 2005).

Zipf empirically finds that the frequency with which words appear in the documents of any corpus follows a distribution $f(k) \propto 1/k^{\alpha}$, where the k-th most frequent word has a frequency $f(k)$, for $\alpha \approx 1$. So, k is the frequency rank of a word, and $f(k)$ is its frequency in the corpus, thus, the most frequent word, with rank $k = 1$ has a frequency proportional to 1, the second most frequent word, $k = 2$ has a frequency proportional to $1/2^{\alpha}$, that is, it is repeated with a frequency of $1/2$ of the frequency of the first rank, the third most frequent word has a frequency proportional to $1/3^{\alpha}$, that is, it is repeated with a frequency of $1/3$ that of the first, and so on (Piantadosi, 2014).

Looking at the list of all the words in the corpus, sorted in descending order of frequency, Zipf was able to establish that the product of the rank by the frequency remains approximately constant (Booth, 1967). This statistical regularity comes from the tension between two forces inherent to natural languages: unification and diversification. The first leads to the use of general terms, while the second leads to the use of specific terms (Jiménez et al., 2005). This relationship can be approximated according to equation 1.

$$f(k) \approx \frac{C}{k^{\alpha}} \quad \textbf{(1)}$$

Where, α takes a value slightly higher than 1, and C is the normalization constant (Montemurro, 2001). The value of α is obtained empirically, while C refers to the highest frequency of elements ordered in descending order in relation to it.

2.2 Transition Point

In IR of NLP, specifically for the automatic identification of index terms that best represent the documents, researchers use Zipf's law and Goffman's TP (Booth, 1967; Urbizagástegui & Restrepo, 2011). They noticed that the most frequent words contribute little to the representation and that the low-frequency words add complexity. They propose dividing the list of words sorted descending in relation to their frequencies into high and low-frequency words so that terms in a window around the TP are chosen as index terms. This technique has proven useful in languages, since they, in most cases, follow Zipf's law. In the work of (Jiménez et al., 2005) this technique is used together with the Nearest Neighbor (NN) algorithm to group texts from a specific domain, with good results according to the evaluation standard used. In (Moyotl & Macías, 2016) TP, Markov chains, and n-grams are used to determine terms and auto-complete queries semi-automatically and assist in IR processes.

The transition point relates Zipf's high and low-frequency laws from equations 2 and 3. In these equations, N represents the number of words present in the text, including repetitions, and k is a constant associated with the text length and frequency rank.

$$I_1 = \frac{1}{2}kN \quad \textbf{(2)}$$

$$I_n = \frac{1}{n(n+1)}kN \quad \textbf{(3)}$$

In such a way that the quotient is as in equation 4.

$$\frac{I_n}{I_1} = \frac{2}{n(n+1)} \quad \textbf{(4)}$$

Where I_n refers to the number of words with a frequency of n, and I_1 is the number of words with a frequency of 1.

Since we are interested in the average frequency terms, we set $I_n = 1$ and therefore $2I_1 = n(n+1)$, so we have $n^2 + n - 2I_1 = 0$, so by solving for n we can determine the transition point, as in equation 5 (Booth, 1967).

$$TP = \frac{\sqrt{1 + 8 \times I_1} - 1}{2} \quad \textbf{(5)}$$

The concept of transition point is useful today in NLP-related research. For example, it is often used to disambiguate the meaning of words, to analyze the context in which language is used, for sentiment analysis, and for automatic text generation.

In the context of this research, the concept of transition point is used to identify popularity biases present in the structure of the training data of the SVD models under study.

In sections 2.3 and 2.4 we present some works from the literature that address RS problems using Zipf distribution and some approaches that address the lack of fairness of recommendations from different perspectives.

2.3 Frameworks that use Zipf's Law in RS

CF-based RS have been widely studied. However, problems such as cold-start, sparsity, and lack of fairness persist despite the efforts of researchers to find appropriate strategies to solve them. Some current work seeking to solve these problems make use of Zipf's law. A pioneering work in the use of the Zipf's distribution to address the problem of unfairness in recommendations is the so-called Zipf Matrix Factorization, in which the author uses a metric called Degree of Matthew Effect to measure the Matthew effect and to be able to penalize it in the loss function based on the scalar product of U by V^T (Wang, 2021b). The author compares his method against the classic Vanilla Matrix Factorization on a MovieLens dataset and obtains a better performance with respect to Mean Absolute Error (MAE). Another work that makes use of the Zipf's distribution to address sparsity and cold-start problems is called ZeroMat (Wang, 2021a). This algorithm does not use explicit feedback, nor information from other domains or meta-learning, for making predictions, however; the author reports good results comparing it with classical Matrix Factorization. Later, in Wang (2022a) the framework called Hybrid-ZeroMat is proposed, in which the author uses the ZeroMat algorithm to fill the unknown ratings of the ratings matrix, in a kind of imputation, and then applies another RS such as Matrix Factorization, to solve the cold-start problem. Finally, in Wang (2022b) a framework called DotMat is presented, in which the cold-start and sparsity problems are attacked without having historical information of the ratings in the learning process, instead it uses the Zipf's distribution, and the inner product of the approximation vectors U and V^T, the loss function is based on the MAE evaluation metric using SGD, and the Degree of Matthew Effect. The authors show competitive results with respect to other frameworks based on Zipf's distribution, and with respect to classical MF algorithms.

2.4 Approaches to Addressing Unfairness in RS

To address the problem of unfairness due to popularity biases in collaborative filtering recommendation models, researchers have proposed various strategies, such as incorporating fairness constraints during the training phase, such as the (Zhu et al., 2018). The authors propose a sensitive latent factor matrix to isolate sensitive features and provide fairer recommendations with respect to the sensitive attribute. In this research, the authors use the MovieLens 100K dataset generated through explicit feedback in their experiments and adjust the ratings to values between 0 and 1, as their strategy is based on implicit feedback. Another approach that addresses unfairness is to incorporate regularization terms into the loss function to penalize unfair recommendations, as in the work of (Wang, 2021b), which uses a popularity bias penalty constant in the model training process. However, one of the weaknesses of this approach is that it does not use information on how users rate items or how each item has been rated. By not incorporating information on user preferences and behaviors, personalizing recommendations becomes difficult. Another approach to improving the fairness of recommendations involves the use of debiasing techniques. In this regard, (Sun et al., 2019) published a novel work in which the authors propose to eliminate popularity biases in the feedback loop that occurs with user interaction with the Recommender System, which is the cause of increased biases in SVD-based RS. The strategy relies on several debiasing algorithms in the simulated chain of events of a Recommender System. The authors demonstrate that it is possible to reduce biases with their strategy, but they acknowledge some weaknesses in the simulation of interactions. Another approach that addresses the problem of unfairness due to popularity biases is based on the use of counterfactual reasoning, as in the work of (Wei et al., 2021). In this work, the authors propose the use of causal graphs that capture cause-effect interactions in the recommendation process and demonstrate that their approach improves fairness.

2.5 Contributions

Our approach seeks to leverage the advantages of classic SVD models, such as ease of implementation, simplicity, high accuracy, and the ability to learn user-item interaction patterns that enable personalized recommendations, while simultaneously improving the fairness of recommendations.

The contributions of this work are as follows:

- A strategy for measuring popularity biases that uses the Zipf distribution and Goffman's TP.
- Three variants of the Bias-SVD model that incorporate popularity biases in the training phase, seeking to improve the fairness of recommendations without significantly losing accuracy.

3 SVD Models Implemented

This section describes the Regularized-SVD and Bias-SVD models implemented in the experiments. Both factorize the rating matrix as the product of two lower-dimensional matrices: $M_r = U \cdot V^T$.

3.1 Regularized-SVD Model

In this model, taken from (Koren et al., 2009; Funk Netflix Update, 2006), the learning of p_u and q_i is achieved by minimizing the regularized quadratic error, as in equation 6, considering that $p_u \in U$ and $q_i \in V$.

$$\min_{q*,p*} \sum_{(u,i)\in K} \left(r_{u,i} - q_i^t \cdot p_u\right)^2 + \lambda(\| q_i \|^2 + \| p_u \|^2) \tag{6}$$

Where K is the set of pairs (u, i) for which $r_{u,i}$ is known, the constant λ controls the degree of regularization and is usually determined by cross-validation (Koren et al., 2009). To minimize equation 6, an optimizer such as SGD is used. The algorithm goes through all the ratings in the training set, calculating in each case the associated prediction error $e_{u,i}$, as shown in equation 7.

$$e_{u,i} = r_{u,i} - q_i^t \cdot p_u \tag{7}$$

The parameters are then changed in a magnitude proportional to γ in the opposite direction to the gradient ∇, i.e.: $q_i \leftarrow q_i - \gamma \nabla$ and $p_u \leftarrow p_u - \gamma \nabla$. As in equations 8 and 9.

$$q_i \leftarrow q_i + \gamma \cdot \left(e_{u,i} \cdot p_u - \lambda \cdot q_i\right) \tag{8}$$

$$p_u \leftarrow p_u + \gamma \cdot \left(e_{u,i} \cdot q_i - \lambda \cdot p_u\right) \tag{9}$$

3.2 Bias-SVD Model

This model considers biases related to the deviation of each rating from the user and active item averages and is compared to the global average (Koren et al., 2009). Therefore, $b_{u,i} = \mu + b_i + b_u$, where μ is the global average in M_r, and the parameters b_i and b_u are the observed deviations of the user u and element i respectively. Therefore, to estimate the rating of the user u for the element i, we have the following: $\hat{r}_{u,i} = \mu + b_i + b_u + q_i^t \cdot p_u$. The learning process is carried out by minimizing the function of equation 10.

$$\min_{q^*,p^*,b^*} \sum_{(u,i)\in K} \left(r_{u,i} - \mu - b_i - b_u - q_i^t \cdot p_u\right)^2 + \lambda(\| q_i \|^2 + \| p_u \|^2 + b_u^2 + b_i^2) \tag{10}$$

Minimization is done by equations 11, 12, 13 and 14.

$$b_u \leftarrow b_u + \gamma \cdot \left(e_{u,i} - \lambda b_u\right) \tag{11}$$

$$b_i \leftarrow b_i + \gamma \cdot \left(e_{u,i} - \lambda b_i\right) \tag{12}$$

$$p_u \leftarrow p_u + \gamma \cdot \left(e_{u,i} \cdot q_i - \lambda p_u\right) \tag{13}$$

$$q_i \leftarrow q_i + \gamma \cdot \left(e_{u,i} \cdot p_u - \lambda q_i\right) \tag{14}$$

Where $e_{u,i} = r_{u,i} - \hat{r}_{u,i}$.

4 Methodology

In this section we present the methodology proposed to measure the popularity biases present in the data.

4.1 Identification of the Ranks

The proposal for calculating the popularity biases of each element of the active dataset involves the rank in which each element is located. The identification of the ranks is posed as follows: Let $L = [e_1, e_2, \ldots, e_N]$ be the list of the N elements of the dataset ordered in descending order with respect to their frequencies. Let $Fr = [fr(e_1), fr(e_2), \ldots, fr(e_N)]$ be the list of frequencies of each element in L. In such a way that $D = \{(e_i, fr(e_i)) | e_i \in L \wedge fr(e_i) \in Fr\}$ is the element-frequency representation of the dataset. The rank of e_1 is $k = 1$, the rank of e_2 is $k = 2$ if $fr(e_1) > fr(e_2)$, otherwise, both e_1 and e_2 will have rank $k = 1$. The rank of e_3 is the rank of $e_2 + 1$ if $fr(e_2) > fr(e_3)$, otherwise, the rank of e_3 will be the same as the rank of e_2, and so on, up to the rank of e_N, with $k \leq N$. That is, if two or more elements have the same frequency, then they correspond to the same rank. Therefore, let $K = \{k_1, k_2, \cdots, k_n\}$ be the set of n ranks of the dataset.

4.2 Reducing the Matthew effect

Analogous to the automatic indexing of terms using Zipf's law and TP, in this work, we propose the use of P_{Matt} as the transition point that divides L into two sets of items, those with popularity biases and those without such bias. The hypothesis is that if P_{Matt} it can be identified with sufficient precision, it is possible to generate a mechanism to penalize popularity biases in the training stage of the SVD model. However, in this context, using equation 5 to determine the TP is not always possible because the datasets do not always follow the Pareto proportion. But, in Zipfian datasets like MovieLens Small, equation 5 can be used to determine the Goffman TP. For which it is only required to identify in the frequency list Fr the number of elements that have a frequency of one (I_1), and apply the formula. Another way to calculate the TP is to identify the lowest frequency that does not repeat; either way, in both methods you just must go through the list, so the order of complexity is $\Theta(N)$ (Pinto et al., 2006).

For our research we propose the identification of P_{Matt} as follows:

- Determine a Q point as the last frequency that does not repeat until before 20% of the ratings of the elements in D.
- Determine a point P as the median frequency of the distribution of the elements in D.
- Determine P_{Matt} as the midpoint frequency between the positions of Q and P, as in the equation 15.

$$P_{Matt} = frec\left(\frac{pos(P) - pos(Q)}{2}\right) \quad \textbf{(15)}$$

Once determined Q, P, and P_{Matt}, we define the window of elements of the equation 16, whose ratings will be used to calculate a local average that we call μ_{QP}.

$$V_{QP} = \{(e_i, fr(e_i)) | (e_i, fr(e_i)) \in D \wedge e_i \in L \wedge Q \geq fr(e_i) \geq P\} \quad \textbf{(16)}$$

Just as in automatic document indexing, where window terms are considered the most representative, in the movie domain, we believe that items found in the window are the least affected by popularity bias and lack of coverage, and therefore should be the most stable, whose average of their ratings (μ_{QP}) should better represent the equilibrium point of the distribution of grades, than the global average (μ), when the variance is large.

In this sense we use the Zipf distribution to determine the popularity biases and with them penalize the elements that are in the range $[1 : \eta]$, where $\eta \in L$ represents the position in L of the first element whose frequency is equal to P_{Matt}. To determine the popularity bias of each rank, you need to know the value of the power α of the Zipf distribution. As in Wei et al. (2021) it is calculated α with a small adjustment as in equation 17. The detailed statistical theory of the formula can be found in (Barabási, 2013).

$$\alpha = 1 + n_k \left(\sum_{i=1}^{n} ln \frac{x_i}{x_{max}}\right)^{-1} \quad \textbf{(17)}$$

Where n is the total number of elements, n_k is the number of ranks of the elements, x_i the popularity rank of the i-th element, x_{max} is the longest rank of the elements. Thus, the Zipf probability distribution is presented in equation 18.

$$f(k; \alpha, \eta) = \frac{1/k^{\alpha}}{\sum_{i=1}^{\eta}(1/i^{\alpha})} \quad \textbf{(18)}$$

In our case k is any rank from 1 up to the range where P_{Matt} is located.

Using equation 18, we find the popularity bias vector of each element relative to the corresponding rank k_{e_i}. For example, to determine the popularity bias of the item e_i, we have: $Me_i = f(k_{e_i}; \alpha, \eta)$. Where Me_i is the popularity bias of the item e_i, k_{e_i} refers to the rank of the element e_i, and $e_i \in [1 : \eta]$, so: $VM = [Me_1, Me_2, \ldots, Me_\eta]$ is the popularity bias vector.

Finally, we need a penalty constant β to adjust popularity biases based on the data structure of the dataset used. Therefore, VM remains as in equation 19.

$$VM = [Me_1, Me_2, \ldots, Me_\eta] \times \beta \tag{19}$$

Therefore, the popularity bias of item e_i is calculated as: $Me_i \times \beta$, for simplicity we denote it as Ψe_i.

4.3 Proposed variants of the Bias-SVD model

In this section we present three variants that we propose in the Bias-SVD model.

4.3.1 VZipfMatt_penalized(μ−)

In this variant, the Matthew effect is penalized with respect to the global average μ. The idea is that by penalizing popularity biases, more equitable recommendations are generated, considering the user-element interactions of each profile, for which: To predict the rating that user u would give the element i is calculated $\hat{r}_{u,i} = \mu - \Psi e_i + b_i + b_u + q_i^t \cdot p_u$, so the loss function is minimized as in equation 20.

$$\min_{q^*, p^*, b^*, \Psi e^*} \sum_{(u,i)\in K} \begin{pmatrix} \left(r_{u,i} - \mu + \Psi e_i - b_i - b_u - q_i^t \cdot p_u\right)^2 \\ +\lambda(\| q_i \|^2 + \| p_u \|^2 + b_u^2 + b_i^2 + \Psi e_i^2) \end{pmatrix} \tag{20}$$

The calculation of b_u, b_i, p_u and q_i are made by 11, 12, 13 and 14, respectively. To calculate Ψe_i with respect to the gradient, we proceed as in equation 21.

$$\Psi e_i \leftarrow \Psi e_i - \gamma \cdot (e_{u,i} + \lambda \Psi e_i) \tag{21}$$

4.3.2 VZipfMatt_penalized(μ_{QP}−)

This variant seeks to penalize the Matthew effect with respect to the local average μ_{QP}. This variant is like the previous one, except that now the calculations of the deviations and the penalty are made based on μ_{QP}. To predict the rating that user u would give the element i is calculated by $\hat{r}_{u,i} = \mu_{QP} - \Psi e_i + b_i + b_u + q_i^t \cdot p_u$, so the loss function is minimized as in the equation 22.

$$\min_{q^*, p^*, b^*, \Psi e^*} \sum_{(u,i)\in K} \begin{pmatrix} \left(r_{u,i} - \mu_{QP} + \Psi e_i - b_i - b_u - q_i^t \cdot p_u\right)^2 \\ +\lambda(\| q_i \|^2 + \| p_u \|^2 + b_u^2 + b_i^2 + \Psi e_i^2) \end{pmatrix} \tag{22}$$

The calculation of b_u, b_i, p_u and q_i are made by 11, 12, 13, and 14, respectively. To calculate Ψe_i with respect to the gradient, we proceed as in the equation 21.

4.3.3 VZipfMatt_increased (μ_{QP}+)

In this variant, the deviations of the user-item interaction ratings are calculated based on μ_{QP}, and the popularity biases are added to the predictions for contrast purposes with respect to the variants that penalize said biases. To predict the rating that user u would give the element i is calculated by $\hat{r}_{u,i} = \mu_{QP} + \Psi e_i + b_i + b_u + q_i^t \cdot p_u$, so the loss function is minimized as in equation 23.

$$\min_{q^*, p^*, b^*, \Psi e^*} \sum_{(u,i)\in K} \begin{pmatrix} \left(r_{u,i} - \mu_{QP} - \Psi e_i - b_i - b_u - q_i^t \cdot p_u\right)^2 \\ +\lambda(\| q_i \|^2 + \| p_u \|^2 + b_u^2 + b_i^2 + \Psi e_i^2) \end{pmatrix} \tag{23}$$

The calculation of b_u, b_i, p_u , p_i, and Ψe_i are made by 11, 12, 13, 14, and 24 respectively.

$$\Psi e_i \leftarrow \Psi e_i + \gamma \cdot (e_{u,i} - \lambda \Psi e_i) \tag{24}$$

SGD is a popular approach because it is easy to implement and has a relatively fast execution time during training (Koren et al., 2009).

5 Experimental Work

Table 1 shows the terminology used in the experimentation, and its description.

Table 1. Terminology

Notation	Description
$[Q : P]$	Window of elements little affected by popularity bias and lack of coverage
$Top - k$	List of the k recommended items for the active user
P_{Matt}	Transition point where popularity biases begin
N_k	Set of predicted items, based on $Top - k$, of all users of the test data
R_k	Set of relevant items, based on $Top - k$, from all users of the test data
N_F	Set of predicted items, and that were relevant outside the $Top - k$ of the test data
$\|N_F\|(\%)$	Percentage of the number of predicted items that were relevant outside the $Top - k$ of the test data
N_E	Set of predicted items, based on $Top - k$, that were not relevant in the test data
$\|N_E\|(\%)$	System error percentage
N_D	Set of different elements predicted by the system
$\|K\|$	Number of ranks

For the experimental work, the basic models Normalized-SVD and Bias-SVD were implemented, and three variants of the second. Table 2 shows the models and variants, specifies their identification keys in the text, and provides a brief description. In all variants, the popularity biases of the elements at the head of the data distribution are included. The implementation was done in Python using libraries such as NumPy, and Tensor Flow from Google.

Table 2. Models and variants studied

Model/Variant	ID	Description
Regularized-SVD	M1	Basic, regularized SVD model
Bias-SVD	M2	Classic SVD model, with biases relative to user-item interaction with respect to the global average μ
VZipfMatt_penalized($\mu-$)	M2V1	Variant of M2: that penalizes popularity biases in predictions
VZipfMatt_penalized$\left(\mu_{QP} -\right)$	M2V2	Variant of M2: which exchanges μ for μ_{QP} and penalizes popularity biases in predictions
VZipfMatt_increased$\left(\mu_{QP} +\right)$	M2V3	Variant of M2: which exchanges μ for μ_{QP} and increases popularity biases in predictions

For experimentation, 60% of the data was taken for training and 40% for testing, and elements with frequency one and two were not considered. The error of the loss function was calculated using the MSE metric, and the minimization was performed using the Adam optimizer. In the experimentation, the models/variants are compared based on Root Mean Square Error (RMSE) and the fairness percentage. For each user, a list of recommendations was generated based on the $Top - 20$ of the test data with prediction $\hat{r}_{u,i} >= 4$. The penalty constant was set to $\beta = 1$ for experiment A and $\beta = 3$ for experiment B. 80 Latent Factors (LF) were used, the regularization constant was $\lambda = 0.05$, the degree of learning was set at: $\gamma = 0.02$.

5.1 Data used

For the experimental work, two MovieLens datasets were used, which contain ratings obtained explicitly in the user-item interaction in the domain of movies (Harper and Constan, 2016). In the area of RS research, they are often used as benchmarks to evaluate and compare the performance of algorithms. As can be seen in Table 3, the dataset Small contains 100,836 ratings submitted by 610 users for 9,724 movies they had seen. Each rating is a value between 1 and 5 in increments of 1, as the rating system is five stars. The MovieLens 100K dataset contains 100,000 ratings of 1,682 movies, submitted by 943 users. Each rating is a value between 1 and 5 with increments of 0.5. Both datasets are structured so that each row represents an individual rating, which facilitates analysis. In Table 3, the “Density” column refers to the percentage density of the ratings matrix for each dataset.

The “Ratings ∝ Movies” column represents the ratings proportion of the 20% most frequently viewed movies. In this sense, the 78%-20% ratio in the small dataset is closer to the Pareto ratio than the 64%-20% ratio in the 100K dataset. The column “|K|” refers to the number of ranks found through the methodology described in section 4. Columns “Q” and “P” indicate the frequencies, in descending order, where the frequency windows containing each P_{Matt} transition point corresponding to each dataset start and end, respectively.

Table 3. Datasets used in the experiments

MovieLens	Users	Movies	Ratings	Density	Ratings ∝ Movies	|K|	Q	P	P_{Matt}
Small	610	9,724	100,836	1.7%	78% - 20%	175	98	8	19
100K	943	1,682	100,000	6.3%	64% - 20%	270	118	37	64

Fig. 1 and Fig. 2 show the long-tail curves of the datasets. The points Q and P delimit the window where the local average of the ratings is calculated μ_{QP}, while P_{Matt} illustrates the Matthew effect penalty limit.

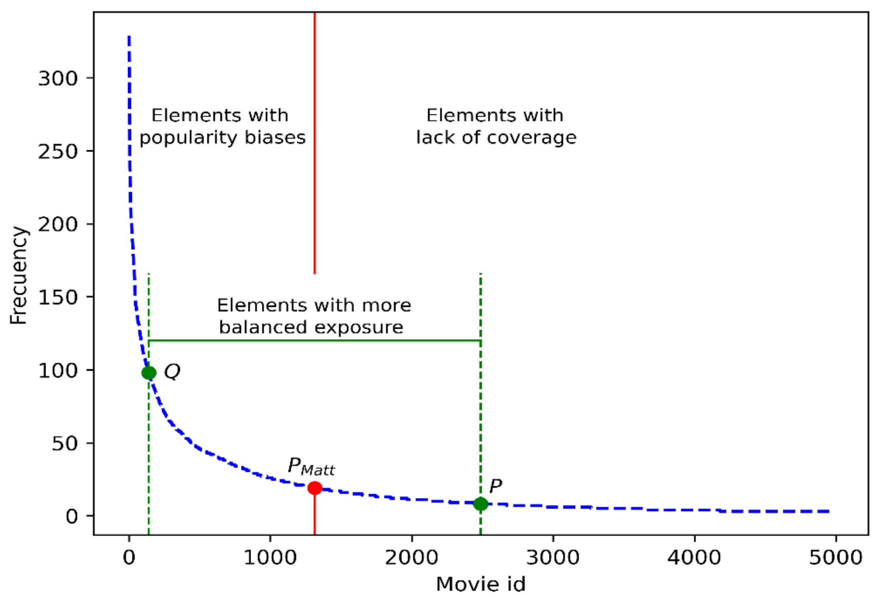

Fig. 1. Long tail curves for the dataset: MovieLens Small.

5.2 Evaluation

To evaluate the efficiency of the predictions, the classic RMSE metric of equation 25 was used, which, in general terms, calculates the average deviation between the prediction P_i and the real value V_i, where N is the number of user-item interactions in RS.

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(P_i - V_i)^2} \quad \textbf{(25)}$$

5.2.1 Fairness and Error

To make a recommendation list fairer, RS should recommend relevant items other than the most popular ones. In this work, we calculate fairness by considering relevant recommendations that are outside the $Top - k$ of the test data, this is because the test data is biased with respect to the popularity of the items. So, we calculate fairness as the percentage of the fraction of N_F with respect to N_k, as shown in equation 26.

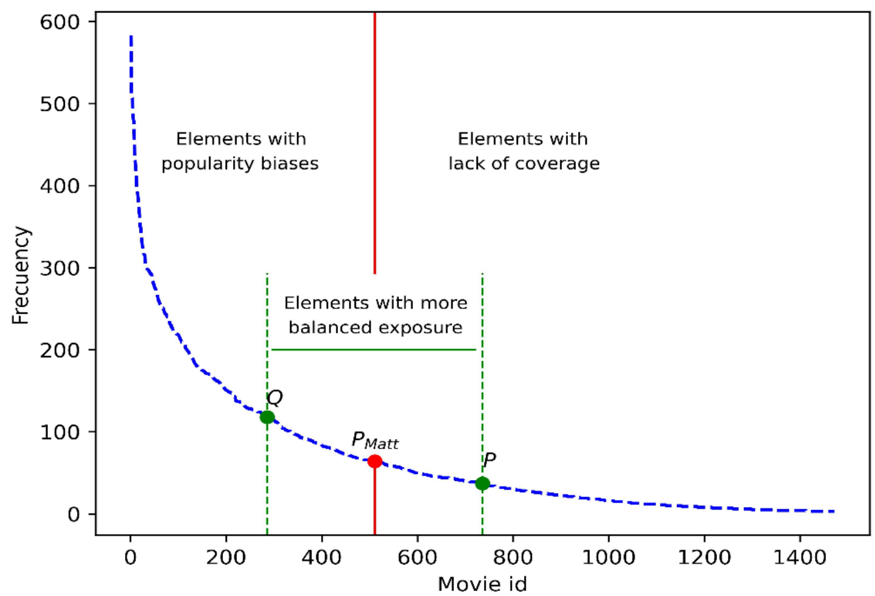

Fig. 2. Long tail curves for the dataset: MovieLens 100K.

$$Fairness = \left(\frac{|N_F|}{|N_k|}\right) \times 100 \quad \textbf{(26)}$$

We also measure how much the probability distribution of the test data differs from the probability distribution of the recommendations of each model using KL-Divergence, as shown in equation 27.

$$D_{KL} = D_{KL}(\mathrm{P} \,||\, \mathrm{Q}) = \sum_{x} P(x) log\left(\frac{\mathrm{P(x)}}{\mathrm{Q(x)}}\right) \quad \textbf{(27)}$$

Where P and Q are the true and approximate probability distributions, respectively. Both distributions are with respect to the same discrete variable x.

A large value of D_{KL} would indicate that the distributions differ significantly, and therefore there would be greater diversity and fairness in the model's recommendations compared to the test data, given that the latter have popularity biases.

On the other hand, the system error refers to the fraction of N_E with respect to N_k, as shown in equation 28.

$$Error = \left(\frac{|N_E|}{|N_k|}\right) \times 100 \quad \textbf{(28)}$$

5.3 Experiment A: Reducing the Matthew Effect in Recommendations

To prove that it is possible to reduce the Matthew effect in the recommendations generated by models based on the Vanilla-Matrix-Factorization framework, incorporating popularity biases, the models M1 and M2 were implemented, and three variants of the second in which said biases are incorporated. The M1 model was implemented for contrast purposes.

The results are presented in Table 4: the RMSE was obtained in the training epoch before deregulation; the best results are shown in bold. The $|N_D|$column refers to the number of different items recommended by each model. This measure gives us an idea of the diversity in the recommendation lists.

Table 4. Results of experiment A with β = 1: RMSE

Model/Variant	MovieLens Small			MovieLens 100k		
	$\|N_D\|$	RMSE	epoch	$\|N_D\|$	RMSE	epoch
M1	754	1.0398748	7,568	559	1.0877778	8,876
M2	1,164	**0.8274392**	11,197	491	**0.9407438**	11,127
M2V1	1,200	**0.8276111**	10,756	498	**0.9460612**	10,431
M2V2	1,246	**0.8277342**	10,983	575	**0.9476216**	11,042
M2V3	1,179	0.8396349	12,425	565	0.9960378	12,409

Tables 5 and 6 present the results for the fairness and error of the recommendation lists. The best results are shown in bold. The $|N_k|$ column refers to the total number of recommendations generated by each model. The $|N_F|$ column shows the number of relevant items that were left out of the top 20. The $|N_E|$ column shows the number of items that were recommended but were not relevant. The mean $\bar{x}$ refers to the average frequency with which items were recommended. The variance σ^2 is included to provide an indicator of the variability in recommendation frequencies. The smaller the variance, the lower the popularity bias of the model's recommendations, and the greater the fairness of the recommendation lists. Both fairness and error were calculated as in equations 26 and 27, respectively. Column D_{KL} shows the measure of information loss when the probability distribution of each model's recommendations is used to approximate the probability distribution of the test data's recommendations.

Table 5. Results of experiment A for MovieLens Small and β = 1: Fairness and Error

Model/Variant	$\|N_k\|$	$\bar{x}$	σ^2	$\|N_F\|$	Fairness %	$\|N_E\|$	Error %	D_{KL}
M1	4,682	6.2177957	79.1238793	1,256	**26.826**	990	21.1448	1.28
M2	5,937	5.1049014	119.485126	1,267	21.340	997	16.7929	2.15
M2V1	5,577	**4.6513762**	**72.7375106**	1,342	**24.063**	911	**16.3349**	**2.29**
M2V2	5,850	**4.6987954**	**72.6072677**	1,404	**24.0**	972	**16.615**	1.84
M2V3	6,582	5.5874363	164.374782	1,302	19.781	1,232	18.7177	**2.22**

Table 6. Results of experiment A for MovieLens 100K and β = 1: Fairness and Error

Model/Variant	$\|N_k\|$	$\bar{x}$	σ^2	$\|N_F\|$	Fairness %	$\|N_E\|$	Error %	D_{KL}
M1	7,563	**13.553763**	**315.247109**	1,704	**22.5307**	1,555	20.5606	0.60
M2	7,672	15.657142	691.600816	1,419	18.4958	1,169	**15.2372**	1.82
M2V1	6,470	**13.018108**	**418.118384**	1,445	**22.3338**	926	**14.3122**	**2.23**
M2V2	8,000	**13.937282**	**446.818366**	1,688	**21.1**	1,308	**16.35**	1.11
M2V3	10,859	19.253546	1,105.34528	1,554	14.3107	2,246	20.6833	**1.85**

Finally, Table 7 presents the distribution of the $|N_k|$ recommendations and their respective percentages with respect to each class. Each class, except for class F, contains 10% of the items in the list sorted in descending order of frequencies. The best results regarding the reduction of the Matthew effect are shown in bold. These results can be observed visually in Fig. 3 and Fig. 4, whose bar graphs show the popularity biases of the elements with respect to the closeness of the frequency distribution in relation to the Pareto proportion. The dotted lines show the Matthew effect of the recommendations generated with respect to the elements of each class.

5.4 Experiment B: Reducing the Matthew Effect with $\beta = 3$

In this experiment, the Matthew effect is penalized with $\beta = 3$, to contrast the results against the experiment that uses $\beta = 1$, it is expected that by making a stronger penalty, the fairness of the recommendations will improve. The results, with respect to RMSE, are shown in Table 8; the best results are highlighted in bold. The ND column shows the number of different recommendations for each model or variant. To contrast the results of M2 and M2V2 with similar RMSE, model M2 was run with controlled error (w.c.e).

Tables 9 and 10 show the fairness of each model, obtained with respect to N_k and N_F. The system error based on the generated recommendation lists is also shown, i.e., the proportion of recommendations that were not relevant to the test data. The distribution of recommendations by class can be observed in Table 7, while the corresponding bar graphs are shown in Fig. 5 and Fig. 6.

Table 7. Results of experiments A and B: Matthew effect

Exp	Model/Variant	MovieLens Small						MovieLens 100k					
		A	B	C	D	E	F	A	B	C	D	E	F
	M1	4,451	193	27	11	0	0	5,288	1,532	488	172	67	16
	% →	95.07	4.12	0.58	0.23	0	0	69.92	20.26	6.45	2.27	0.89	0.21
	M2	4,825	556	281	158	103	14	5,285	1,407	506	276	105	93
A:	% →	81.27	9.36	4.73	2.66	1.73	0.24	68.89	18.34	6.59	3.60	1.37	1.21
$\beta = 1$	M2V1	4,378	598	306	169	111	15	4,077	1,338	486	327	130	112
	% →	**78.50**	**10.72**	**5.49**	**3.03**	**1.99**	**0.27**	**63.01**	**20.68**	**7.51**	**5.05**	**2.01**	**1.73**
	M2V2	4,587	630	320	177	120	16	4,754	1,738	464	464	188	163
	% →	**78.41**	**10.77**	**5.47**	**3.03**	**2.05**	**0.27**	**59.43**	**21.73**	**5.80**	**5.80**	**2.35**	**2.04**
	M2V3	5,514	533	267	151	102	15	8,015	1,783	260	260	106	111
	% →	83.77	8.09	4.06	2.29	1.55	0.23	73.81	16.42	2.39	2.39	0.98	1.02
	M2	4,947	533	264	158	112	11	5,249	1,409	509	271	117	90
	% →	82.11	8.85	4.38	2.62	1.86	0.18	68.65	18.43	6.66	3.54	1.53	1.18
	M2V1	3,884	618	309	174	121	11	3,050	1,223	440	364	171	133
B:	% →	**75.90**	**12.08**	**6.04**	**3.40**	**2.36**	**0.21**	**56.68**	**22.73**	**8.18**	**6.76**	**3.18**	**2.47**
$\beta = 3$	M2V2	4,063	669	334	183	131	12	3,314	1,550	615	540	254	196
	% →	**75.35**	**12.41**	**6.19**	**3.39**	**2.43**	**0.22**	**51.23**	**23.96**	**9.51**	**8.35**	**3.93**	**3.03**
	M2: w.c.e.	5,418	469	232	132	85	10	6,480	1,564	447	211	90	76
	% →	85.38	7.39	3.66	2.08	1.34	0.16	73.07	17.64	5.04	2.38	1.01	0.86

Table 8. Results of experiment B with β = 3: RMSE

Model/Variant	MovieLens Small			MovieLens 100k		
	$\lvert N_D \rvert$	RMSE	epoch	$\lvert N_D \rvert$	RMSE	epoch
M2	1,163	**0.8292837**	11,204	487	**0.9355556**	11,677
M2V1	1,230	0.8507493	10,345	496	0.9961429	10,088
M2V2	1,289	0.8487337	10,462	564	0.9831947	10,346
M2: w.c.e	1,127	0.8487385	9,648	504	0.9831259	9,543

Table 9. Results of experiment B for MovieLens Small and β = 3: Fairness and Error

Model/Variant	$\lvert N_k \rvert$	$\bar{x}$	σ^2	$\lvert N_F \rvert$	Fairness %	$\lvert N_E \rvert$	Error %	D_{KL}
M2	6,025	5.1850258	125.772133	1,320	21.9087	1,041	17.278	2.25
M2V1	5,117	**4.1635475**	**40.9162953**	1,426	**27.8678**	858	**16.7676**	**2.99**
M2V2	5,392	**4.1863351**	**41.5972667**	1,489	**27.6149**	929	**17.229**	**3.24**
M2: w.c.e.	6,346	5.6358792	131.563686	1,349	21.2574	1,210	19.0671	2.02

Table 10. Results of experiment B for MovieLens 100K and β = 3: Fairness and Error

Model/Variant	$\lvert N_k \rvert$	$\bar{x}$	σ^2	$\lvert N_F \rvert$	Fairness %	$\lvert N_E \rvert$	Error %	D_{KL}
M2	7,645	15.730452	668.410883	1,421	18.587	1,225	16.0235	1.87
M2V1	5,381	**10.870707**	**237.595401**	1,480	**27.504**	807	**14.997**	**2.52**
M2V2	6,469	**11.490231**	**260.228590**	1,772	**27.392**	1,027	**15.875**	**2.60**
M2: w.c.e.	8,868	17.630219	683.298649	1,573	17.7379	1,764	19.8917	1.27

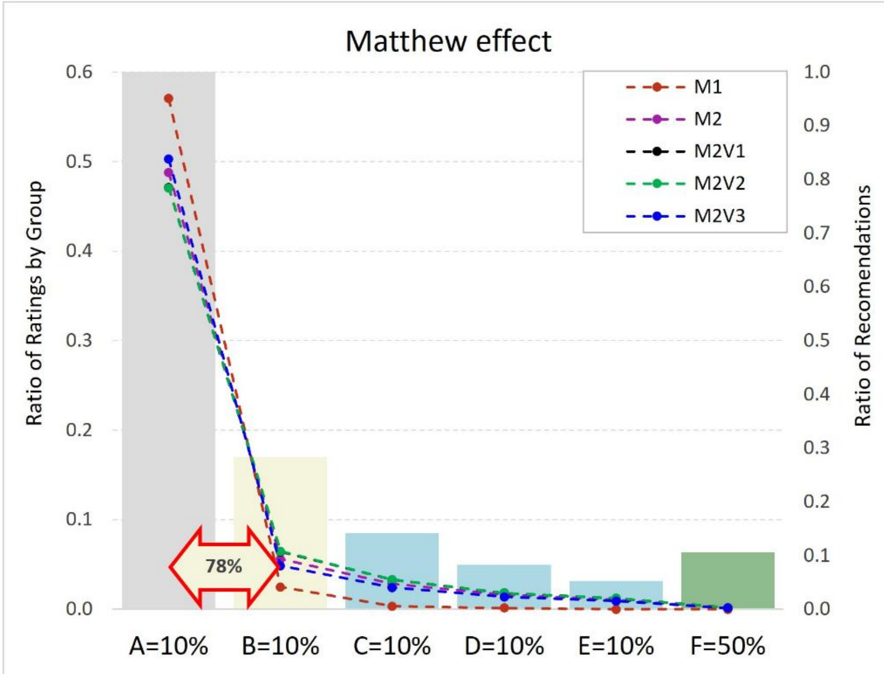

Fig. 3. Experiment A, with MovieLens Small: Matthew effect on recommendations. Bars A and B illustrate the closeness of the distribution to the Pareto proportion; 20% of the elements have 78% of the ratings. The lines show the ratio of recommendations generated for each group.

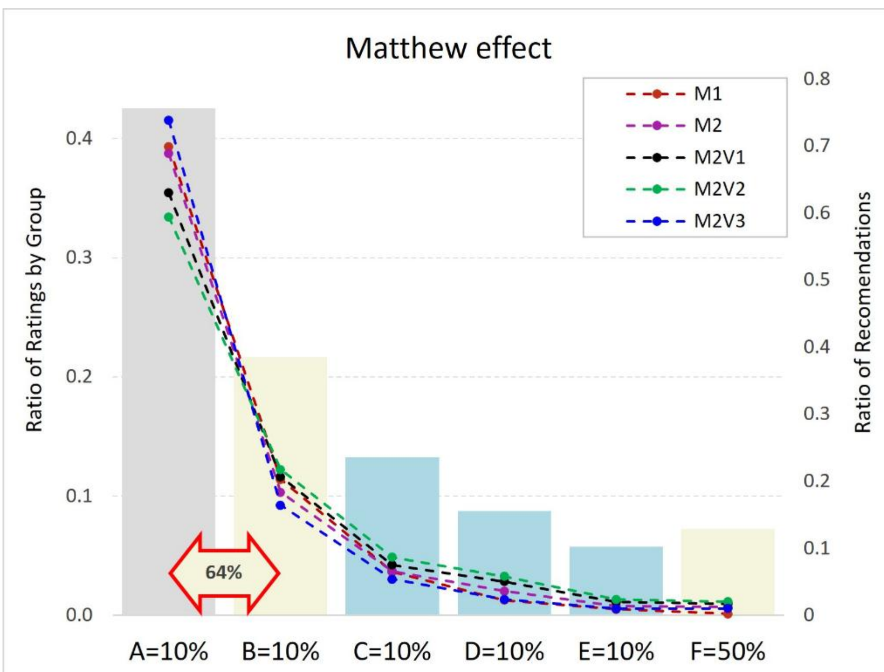

Fig. 4. Experiment A, with MovieLens 100K: Matthew effect on recommendations. Bars A and B illustrate the closeness of the distribution to the Pareto proportion; 20% of the elements have 64% of the ratings. The lines show the ratio of recommendations generated for each group.

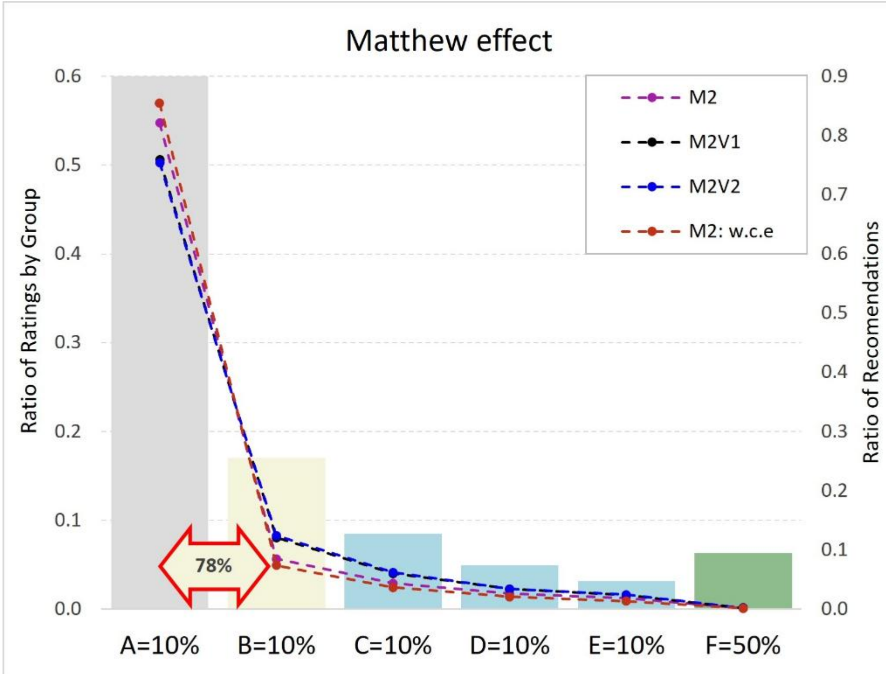

Fig. 5. Experiment B, with MovieLens Small: Matthew effect on recommendations. Bars A and B illustrate the closeness of the distribution to the Pareto proportion; 20% of the elements have 78% of the ratings. The lines show the ratio of recommendations generated for each group.

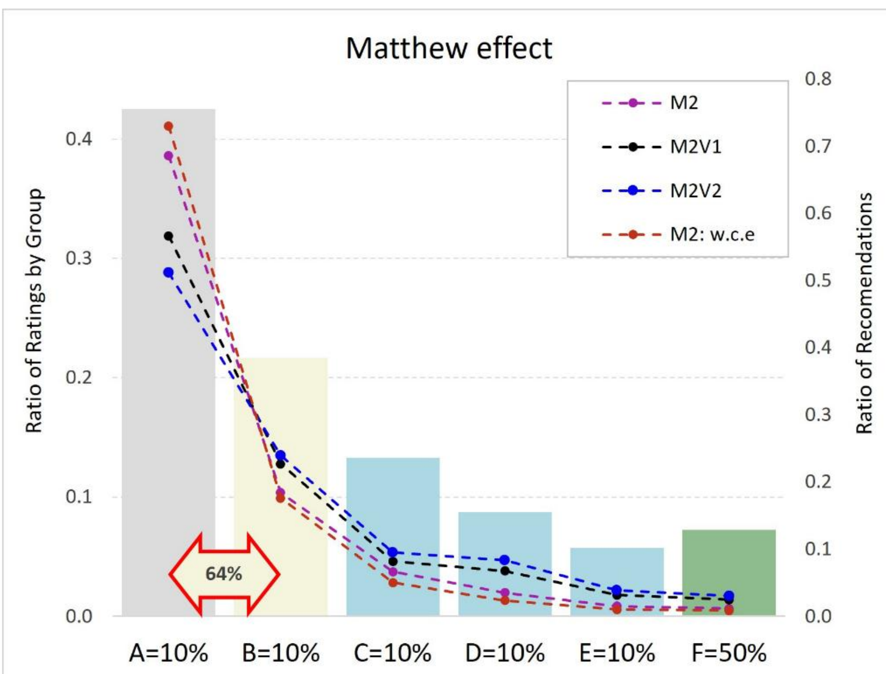

Fig. 6. Experiment B, with MovieLens 100K: Matthew effect on recommendations. Bars A and B illustrate the closeness of the distribution to the Pareto proportion; 20% of the elements have 64% of the ratings. The lines show the ratio of recommendations generated for each group.

6 Analysis of results

With experiments A and B, we have studied the base models Regularized-SVD and Bias-SVD with respect to the variants M2V1, M2V2, and M2V3, with penalty degree $\beta = 1$ and $\beta = 3$.

Table 4 shows the results of experiment A. As can be seen, M2 gave the best performance and M1 the worst with respect to RMSE. This was expected, since M2 is an evolution of M1. In the same sense, the M2V1 and M2V2 variants had a similar and competitive RMSE with respect to their base model, M2. The M2V3 variant performed the worst of the three variants, with RMSE = 0.9960378, very close to that of M1 on the MovieLens 100K data. An error of 1 would mean that a rating of 4 in the test data can be predicted with a value of 5 (relevant) or 3 (not relevant) if the relevance was set to a value greater than or equal to 4. On the other hand, the M2V3 variant was the one that achieved the most recommendations compared to the other variants, but at the same time, it was the one that generated the fewest different recommendations. This indicates a reduction in diversity; furthermore, both $\bar{x}$ and σ^2 were the highest with respect to M2 and its variants, which means that M2V3 carries the popularity biases and, therefore, tends to recommend popular elements. The M2V1 and M2V2 variants have subtle differences that can be seen in both datasets: While M2V1 appears to compete better with M2 with respect to RMSE, M2V2 seems to be better suited to generating fairer and more diverse recommendations. In experiment B, with $\beta = 3$, the same behavior can be observed, as in experiment A, between M2, M2V1, and M2V2, with respect to RMSE. It can be observed that as the popularity bias penalty increases, RMSE also increases, and for the MovieLens 100K data, the variants are no longer competitive with respect to the base model.

Regarding the Matthew effect, in the results of experiment A in Table 7, it can be observed that M1 and M2V3 are the ones that generated the highest percentages of predictions of class A, which is shown graphically in Fig. 3 and Fig. 4. In that sense, M2V1 and M2V2 show better performance than their base model, M2, but M2V2 subtly outperforms M2V1. This reduction of the Matthew effect in the variants with respect to the base model can be observed more clearly in the results of experiment B, when $\beta = 3$, see Table 7 and Fig. 5 and Fig. 6. As with $\beta = 1$, with $\beta = 3$ the M2V2 variant generates recommendations that are better distributed across all classes, and M2: w.c.e had the worst performance. Which proves that the decrease in the Matthew effect of M2V2 recommendations is not due to the RMSE value but to the penalization of popularity biases and the use of μ_{QP} as the equilibrium point of the system in generating the predictions.

Regarding the fairness of the predictions, in the results of Tables 5 and 6 of experiment A, it can be observed that the variants that penalize the Matthew effect, that is, M2V1 and M2V2, achieved a higher percentage of fairness and a lower error than M2. This result is more conclusive in experiment B; see Tables 9 and 10. Although the performance of these variants is similar, M2V1 achieved subtly higher fairness and lower error than M2V2. M2:w.c.e generated less fair and more erroneous predictions in all cases. Recall that M2 stopped when its RMSE was comparable to that of M2V2; this is done to observe its behavior with respect to this variant. However, the results show that M2V2 predicts more fairly and with less error than M2:w.c.e. So, the improvement in M2V2 fairness does not depend on RMSE but on the Matthew effect penalty and the use of μ_{QP} in the model training stage.

Regarding D_{KL}, the results in Tables 5 and 6 show that the probability distribution of recommendations for variants M2V1 and M2V3 differ more significantly from the distribution of the test data than the probability distribution of M2. This is the case except for M2V2, which had a slightly lower result; this is because this variant requires a greater Matthew effect penalty; this can be seen in the results in Tables 9 and 10, where M2V1 and M2V2 achieved the best D_{KL} results when $\beta = 3$ on both datasets. The fact that the variants differ with respect to D_{KL} means that they generate recommendations that are less biased toward popularity, since it is known in advance that the test data have this bias. On the other hand, although variant M2V3 has a high D_{KL} value in Experiment A, it also has higher error, higher variance, and lower diversity, which would explain this result. Therefore, the results of both experiments with respect to D_{KL} confirm that variants M2V1 and M2V2 improve the diversity and fairness of the recommendations of their base model M2, but the value of β is relevant. This indicates that the proposed methodology for reducing popularity biases in the model training stage leads to fairer recommendation lists.

7 Conclusions and future work

In this research, a methodology has been proposed that makes use of the Zipf distribution and the transition point P_{Matt} between the elements that have popularity biases and those that do not to reduce the Matthew effect of the recommendations generated by the Regularized-SVD and Bias-SD base models. The underlying idea is to calculate the popularity biases of the elements at the head of the data distribution, and penalize or add them as appropriate, based on μ or μ_{QP}, in the training stage.

Two groups of experiments were implemented: in experiment A, with $\beta = 1$, the behavior of the models M1 and M2, and the variables M2V1, M2V2 and M2V3 are explored, with respect to RMSE and fairness. In experiment B, these same characteristics are observed in M2, M2V1, M2V2, and M2:w.c.e, with penalty constant $\beta = 3$. It is proven, in a data-driven manner, that penalizing popularity biases in the data reduces the Matthew effect and improves the fairness of predictions generated by Bias-SVD variants. Although the level at which this feature improves depends on the degree of penalty β.

Another result was that M2V1 and M2V2 have similar behavior, but although subtly, M2V1 is better than M2V2 in RMSE and fairness. While M2V2 had better results regarding the reduction of the Matthew effect.

Therefore, we can conclude that the proposed methodology, which uses the Zipf distribution, and P_{Matt} as a transition point between popular and unpopular elements, as well as μ_{QP} as a balance point of some proposed variants, is suitable to reduce the Matthew effect and improve the fairness of recommendations.

As future work, we would like to test the usefulness of the methodology in models that use biases based on demographics and temporality, as well as continue to investigate strategies that further improve the fairness of recommendations. We would like to investigate the possibility of improving the RMSE of the Bias-SVD model by incorporating elements of this methodology that incorporates popularity biases.

Acknowledgments

To the Autonomous University of Tlaxcala and the Technological University of Izúcar of Matamoros for supporting this research.

References

Barabási, A. C. (2013). Network science. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 371(1987), 1–63.

Booth, A. D. (1967). A law of occurrences for words of low frequency. *Information and Control*, 10(4), 386–393.

Chen, J., Dong, H.,Wang, X., Feng, F.,Wang, M., & He, X. (2023). Bias and Debias in Recommender System: A Survey and Future Directions. *ACM Transactions on Information Systems*, 41(3), 1–39.

Gower, S. (2014). Netflix prize and svd. *University of Puget Sound*, 1–10.

Harper, F. M. & Konstan, J. A. (2016). The MovieLens Datasets: History and Context. *ACM Transactions on Interactive Intelligent Systems*, 5(4), 1–19.

He, X., Liao, L., Zhang, H., Nie, L., Hu, X., & Chua, T. S. (2017). Neural collaborative filtering [Preprint]. *Arxiv.* https://arxiv.org/abs/1708.05031.

Jiménez-Salazar, H., Pinto-Avendaño, D. E., & Rosso, P. (2005). Uso del punto de transición en la selección de términos índice para agrupamiento de textos cortos. *Procesamiento del lenguaje natural*, (35), 383–390.

Koren, Y. (2008). Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 426–434), Las Vegas Nevada USA. ACM.

Koren, Y., Bell, R., & Volinsky, C. (2009). Matrix Factorization Techniques for Recommender Systems. *Computer*, 42(8), 30–37.

Kumar, R., K. Verma, B., & Sunder Rastogi, S. (2014). Social Popularity based SVD++ Recommender System. *International Journal of Computer Applications*, 87(14), 33–37.

Lee, I. (2017). Big data: Dimensions, evolution, impacts, and challenges. *Business Horizons*, 60(3), 293–303.

Liang, D., Charlin, L., McInerney, J., & Blei, D. M. (2016). Modeling User Exposure in Recommendation. In *Proceedings of the 25th International Conference on World Wide Web* (pp. 951–961), Montréal Québec Canada.

Lucero-Alvarez, C., Quintero-Flores, P. M., Perez-Cruz, P., Ortiz-Ramirez, C. A., Mendoza-Crisostomo, P., & Montiel-Hernandez, J. (2021). Literature Review on Information Filtering Methods in Recommendation Systems. In *2021 Mexican International Conference on Computer Science (ENC)* (pp. 1–8), Morelia, Mexico. IEEE. https://doi.org/10.1109/ENC53357.2021.9534807

Lucero-Alvarez, C., Quintero-Flores, P. M., Moyotl-Hernández, E., Ortiz-Ramírez, C. A., Mendoza-Crisóstomo, P., & Vázquez-Vázquez, M. (2023). Behavior's study of some classic svd-models with noisy data in movie recommender systems. *Research in Computing Science*, 152(11), 5–16.

Montemurro, M. A. (2001). Beyond the Zipf–Mandelbrot law in quantitative linguistics. *Physica A: Statistical Mechanics and its Aplications*, 300(3-4), 567-578.

Moyotl-Hernández, E. & Macías-Pérez, M. (2016). Método para autocompletar consultas basado en cadenas de Markov y la ley de Zipf. *Research in Computing Science*, 115(1), 157–170.

Netflix Update: Try This at Home. Retrieved May 30, 2025, from https://sifter.org/simon/journal/20061211.html.

Newman, M. (2005). Power laws, Pareto distributions and Zipf's law. *Contemporary Physics*, 46(5), 323–351.

Paterek, A. (2007). Improving regularized singular value decomposition for collaborative filtering. In *KDD Cup Workshop at SIGKDD'07, 13th ACM International Conference on Knowledge Discovery and Data Mining* (pp. 5–8).

Perc, M. (2014). The Matthew effect in empirical data. *Journal of The Royal Society Interface*, 11(98), 20140378.

Piantadosi, S. T. (2014). Zipf's word frequency law in natural language: A critical review and future directions. *Psychonomic Bulletin & Review*, 21(5), 1112–1130.

Pinto, D., Jiménez-Salazar, H., & Rosso, P. (2006). Clustering abstracts of scientific texts using the transition point technique. In *International Conference on Intelligent Text Processing and Computational Linguistics* (pp. 536–546), Springer Berlin Heidelberg.

Sharifi, Z., Rezghi, M., & Nasiri, M. (2013). New algorithm for recommender systems based on singular value decomposition method. In *ICCKE 2013* (pp. 86–91), Mashhad, Iran. IEEE.

Sun, W., Khenissi, S., Nasraoui, O., & Shafto, P. (2019). Debiasing the human-recommender system feedback loop in collaborative filtering. In *Companion Proceedings of The 2019 World Wide Web Conference* (pp. 645-651).

Urbizagástegui Alvarado, R. & Restrepo Arango, C. (2011). La ley de Zipf y el punto de transición de Goffman en la indización automática. *Investigación Bibliotecológica. Archivonomía, Bibliotecología e Información*, 25(54), 71.

Wang, H. (2021a). ZeroMat: Solving Cold-start Problem of Recommender System with No Input Data. In *2021 IEEE 4th International Conference on Information Systems and Computer Aided Education (ICISCAE)* (pp. 102–105), Dalian, China. IEEE.

Wang, H. (2021b). Zipf Matrix Factorization: Matrix Factorization with Matthew Effect Reduction. In *2021 4th International Conference on Artificial Intelligence and Big Data (ICAIBD)* (pp. 217–220), Chengdu, China. IEEE.

Wang, H. (2022a). Alleviating Sparsity Problem of Recommender System with No Extra Input Data. In *2022 5th International Conference on Advanced Electronic Materials, Computers and Software Engineering (AEMCSE)* (pp. 691–695), Wuhan, China. IEEE.

Wang, H. (2022b). DotMat: Solving Cold-Start Problem and Alleviating Sparsity Problem for Recommender Systems. In *2022 IEEE 5th International Conference on Electronics Technology (ICET)* (pp. 1323–1326), Chengdu, China. IEEE.

Wang, H.,Wang, Z., & Zhang,W. (2018). Quantitative analysis of Matthew effect and sparsity problem of recommender systems. In *2018 IEEE 3rd International Conference on Cloud Computing and Big Data Analysis (ICCCBDA)* (pp. 78–82), Chengdu. IEEE.

Wang, Y., Liang, D., Charlin, L., & Blei., D. M. (2019). The deconfounded recommender: A causal inference approach to recommendation [Preprint]. *Arxiv.* https://arxiv.org/abs/1808.06581.

Wei, T., Feng, F., Chen, J., Wu, Z., Yi, J., & He, X. (2021). Model-Agnostic Counterfactual Reasoning for Eliminating Popularity Bias in Recommender System. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining* (pp. 1791–1800), Virtual Event Singapore, ACM.

Zheng, Y., Tang, B., Ding, W., & Zhou, H. (2016). A neural autoregressive approach to collaborative filtering. In *33 International Conference on Machine Learning* (pp. 764–773), New York, NY, USA. JMLR: WCP.

Zhu, Y., Zhang, B., Wang, Q. A., Li, W., & Cai, X. (2018). The principle of least effort and Zipf distribution. *Journal of Physics: Conference Series*, 1113, 012007.

Zhu, Z., Hu, X., & Caverlee, J. (2018). Fairness-aware tensor-based recommendation. In *Proceedings of the 27th ACM international conference on information and knowledge management* (pp. 1153-1162).

Zipf, G. K. (2016). Human behavior and the principle of least effort: An introduction to human ecology. Addison-Wesley.