NOVA IMS Information Management School

MEGI

Mestrado em Estatística e Gestão de Informação

Master Program in Statistics and Information Management

PRESENTATION BIAS IN MOVIE RECOMMENDATION ALGORITHMS

Fernanda Velasco Garat

Dissertation presented as partial requirement for obtaining the Master's degree in Statistics and Information Management

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação

Universidade Nova de Lisboa

NOVA Information Management School

Instituto Superior de Estatística e Gestão de Informação

Universidade Nova de Lisboa

PRESENTATION BIAS IN MOVIE RECOMMENDATION ALGORITHMS

by

Fernanda Garat

Dissertation presented as the partial requirement for obtaining a Master's degree in Statistics and Information Management, specialization Information Analysis and Management

Advisor: Flávio L. Pinheiro

Co Advisor*:* Francisco R. Almeida

September 2020

ABSTRACT

The emergence of video on demand (VOD) has transformed the way the content finds its audience. Several improvements have been made on algorithms to provide better movie recommendations to individuals. Given the huge variety of elements that characterize a film (such as casting, genre, soundtrack, amongst others artistic and technical aspects) and that characterize individuals, most of the improvements relied on accomplishing those characteristics to do a better job regarding matching potential clients to each product. However, little attention has been given to evaluate how the algorithms' result selection are affected by presentation bias. Understanding bias is key to choosing which algorithms will be used by the companies. The existence of a system with presentation bias and feedback loop is already a problem stated by Netflix. In this sense, this research will fill that gap providing a comparative analysis of the bias of the major movie recommendation algorithms.

KEYWORDS

Recommendation system; Algorithm; Presentation bias; Netflix; Video on demand (VOD)

INDEX

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

Item-KNN	Item K-Nearest Neighbor
RBM	Restricted Boltzmann Machines
RMSE	Root-Mean-Square Error
SVD	Singular Value Decomposition
SVD++	Singular Value Decomposition ++
VOD	Video on demand

1. INTRODUCTION

Turning data into meaningful information has been one of the greatest challenges of this century due to the exponential and continuous increase in the amount of digital information. With the wide spread of data came the difficulty of choosing the most valuable one, instantaneously. In order to bypass this issue, information filtering systems started to be developed to select and present relevant information to an individual. It gained relevance in the business environment when a new technology, called recommendation system, implemented their methods focusing on personalizing users' experience by predicting the users´ preference and suggesting products to them accordingly to their taste. Recommendations systems has become an essential tool in many technology-driven companies such as Amazon, Facebook, LinkedIn, Netflix, Spotify, Twitter and YouTube, and changed the way websites interact with their users (Bell & Koren, 2007a). Today, it is one of the most popular application of data science and big data processing.

In this framework, movie recommendation system is one which filter an extensive catalogue of films, with the aim to generate predictions of which movies the consumer would enjoy watching, and thus to provide those predictions as suggestions to them. By doing this, the system also supports the individual decision-making (Lops, Musto, Narducci, & Semeraro, 2019).

Within the film market, with arrival of video on demand (VOD), movie recommendation system acquired a strong commercial application. Therefore, the development of algorithms used by this system has become a popular subject, both in terms of academic and business research (Adomavicius & Tuzhilin, 2005).

Much has been done in respect of creating new algorithms or improving the performance of the existing ones (Su & Khoshgoftaar, 2009). Instead of focus directly on that kind of development, this research is centred on presentation bias analysis (also known as position bias[1]), which is a bias that influences the selection of the movies recommended in a user interface (Corduneanu & Kim, 2015). This bias had already affected the system of the online movie rental company Netflix, requiring control (Gomez-Uribe & Hunt, 2015). The aim of this work is to explore presentation bias on the most used movie recommendation algorithms.

1.1. BACKGROUND AND PROBLEM IDENTIFICATION

In VOD sector, initially led by Netflix, a recommender system was developed to predict the rating each person would give to a movie, on the scale from 1 to 5 stars (from the worst to best classification). During 2006 and 2009, Netflix promoted the popular "Netflix Prize", a contest focused on improving accuracy - evaluated by the root-mean-square error (RMSE) - of its recommendation system (Bennett & Lanning, 2007). The competition was a huge success and was attended by more than 40.000 teams from 186 countries (Hallinan & Striphas, 2016).

[1] in this work we also use "popularity bias" as a term related to position bias.

The ratings given by the users are an example of explicit feedback data. Currently, implicit feedback (individual´s purchases, browsing history, number of clicks, etc) are also taken into consideration (Hu, Koren, & Volinsky, 2008). For example, VOD companies nowadays monitor the consumption pattern of movies: the recommendations offered by the system that were not selected by a person, the moment the consumer ceases to watch certain content, etc. (Gomez-Uribe & Hunt, 2015). Explicit feedback data remains widely studied (Li & Chen, 2016), and each time more and more variety of data (explicit and implicit feedback) is collected, increasing the complexity of its analysis.

Despite the rapid growth, VOD still is a nascent market and a new field of research. The existence of a system where the recommended movies have more chance to be recommended again has received little attention by researchers. Netflix pointed out that one of their concern is the existence of a system with a strong positive feedback loop, since the movies with more interaction tend to be suggested to other individuals (as they have a higher probability of being approved by the average public), which ends up giving them greater visibility and increasing their chance to be watched (Gomez-Uribe & Hunt, 2015). This study is intended to explore this problem through a better understanding of presentation bias by inspecting the outputs generated by the algorithms.

1.2. STUDY OBJECTIVES

Our first goal is to identify the main algorithms (or algorithm families) used in video recommendation system and to define the ones to be studied. In this examination, it is important to find points of congruence between what has been explicitly mentioned in the academic community and what has been used in the business context, which is not always so explicit and observable.

Once done that, our major objective is to discuss the presence of position bias on movie recommendation algorithms. There are two antagonistic assumptions inherent to this topic. In one instance, it is known that recommender algorithms, as a mathematical and computational procedure, are designed to reflect the information contained in data. Thus, accordingly to this assumption, presentation bias in strict sense wouldn´t exist since the algorithm's outputs would be reflecting a pre-existing bias in human´s preferences. That is, humans would be the responsible ones for being biased towards a particular set of movies. In another instance, it is known that the prominence given to any product influences consumers' decisions; otherwise, there would be no need for goods to be promoted. That is, if humans are biased and has their preference entirely predetermined, there is no point in suggesting anything to them. The algorithm´s purpose itself is lost. Under this perspective, presentation bias would end up being coming from some inherent structure of the algorithms. Therefore, this work investigates whether the humans or the algorithms are the responsible for generating presentation bias.

There is also a discussion if VOD could potentialize the distribution of a wider variety of movies (Fontaine & Simone, 2017). First because, compared to the cinema, some movies could benefit from the fact that VOD could allow users to access them easily. For instance, films that have problem in being shown on the main screening rooms of the cities, or that experience difficulty in being shown in a larger number of screening rooms, or that face trouble to be shown at better showtimes.

Analogously, some movies could benefit from being available on VOD for longer period than in the cinemas, especially those with a very short exhibition circuit, often confined to festivals. Second because the recommendation system technology could help people to discover new movies and could also help films to find their target audience easily. In other words, VOD provides the opportunity to exploit the niche markets.

At the same time, we could think the VOD market like any other where the visibility of the products (movies) is crucial for it to be consumed, especially due to the high competitiveness among products, with many being substitute goods, in a much more larger size catalogue. So if the user´s interest decreases as we move through the recommendation list, the interest for an item at the end of a big list would be minimal (Tewari, Singh, & Barman, 2018), and for a film that are not suggested would be almost nil. So, if on the one hand, recommend systems can help the consumption of a great variety of movies, on the other hand, recommend systems can be only reinforcing the popularity of already popular films (Fleder & Hosanagar, 2009).

There are some authors that believes that recommend systems can split consumers into consumption bubbles (Lee & Hosanagar, 2014), encouraging the culture to be segregationist, which could be harmful in terms of the public's right to watch different and varied content from those they already know and approve. Also, a market dominated by a few famous artist or movies becomes homogeneous and with little space for creativity (Abdollahpouri, Burke, & Mobasher, 2019). From this perspective, this research attains to analyse if presentation bias tends to overly reduce diversity in the catalogue of movies.

1.3. STUDY RELEVANCE AND IMPORTANCE

The efficiency of recommender systems in increase sales and in add value to the companies had been demonstrated by multiples articles (Lops et al., 2019), and confirmed by many enterprises, such as Amazon, Google news and Netflix (Lee & Hosanagar, 2014). Netflix stated that their recommender system is one of their most value asset (Amatriain & Basilico, 2012a), and the key pillar (Gomez-Uribe & Hunt, 2015).

The engineering behind recommendation system, which uses several algorithms to predict the best item based on user’s choice in minimum time, demands that companies generates enough revenue to cover that cost (Resnick, Varian, & Editors, 1997). Because of this, some believes that the competition thought market would lead to few players in each area (movies, books, music, etc) able to provide recommendations as a value-added service.

It is important to note that large conglomerates have already announced their entry into the VOD market with their own platform (Economist, 2019): Disney (composed of Pixar, FOX, ESPN, Marvel Studios, Lucas Films, Hulu, among others); A&T (composed by Time Warner, among others), Comcast (composed of Sky, NBC, Universal, among others) and Apple, which indicates that this industry will have a great impact and will face an accelerated transformation in the coming years. In November

2019, Disney Plus streaming service was launch in USA, Canada and Netherlands and in March 2020 it was expanded to Australia, New Zealand, Puerto Rico and some European Countries[2].

With the upcoming competitiveness in the VOD market, the companies will be interested in the usage of algorithms to boost their product and in recommender systems for its economic potential (Bell, Koren, & Volinsky, 2007a). Offering the proper product to the right consumer is not an easy job, since they tend to be more distracted by many other products which also seek to capture their attention. Having a good recommendation system will be a key factor to attract and retain customers (H. Wang & Zhang, 2018).

For a company to choose the better algorithms that are going to be used on their recommendation system it is important to understand its engines in identifying similarities between user profiles or between films, and to measure the system´s accuracy. But, understanding how presentation bias affects those algorithms is a crucial point for a successful business. It is believed that some retailers are blindly applying algorithms without deeply understanding the impact they may have in consumer behaviours and in the company performance (Lee & Hosanagar, 2014). However, only throughout a comprehension of all aspects that affects algorithms, corporations will be able to have a better control on how a movie will be disposed on their platform, wining competitiveness and having a greater chance of success.

It is recognized that recommendation systems have a substantial influence on consumer behaviour (Lops et al., 2019). This study may give a theoretical background to what influences a movie to be watched, in terms of presentation bias. Researchers can use the results of this work to obtain a greater knowledge of presentation bias and to have another perspective from which algorithms can be improved. The result of this study can also be accessed by film producers to evaluate whether their movies have a real chance of being displayed on VOD platform or whether those movies may face similar problems as the ones in being shown on the big screen.

[2] https://en.wikipedia.org/wiki/Disney%2B accessed on June 2020.

2. LITERATURE REVIEW

Recommender system is an artificial intelligence that identify recommendations for individuals based on his previous behaviour and on other individual's historical data. There are two main approaches in terms of reconciling the user's interest with the recommendation to be offered: content-based filtering and collaborative filtering (Patel, 2019). Content-based technique suggests movies to an individual that are similar to previous movies highly rated by the same person. The similarity, therefore, is based on the content similarity. For example, movies' attributes such as genre, director, country, budget, etc, can be used to access similarity (Khan, Chan, Chua, & Haw, 2017). Collaborative filtering suggests movies to an individual based on what other person with similar tastes has liked before, assuming that both pattern of interest can be related (Resnick, Bergstrom, Riedl, & Iacovou, 1994). The similarity, therefore, is based on the individual's preferences similarity. There are different metrics used to measure similarity. The most commonly used are[3]: Pearson correlation coefficient, Euclidean distance and cosine similarity[4]. When content-based filtering and collaborative filtering are combined, or different techniques are used to improve performance, we have a hybrid approach.

Some authors stated that collaborative filtering are biased toward popular items, creating a popularity bias (Zolaktaf, Babanezhad, & Pottinger, 2018), (Celma & Cano, 2008). But before analysing towards which kind of items the algorithms could be generating a bias, we need to better understand what presentation bias is.

There is a fine line that distinguish Information Retrieval (which the main application is search engine) from Information Filtering (which the main application is recommendation systems). Position bias is not a particular problem of recommendation systems but had also been a concern in studies related to algorithms used to search and rank results, such as PageRank[5] (Corduneanu & Kim, 2015).

Concerning the ranking of a search result, the assumption is that the movies recommended are more relevant than the other alternatives. Other strong assumption is that humans must be the true judges of relevance, and the work of algorithms is to try to predict it (Corduneanu & Kim, 2015). However, humans have a tendency to choose the higher ranked outputs (Yue, Patel, & Roehrig, 2010). It had been proved that the position (or the order) of the products is significant in determining user preference for search results (Bar-Ilan, Keenoy, Levene, & Yaari, 2008). In this sense, this implies that the output and suggested results of the algorithms may induce an individual to click on it. If this is the case, it ends up favouring the movies that were initially shown in a device screen.

Meanwhile, it is important to remark that one of the goals of movie recommender systems is the customization of the suggestions according to each person's preference. In order words, recommender system is a new technology that allows to deliver highly targeted movies to the consumer. This is opposite to the concept of choosing a film by its popularity, which is supported by presentation bias. In the extreme case, position bias would induce the system to present the same list to all members (Amatriain & Basilico, 2012b). Besides that, it would be pointless to invest in a

[3] There are many other distance measures such as: Manhattan distance, Minkowski distance, Mahalanobis distance, Hamming Distance, Jaccard index or Tanimoto coefficient.

[4] The definitions will be deepened later, when necessary for the calculation step.

[5] Google´s algorithm.

system whose recommendations were so popular that would be easier to people find them by others means (Hurley & Zhang, 2011).

Nevertheless, in the opposite extreme scenario in which it is only presented unfamiliar movies, the individual may conclude that the system is not good. That is why it is believed that familiar movies create user trust in the recommendation system. And that is why diversity cannot be considered positive in any case, since it would be easily achieved by the recommendation of random films. One of the complexities of setting up a basket of movies to be suggested is managing the balance between familiar and popular movies with unfamiliar and new ones (Kane, 2019).

A broader vision can show us that this challenge may encompass the exploration of the VOD market itself. It is said that long-tail distribution can be applied to the film market (Anderson, 2006). If we plot a graph in which on the vertical axis is measured the number of sales (or the number of 'play') and on the horizontal axis the titles are sorted in descending order by the sales volume, we will find that fewer movies are popular and responsible for sales (formed by the "head" of the distribution curve) and where the long-tail and most of them are the least popular (or the niches ones) with much lower sales (formed by the long-tail). As the number of films in long-tail is high, the total volume of sales coming from there can still being significant.

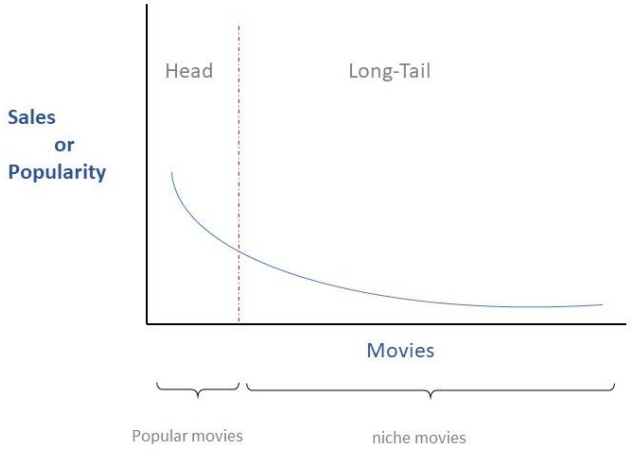


Figure 1 – Long-tail distribution

Some authors claim that companies like Netflix are already benefiting from the long-tail consumption (Yin, Cui, Li, Yao, & Chen, 2012). Many studies have been made to improve the long-tail performance since it is expected that recommendation could turn it more economic valuable (Park & Tuzhilin, 2008). However, the exploration of this tail had been considered a challenge because suggesting less popular movies usually leads to worse accuracy (Huang, Zhao, Yan, & Yang, 2015). So, in other words, there is a the trade-off between accuracy and diversity (L. Chen, Zhang, & Zhou, 2017). In fact, one of the criticisms made is that most of the contributions made on recommender system were mainly evaluated by predictive accuracy metrics, which is not always translated into user experience approval (Musto, Narducci, Lops, de Gemmis, & Semeraro, 2019). The long-tail problem had been commonly attached to popularity bias problem (Abdollahpouri, Burke, Mansoury, & Mobasher, 2019).

A lot had been done to improve the algorithm´s performance with the concern of encompass the whole aspect that compose a movie so that they could better identify the similarities between them and better represent the tastes of the users. For example, in one study regarding the long-tail movies, it was found that the dominant attribute was the director (Jung, 2012). The complexity is not limited to choosing the main attributes, but also, having chosen some, how to make use of an infinity of information. For example, it was also very common to conduct research taking into account the genre of the film (Choi, Ko, & Han, 2012). In one of those studies, it was attempted to identify from 18 genres which ones were most significant for characterizing a movie, and the number was reduced to 6 main genres (Ahmed, Imtiaz, & Khan, 2018). Therefore, dimensionality reduction techniques had been very common in the development of recommendation algorithms since not all information is helpful and that why it is important to focus on the significant ones (Konstan & Riedl, 2012).

Interesting works had also been done to analyse the textual, the audio and the visual domain (Bougiatiotis & Giannakopoulos, 2018), or the stylistic visual features (Quadrana et al., 2016), or the trailer (Deldjoo, Elahi, & Cremonesi, 2016), or to detect key story elements such as turning points (Liu, Last, & Shmilovici, 2019), etc. Sentiment analysis of social media also had been incorporated by recommendation system (Kesharwani, Rakesh, Tech, Abdul, & Prof, 2017). Even the effect of artwork presentation and customization had been refined: which items of the movie should be presented to each user in the VOD platform (for example, it can be choose to emphasize the main actor when the user is a fan), which image is selected by them, or where the image should be positioned in the screen device (Chandrashekar, Amat, Basilico, & Jebara, 2017). In a nutshell, it is vast the number of elements on which recommendation systems can be based.

During the Netflix prize contest, BellKor team (one of the winners) stated that they used 107 different algorithms in the final solution (Bell, Koren, & Volinsky, 2007b). In 2016, Netflix recommendation system was composed by a collection of algorithms that used both supervised and unsupervised machine learning method, serving at least eight main functionalities: Personalized Video Ranker (PVR), Top-N Video Ranker, Trending Now, Continue Watching, Because You Watched (BYW), Page Generation, Evidence and Search (Gomez-Uribe & Hunt, 2015).

Fortunately, in the interest of this research, it was also mentioned by BellKor team that many of those 107 algorithms were close variants and that fewer would be necessary (Bell et al., 2007b). In addition, it was indicated the main approaches behind them: Neighborhood-based model (K-NN), Matrix Factorization models (such as Singular Value Decomposition - SVD), Restricted Boltzmann Machines (RBM) and Regression model. Netflix managed to identify the two best performing algorithms among the 107: SVD and RBM (Amatriain & Basilico, 2012a).

In general, recommender system are mostly relied on collaborative filtering technique (Himabindu, Padmanabhan, & Pujari, 2018). Besides the long-tail problem, collaborative filtering it is also recognized for presenting the following main problems (Ghazanfar & Prugel-bennett, 2010): scalability, cold-start, and sparsity.

Scalability problem regards to fact that the real dataset is massive compared to the dataset used to train the algorithms and can be computational expensive and worse the real-time performance.

Cold-start problem is originated by the learning process of the algorithms since they only can learn from input data. So, when there is a lack of information, typical when there are new clients - for

which any previously rating profile exists - the algorithms have difficult to find the similarities and to make inference about them (Ghazanfar, Iqbal, Azam, Aljohani, & Alowibdi, 2018). Cold-star can be considered the extreme situation of sparsity (He & Chu, 2010).

Sparsity occurs when there is not enough information about the user for prediction or when there are movies with not enough evaluations (Ayyaz, Qamar, & Nawaz, 2018). Sparsity can affect accuracy of recommendation (Choi et al., 2012). Sparsity was pointed as one of the two main reasons the suggestions are biased towards popular movies given that to improve accuracy the recommendations focus the dense region, where are located the few popular movies (Zolaktaf et al., 2018). Analogously, sparsity was pointed to reduce the coverage, which is the percentage of movies recommended in relation to the total films on the catalogue (Ghazanfar & Prugel-bennett, 2010).

It was reported that model-based methods was developed to surpass the scalability problem (Deshpande & Karypis, 2004). Collaborative filtering is classified into memory-based (or user-based, or neighborhood based) and model-based (or item-based) (Sarwar, Karypis, Konstan, & Riedl, 2001). Memory-based measures the similarity between individuals (neighbors) in terms of preferences. In model-based, a model is developed to identify similarities of all pairs of movies and to learn user preferences. Many models' algorithms are used such as Bayesian networks, clustering, matrix factorization, latent semantic, latent dirichlet allocation, etc.

Among the model-based techniques, matrix factorization is the prevalent (Z. Wang, Guo, & Du, 2018). Some authors claims that matrix factorization tackle both cold-start and sparsity problem (He & Chu, 2010), others claims that it tackle scalability and sparsity issue (Wu & Li, 2008). SVD is the most popular application of matrix factorization and it is studied here. Also, we study SVD++, a variant of SVD.

From memory-based models, it is selected the neighbor approach (KNN) algorithm, as it is the most popular (Bell & Koren, 2007a). And from deep learning, it is picked Restricted Boltzmann Machines (RBM). In summary, the algorithms in this paper were selected and motivated by the best models explicitly cited for its usage in the database of Netflix (Bell & Koren, 2007b).

But before exploring them, it is necessary to emphasize two important points. The first is related to the fact that the objective of a recommender system is to show a new movie that a consumer would appreciate. However, this is impossible to test offline because with historical data it is only possible to test the capacity to predict how individual rated shows they already watched (Kane, 2019). It is considered an open question the extent of offline testing may have in the corporate environment (Jannach & Jugovac, 2019). In the real-world, Netflix uses online testing (for instance, A/B test is commonly quoted by them), which allows to evaluate how people appreciate movies that they´ve never seen before. It is important to say that although the higher predictiveness of A/B testing, Netflix doesn't discard the use of offline test, but they have been more limited to preliminary test to save cost, diminish risk and accelerate the innovation (Jannach & Jugovac, 2019), (Gomez-Uribe & Hunt, 2015). Also, offline models prevent scalability problems by reducing time response. As it is not possible to use online test in our research, all mentioned algorithms will be tested offline.

The second point regards to the fact that a good recommendation system should indicate the best top recommendations in front of users, which is a different problem than predict the rating for a movie (Deshpande & Karypis, 2004). After all, what is important is the top-n movies that a person will

get and not if the systems can predict the review they would give (Kane, 2019). To illustrate it better, it shouldn´t matter the films that received bad scores because they are not going to be between the top-n recommendation; but under the predictive rating task, all the opinions (good and bad) are taken into account in order to train the algorithms and achieve better efficiency. And better efficiency doesn't mean having the most useful recommendations for users (L. Chen et al., 2017). Therefore, the usage of accuracy metrics, which is mainly focus on the predictive rating problem, maybe is not the better one. Because of that, Information Retrieval metrics started to be adapted to evaluate the performance of the recommendation (Bellogı, Castells, & Cantador, 2017).

2.1. K-NEAREST NEIGHBOR (K-NN)

As it is of the interest to our research, K-NN search had a large application in recommender systems. Nearest neighbor approach is based on the similarity between pairs of movies or users (Goodfellow, Bengio, & Courvill, 2016). K-NN is considered a simple and a nonparametric and supervised machine learning algorithm that can be used for both classification and regression[6]. As K-NN is more popularly used for classification, K-NN classifier is going to be the one chosen in our research.

In this technique, the output is a predicted class for an object based on the majority class of its neighbors. It uses an instance-based learning algorithm and can be viewed from two perspectives: the customer-based and the movie-based.

The movie-based approach was used by "Big Chaos Solution" and by "BellKor" teams[7], winners of the Netflix Prize (Toscher & Jahrer, 2009b), (Bell et al., 2007b). In this approach, the goal is to find a target number (which is referred to as "k") of the most similar movies.

For the customer-based approach the difference is that the objective is to find first the similarities between individuals and then select a number of "k" persons that are the most similar. However, this approach hasn't been much used in real life because the user base is usually much bigger than the movie base (which can be verified even in the datasets used in our research) and because movie's similarities changes much slower compared to user's similarities (human tastes and preferences are more complex, volatile and unpredictable, whereas the similarity calculated between movies can remain stable for years) (Bell et al., 2007b).

However, the use of item-based KNN would be impossible with elevated scalability problem (Moreno, Segrera, López, Muñoz, & Sánchez, 2016). The application of KNN algorithm is also limited in the presence of big data since the computation requirement increases with the number of users and movies (Karypis, 2001). The usage of Item-KNN with a sparse matrix can disturb the quality of its recommendation (Z. Wang et al., 2018) and have a significant deterioration in its performance

[6] https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm

[7] BellKor's Pragmatic Chaos was the winner of Netflix Prize 2009. This team was composed by three teams: BellKor (Yehuda Koren, Robert Bell and Chris Volinsky), Big Chaos (Andreas Töscher and Michael Jahrer) and Pragmatic Theory (Piotte & Chabbert, 2009).

(Nikolakopoulos, Berberidis, Karypis, & Giannakis, 2019). We should keep this in mind when analysing the results of our study.

2.2. SINGULAR VALUE DECOMPOSITION (SVD)

Singular Value Decomposition (SVD) and SVD++ are some of the many variants of Matrix Factorization technique, which, in turn, is a widely used type of latent-factor models.

Latent-factor model is a family of methods which gained popularity for its usage in the Netflix Prize contest. It is an advancement made on neighborhood-based method because it tries to explain the ratings by characterizing both movie and user. Latent factors can be thought as the characteristics, for example, for the movies it could be its genre. It is something that is not explicit observed in the data, but inferred from the data, that is why is latent (the proper definition of latent is related to something that is present but not visible). So, the generation of latent factors for users and movies transforms both to the same latent factor space (H. Chen, 2017). Through dimensionality reduction, both representations became comparable and predictions on ratings can be made by multiplying those factors.

SVD represents the data in form of a matrix (where the rows are user IDs, the columns are movie IDs, and the space is filled by the ratings), which is reduced in simpler matrices (Hallinan & Striphas, 2016). As a matrix factorisation technique, it reduces the number of features of the data by reducing the space dimension from N-dimension to K-dimension (where K<N). The idea is that lower dimension could describe better the data. In fact, if we think of a real database in a matrix format, such as the Netflix database, we can imagine that the huge number of rows and columns needed to place all users and movies is excessive and redundant, besides not being practical to work with. Intuitively, we know that the information can be summarized, and we don´t need that huge dimension. For example, if a person gave 5-stars to *Star Trek* and to *Star Wars*, maybe that is not two separated information but one information about his preference. So, instead of representing our preference in terms of what movies I like or I dislike, we represent in terms of attributes. By doing so, it is possible to find similar individuals that have any movie in common, if they have the same preference in term of those attributes. These attributes (or features) not necessarily are understandable, but their importance is to have a predictive power.

So, SVD describes preference in terms of latent features that are learned from the data. The technique consists in decomposing this big matrix through factorization into the product of three new matrices of lower rank that represent user factors and movie factors: the user feature matrix (or user profile, which represents the relationship between users and latent factors. It represents how much each user appreciate a particular feature of the movie), the singular value matrix (which contains the relative importance of each factor for prediction), and the movie feature matrix (which indicates the similarity between movies and latent factors. It represents how much each movie manifest the same particular feature).

Those matrices carry some proprieties: two of them are orthogonal, they produce orthonormal bases, and they are dense, unlike the original matrix, which is sparse. It can be stated SVD algorithm

provides a procedure that factorize a matrix into singular vectors and singular values (Goodfellow et al., 2016). The relationship between the user and item is given by the inner product of their vectors.

As it was already indicated, there are studies that quote SVD as a method to handle sparsity and scalability problems (Moreno et al., 2016). This method was widespread during the Netflix Prize by Brandyn Webb, known as Simon Funk (Funk, 2006), who had shared the theorical background and the code used for the newly launched competition (Levene, 2010). The greatest contribution of Funk was to show a solution to compute SVD when the matrix had missing data. Since we never have a complete matrix - that's is, we don´t have a matrix where all user have given a rate for all movie - it was needed a way to fill the missing values so that it could be possible to apply factorization. This is done through an iterative method where the loss (cost) function can be minimized thought two main different optimization techniques: Stochastic Gradient Descent (SGD) and Alternating Least Squares (ALS).

SGD is one of the most popular algorithm used for optimization that computes, for each iteration, the error and then update the parameters by a factor in the opposite direction of the gradient, converging to local optima (Ruder, 2016), (Luhaniwal, 2019). ALS also works iteratively optimizing the parameters one by one while leaving the others fixed, minimizing the sum of squared residuals. Both were used by the winners of Netflix Prize.

This method had been refined during the progress prize, creating and implementing variations of it. As it was pointed out, SVD was one of the centrepieces of all algorithms used by competitors.

2.3. SVD++

As already studied, implicit feedback gives additional indication of user preferences. SVD++ algorithm is an improvement of SVD to incorporate implicit feedback (Xian, Li, Li, & Li, 2017). It was introduced by members of "BellKor" team (Bell, Koren, & Volinsky, 2009), and basically this method uses two matrices relatives to the movie, one with the explicit feedback and the other with the implicit feedback (Aggarwal, 2016). It was noticed that the inclusion of implicit feedback resulted in better prediction, and it turned to be the most accurate from the previous models tried on Netflix data (Koren, 2008).

The main idea resides on the fact that if the user chooses a moving picture to give an evaluation, despite which was the evaluation assigned, this is an implicit feedback. That is to say, the individual indirectly reveals his preferences by selecting which film he wants to give opinion. It was found that incorporating this binary data (where "1" means that a person rated a movie and "0" means that a reviewer did not rated it) improved the prediction accuracy considerably (Koren, 2008).

Implicit data are much more abundant than explicit data in most corporate databases (Steck, 2018). Since we don´t have access to data from companies of VOD and to better implicit measures such as user´s consumption or browsing history, SVD++ have an interesting and useful approach to include implicit feedback in this research.

2.4. RESTRICTED BOLTZMANN MACHINES (RBM)

RBM together with matrix factorization had the best performance measured by RMSE, according to the winners of Netflix Prize (Bell et al., 2007b). Both methods became more popular thanks to the successfully application during the competition. RBM also was considered to be a powerful technique to be used over large datasets, since it provided very accurate results (Abdollahi & Nasraoui, 2016).

RBM is an unsupervised machine learning method, coming from the deep learning family, methods based on artificial neural networks[8]. Neural networks are able to learn internal representation and underlying hidden factors, discovering complex correlation hidden in the data. RBM can learn to infer lower-dimensional representations automatically. The purpose is to model the statistical behaviour of data, looking for some distribution pattern (Louppe, 2010). One difference in relation to the previous methods studied here is that RBM discovers the probability distribution over the inputs, that´s it, the probability distribution of ratings and use it to prediction (Han, Wang, & Hong, 2017). Hence, the model makes possible to measure the confidence of the prediction made.

RBM is one of the simplest neural networks composed of stochastic[9] binary[10] units with symmetric connections and the layers (or the nodes) are divided in two types: a visible layer and a hidden layer (Hwang & Chen, 2017). Usually, the hidden layer represents the latent factors (example: feature of movies) and the visible layer represents the ratings of a user on a film (Truyen, Phung, & Venkatesh, 2009).

RBM is able to find the relationship between the movies and the relationship between the individuals, identifying the structure and relation in data. Once trained, RBMs can be used for predicting a user´s preference.

[8] https://en.wikipedia.org/wiki/Deep_learning

[9] When it has a random probability distribution or pattern that may be analysed statistically but may not be predicted precisely - https://www.lexico.com/definition/stochastic

[10] Each unit can have a state which can be *on* or *off*.

3. METHODOLOGY

The main idea behind the employed methodology is based in one Netflix's suggestion about presentation bias problem: by introducing randomness into the recommendations and learning better models (Gomez-Uribe & Hunt, 2015). Netflix stated that since the feedback loop is not considered in most of the standard recommendation´s techniques and in most of their own statistical model, there is room for the development of better algorithms that take it into account.

3.1. DATASETS

MovieLens is a recommender system developed in 1997 by GroupLens Research[11] that helps humans finds movies to watch[12]. GroupLens Research is a research lab in the Department of Computer Science and Engineering at the University of Minnesota and is a pioneer with projects in the field of recommender systems[13].

MovieLens datasets are heavily used in research and industry (Harper & Konstan, 2015). For example, if you had searched for “MovieLens” on Google Scholar on 01 of January of 2020, you would have found 17,8K results. MovieLens provides different sizes of datasets, with some difference relative to specific characteristics of the data content.

MovieLens datasets use the star rating as the measure of an individual opinion about some movie, in which 5-stars is the most favourable opinion. We can think the rating score as a measure that expresses the whole set of a movie attractiveness (genre, art, cast, plot, etc) from an individual perspective. That is, there is a razon why an individual gave 5-stars to a movie, and we can say that this value reflects some aspect of the movie which was enjoyed. Of course, the rating score also expresses characteristics of the individual itself as well, such as a person who is more severe in criticism, for example.

It is important to mentioned that although MovieLens datasets are considered confidents (Yang & Qiu, 2018), the disposal of the movies in its webpage is affected by search, filter an order mechanisms, which had an impact in the shape of datasets (Harper & Konstan, 2015). Until September 2008, MovieLens had four major releases that changed those mechanisms, but it was not clear if the datasets used in this study aggregate data coming from all these releases or just from some specific one. However, it was explicit that the webpage applied collaborative filtering and that the disposal of the movies were affected by the feedback loop coming from the rating given by the users (Harper & Konstan, 2015). Besides, there are sources indicating that MovieLens uses not only one but a variety of algorithms from collaborative filtering such as item-based and SVD[14].

Thus, although MovieLens datasets are composed by random selection of data, this does not eliminate the possibility that their inputs contain information relative to those recommendation mechanisms used in the webpage, and consequently, relative to presentation bias.

[11] https://en.wikipedia.org/wiki/MovieLens
[12] https://grouplens.org/
[13] https://en.wikipedia.org/wiki/GroupLens_Research
[14] https://en.wikipedia.org/wiki/MovieLens

For our analysis it was selected two different MovieLens databases.

3.1.1. MovieLens Latest Datasets - Small

The selected database met the criterion of the largest amount of data followed by the physical limitation of data processing by the computers available for this research.

It is used MovieLens Latest Datasets – Small (MovieLens Small), released in September 2008, which contain 100.836 ratings given by 610 users on 9.724 movies, measured in a 5-stars scale, in half-star increments (0,5 stars - 5,0 stars) (Harper & Konstan, 2015).

MovieLens Small database contains the ratings with the relative user IDs, movie IDs and the date of the rate (a variable called "timestamp"). This dataset is composed by a random selection of individuals that rated at least 20 movies, ensuring a minimum information about the users, and by films with at least 1 evaluation received.

In this research we ignore time dependence and make a simplifier assumption that the ratings do not change over time (neither the movie popularity nor the user preference change over time).[15]

3.1.2. Sample of MovieLens 1M Dataset

For the continuation of our research, we use a sample of the MovieLens 1M Dataset (Harper & Konstan, 2015). Originally this base, released in February 2003, contains 1 million ratings given by 6.040 users on 3.952 movies, measured in a 5-stars scale in whole increments (1 star - 5 stars). The dataset has the same structure as the previous one (user IDs, movie IDs, ratings and timestamp) and is composed by MovieLens by the same criteria reported above.

We subsample this database so that each movie was rated by at least 1.000 users. To achieve that goal, the sample was formed by the following rules: first, the movies that contain at least 1.000 users were selected. Then all user's IDs were detected, and it was counted the number of films evaluated by each person. What happens here is that now the minimum number of movies evaluated by user is no longer 20. But if we eliminate all individuals with less than 20 reviews, at the same time we reduce the number of movies with 1.000 consumers, and this process ends on a very reduced base. Therefore, what we did was to stop the elimination process until we reached a reasonable number: the minimum amount of reviews per user is 16 movies and the minimum amount of reviews per film is 912, ending with a total of 292.684 ratings (our sample is almost three times higher than MovieLens Small), with 4.820 users and 207 films.

[15] Besides, the variable "timestamp" wouldn´t be useful for this research. First because it corresponds to the date on which the rate was given and not to the consumption day of the movie. And second because the users tends to give many classifications in a single day, which means that it is difficult to interpret this variable, as it could be either an indication of the moment the user searched them or of the moment the movies were suggested to the consumers (Harper & Konstan, 2015).

This sample serve to conclude this study, since in the development of the research it was noticed that there was a constrain with MovieLens Small that distorted the results. This is further explained in Chapter 5, which is about the discussions regarding the outcomes achieved.

As MovieLens 1M sample was compiled with the intention of only selecting films with a large number of different evaluators, it will not be useful for diversity analysis since we discard large part of the dataset, specially the less popular movies, tending to select the most popular ones. The popularity here cannot be analysed either, as in the process of user´s elimination, we modify the popularity of all movies (including the most popular) without worrying about preserving their original popularity of the database.

3.2. RESEARCH STRATEGIES AND HYPOTHESIS

3.2.1. Analysis of recommendations

In the first part of our analysis, we focus on analysing how the number of recommendations per film are distributed. The purpose is to identify if there are any bias along the distribution concerning the presence of movies that, prominently, are much more suggested than others.

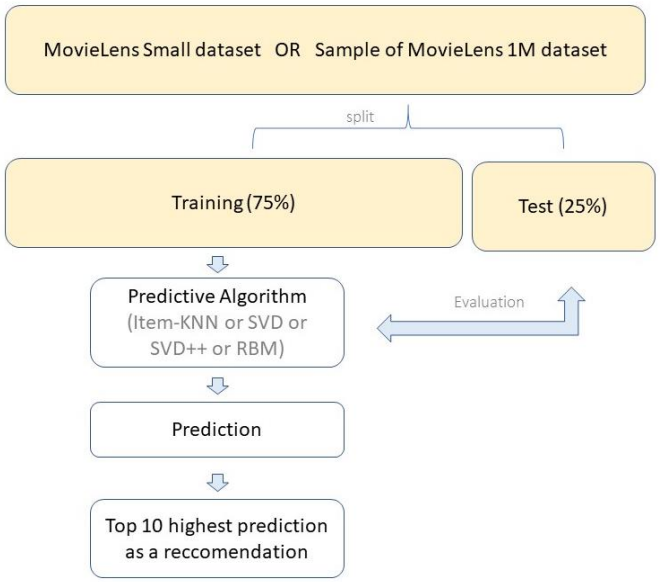


Figure 2 – The recommendation generation process with the original bases

We split the selected bases[16] at random into 75% for the training set and 25% for the test set. Once the algorithm is trained, it is used to predict the ratings of all movies that weren´t rated by each user and return the top 10 highest predictions for each of them as a recommendation. So, the first thing to be done is to generate recommendations for the original bases with each of the algorithms

[16] The selected original bases are: MovieLens Small dataset and the sample of 1M dataset.

selected, in order to obtain a benchmark. At the end of the process, we finish with four set of recommendations for each dataset (one using KNN algorithm, one employing SVD, one using SVD++ and one applying RBM model).

The research methodology adopted follow a stepwise approach, with three steps where each is marked by a different randomization procedure of the original database that generate three synthetic datasets. The main difference between the synthetic and the original datasets is that the firsts are generated programmatically and, therefore, artificially manufactured. The purpose of the synthetic data is to provide results (recommendations) that can be analysed and compared with the outcomes of the real data. The synthetic datasets mirror the real data by preserving some statistical properties and relationship of it. All synthetic datasets keep constant the total number of movies, of individuals, of ratings and the number of ratings by user of the original base. The objective of maintaining aspects of the original database is to make valid the comparison of the results. In other words, it is generated synthetic datasets where only one point is random, which is the point that is in the interest of the research, keeping the rest equal (Farine, 2017).

Once created the synthetic bases, each one goes through the same process the real base went through. At the end of the process, we finish with four set of recommendations for each synthetic dataset.

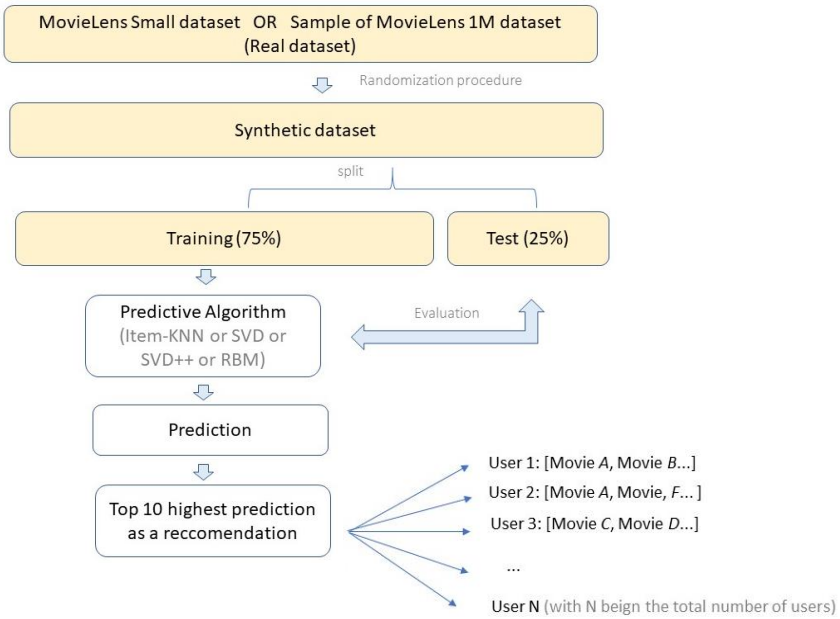


Figure 3 – The recommendation generation process with the synthetic datasets

With the results of each set of recommendation, it is counted the number of times each film was suggested. For instance: *Movie A* was indicated two times: one for *User 1* and another for *User 2*. In the extreme case, a film can be recommended for all users or none; and each algorithm can recommend all movies from the catalogue or only ten.

With this, it is constructed a frequency table with the number of recommendations per movie, with the number and the percentage of movies that obtained that amount of recommendations in relation to the total recommended.

The frequency table will be formed taking the number of recommendations as the data class. The total number of classes (and consequently the class size and intervals boundaries) is determined according to the *2 to k rule*[17]. Considering the number of classes, the total amplitude is calculated, and this value by the number of classes, so that the classes have the same range. Also, in order to facilitate the visualisation, the analysis can be incremented by plotting a histogram in which those classes will be on the horizontal axis and its frequency on the vertical axis. Thereby, we end up with an approximate representation of the distribution of the quantity of recommendations per movie.

That said, let's continue with the description of the stepwise approach. In the first step, it is analysed the recommendations coming from the original database. It is hoped here that it is confirmed the hypothesis that there are certain movies (which must be few in relation to the majority) that are clearly more suggested. In other words, we expect that the histogram (or the frequency table) is asymmetric (imbalance frequencies among the different classes).

In the second step we start with the randomization procedures. First, it is randomized the individual´s preference with the intention to analyse the behaviour of the algorithms when it is removed the user taste, but it is preserved the film popularity[18]. We consider the rating value (the number of stars) as a proxy of the individual preference. So, the randomization is done on the ratings given by user, which are going to be shuffled among the movies the same person had watched. This is the first stage in making the user's preferences more flexible: the consumer continues to watch the same shows, the rating pattern per user keeps the same[19], however, his preference (rating) for each movie is changed by those of others watched by him. Therefore, it is important to note that although the movie popularity is preserved, the total average of star-score received by it is modified, which is expected, because if we are changing people's tastes, the average rating of them can no longer be the same. In the example below, the average rating of *Finding Nemo* in the original data rating is 2,7-stars and change to 4,8-stars in the first synthetic dataset.

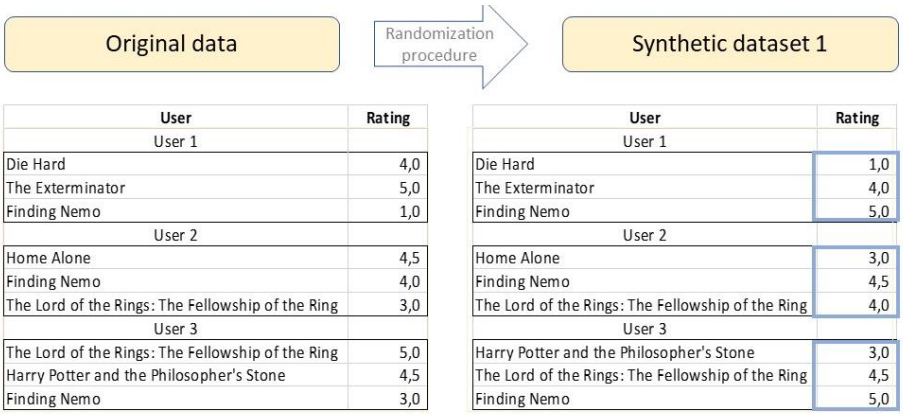
So, the expectation in this step is that there is a relevant difference between the frequency of recommendations coming from the original database to the suggestions coming from the first synthetic dataset. It is presumed that the new histogram (or the frequency table) have less

[17] The number of classes (k) is defined such as $2^k > n$, where k must be the smallest possible number and where n is the number of observations (in our case, the total number of recommended movie). For example, if it was suggested in total 207 films (n = 207), the number of k is 8, as 2^8 = 256 and 256 > 207.

[18] Regarding the popularity, it is measured by the percentage of total ratings received by movie. We are not going to consider the value of the rating itself to compute popularity, since it is difficult to evaluate the quality of a movie with a dataset that don´t have a uniform number of ratings by film, which it is illustrated on the next Chapter. For example, a movie with an average rate of 4,6-stars from 20 users shouldn´t be considered better than other with an average rate of 4,5-stars from 3.000 users. To work around this issue, some authors use a Weighted Rating measure that requires the determination of the minimum number of ratings received by a movie. Besides the fact that the determination of this value is harmful - since it would remove the films in the interest of our research, as the removed ones would not be the popular movies -, we do not have so much data to be willing to lose information, as it is also going to be shown on the next Chapter.

[19] We maintain the rating pattern per user in order to preserve some consistency and identity of themselves. For example: if a person tends to give a high star-score, he will continue with his profile.

asymmetry. If that is the case, the difference between them is considered as proxy of the bias caused by the presence of individuals' preferences.



User	Rating
User 1	
Die Hard	4,0
The Exterminator	5,0
Finding Nemo	1,0
User 2	
Home Alone	4,5
Finding Nemo	4,0
The Lord of the Rings: The Fellowship of the Ring	3,0
User 3	
The Lord of the Rings: The Fellowship of the Ring	5,0
Harry Potter and the Philosopher's Stone	4,5
Finding Nemo	3,0

User	Rating
User 1	
Die Hard	1,0
The Exterminator	4,0
Finding Nemo	5,0
User 2	
Home Alone	3,0
Finding Nemo	4,5
The Lord of the Rings: The Fellowship of the Ring	4,0
User 3	
Harry Potter and the Philosopher's Stone	3,0
The Lord of the Rings: The Fellowship of the Ring	4,5
Finding Nemo	5,0

Figure 4 – First randomization procedure

In the third step it is given a greater flexibility in the randomization of preferences, still maintaining the movie popularity. Now, the films classified by users are randomly drawn among all customers. That is to say, the rating pattern per user keeps the same, however, the shows watched by an individual can be any of the movies classified. As a result, if a film was seen by 25 reviewers on the original base, it continues to be seen by 25 individuals on the second synthetic dataset, but now by any 25 individuals. That is why the popularity remains the same. Here is valid the same comment made in the previous procedure: despite the popularity being preserved, the average star-score of each movie is changed.

User	Rating
User 1	
Die Hard	4,0
The Exterminator	5,0
Finding Nemo	1,0
User 2	
Home Alone	4,5
Finding Nemo	4,0
The Lord of the Rings: The Fellowship of the Ring	3,0
User 3	
The Lord of the Rings: The Fellowship of the Ring	5,0
Harry Potter and the Philosopher's Stone	4,5
Finding Nemo	3,0

User	Rating
User 1	
Finding Nemo	4,0
Harry Potter and the Philosopher's Stone	5,0
The Lord of the Rings: The Fellowship of the Ring	1,0
User 2	
The Exterminator	4,5
Finding Nemo	4,0
Die Hard	3,0
User 3	
Finding Nemo	5,0
The Lord of the Rings: The Fellowship of the Ring	4,5
Home Alone	3,0

Figure 5 – Second randomization procedure

So, in the previous step, we randomized the individual´s preference among their own choices. That is, each person continues to classify the same shows. Now in this step, we randomized the set of all

preferences among all the customers so that the concept of preference was expanded to cover the movie selection made by each individual. This is illustrated in the example above.

Again, the expectation in this step is that there is a relevant difference between the frequency of recommendations coming from the original data and coming from the second synthetic dataset. It is presumed that the new histogram (or the frequency table) have less asymmetry. If that is the case, the difference between them is considered as proxy of the bias that is caused by the presence of individuals' preferences.

The purpose of the four and last step is to evaluate the performance of the algorithms under the assumption that the individual's choices occur in a random way. The movie catalogue is sorted and drawn for each person. Here, again the rating pattern per user keeps the same, the individual continues to evaluate the same quantity of films, but now we randomize the movie popularity. In fact, we do more than that because with a random drawn of movies, we have as a result a symmetrical distribution of the number of evaluations received, reminding the shape of a normal distribution (we better visualise this in Chapter 4). In that sense, we eliminate the long-tail problem.

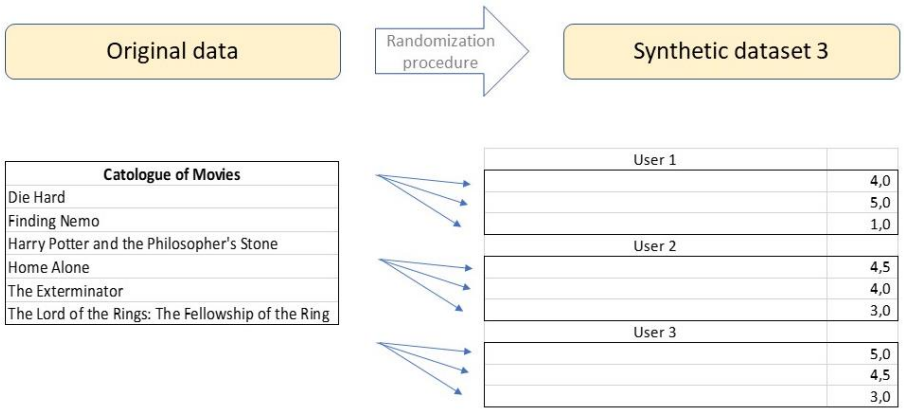


Figure 6 – Third randomization procedure

Therefore, the expectation in this step is that there will be a substantial difference between the frequency of recommendations coming from the original base and coming from the third synthetic dataset. It is expected that the algorithms should give random suggestions, that is, they should not be biased towards few movies (it should provide better results than the original base, in terms of bias). If the asymmetry persists under this step, it may suggest that there is something in the structure of the algorithms that made them tend to give a greater weight to a particular set of films than users would habitually do.

To recap, below are the databases used in the research.

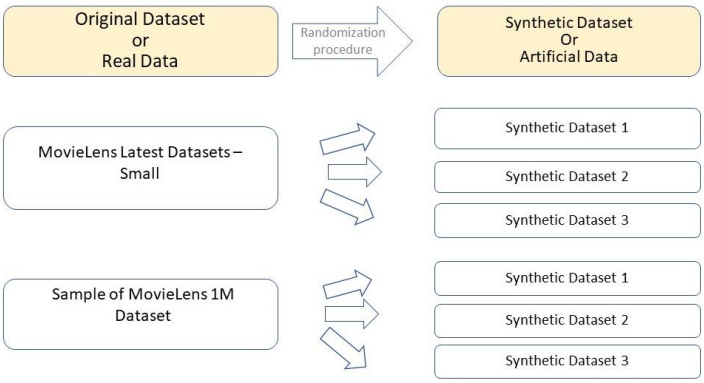


Figure 7 – All datasets

3.2.2. Analysis of RMSE

In the second part of our analysis, we will focus on analysing how the introduction of randomness at the base impacts the performance of the four algorithms. We will use RMSE as the measure of performance. The analysis will be made on the three synthetic datasets.

RMSE is the most common evaluation measure used in by recommendation algorithms (Cremonesi, Koren, & Turrin, 2010). During the machine the learning process (in which the training set is used), the algorithms learn about the data and, based on it, they predict the ratings for the test set. In this operation, the algorithms will seek to minimize the error (RMSE) between their prediction and the actual rating (test set).

Regarding the randomization procedure, it will be followed what was described in the previous section to form each synthetic base. In order to study how the introduction of randomness influences RMSE, we will gradually randomize the base and successively take note of the RMSE. For example, first we will randomize 10% of the original base according to the procedure described for the first synthetic base and record its RMSE. Then, 20% of the original base will be randomized following the procedure described for the first synthetic base and the RMSE will be noted. This will happen successively, with increments of 10%, until reach the percentage of 100%, a percentage that coincides with the synthetic base used in the first part of our analysis. The choice of the selected data (the percentage of the base) to be randomized will occur randomly. This procedure will be repeated for all algorithm.

Technically, by introducing randomness, we are removing some type of pre-existing bias in the original base, having as a result a base without this bias, which are the synthetic bases. Therefore, the objective of this part of the research is to evaluate whether the removal of the bias has an impact on the performance of the algorithm. With this we can also check if there is an algorithm more robust than another. That is, if there is an algorithm whose performance does not vary so much with the introduction of randomness. And we can also understand if more randomness in the data has a positive or negative effect over the model's performance.

The observation of the introduction of randomness, therefore, can be analysed by two ways. One way is to compare if there is a difference between 10% or 100% of the randomization of the base, for

example. And the other way is comparing the three synthetic bases (remember that the first procedure done at the base has less interference, the second synthetic base has a great disturbance than the first, and the third has the greatest intervention).

It is worth remembering that third synthetic base differs greatly from the others two in the randomization process, as the increase of randomization from 10% to 100% leads the movie´s rating distribution takes the shape of a symmetrical distribution. Also, it is worth remembering that the first and the second synthetic bases maintain the popularity of films, which means that they preserve the distribution in long-tail format.

3.2.3. Analysis of diversity

Regarding the diversity, two measures are going to be used to be evaluated:

- Coverage: recommended movies / total number of movies
- Diversity: (1-S), where "S" is the average between recommendation pairs. Explaining better, if we take into consideration the similarities scores of every possible pair of the recommended movies, these scores can be averaged in order to have a measure that express the similarity of the recommendations, which is "S". Thus, by subtracting "S" from 1, it is possible to have a measure related to the recommendation's diversity (Kane, 2019).

In each step, they are going to be calculated to evaluate if there was reduction or improvement on diversity. So, for all steps the expectation is that there will be a relevant difference in the results coming from the original and the synthetic dataset, in which is expected to be observed an improvement on the synthetic outcomes.

In this research, we use Python to run all the algorithms. This analysis does not take into consideration advertising payment for promotion of the movies as suggestions in the platform.

The following section describe the settings specifications of each selected algorithm.

3.3. DEFINITION OF PARAMETERS

We use Surprise library[20], which is Python package for recommender systems and TensorFlow library[21] an open source machine learning for Python.

Therefore, the model equations are established in advance by these libraries. In each respective library documentation can be found the research sources on which the equations were based.

[20] http://surpriselib.com/
[21] https://www.tensorflow.org/

Therefore, the use of these libraries brings us the security of research work done by professionals, in addition to the fact that they are public libraries and already experienced by the general public.

We remind you that in the literary review session, the reason for choosing and using each of these algorithms was explained. Therefore, this is section is reserved for definition of parameters.

We also recall that the purpose of this research is not to look for the most suitable SVD model, for instance, and fine tuning it. There are several SVD specifications used differently by each VOD company and even by one company itself. The work here is not to find a common sense model (even because we don't have access to them). The idea is to identify which are the used algorithms, which is the main idea of it, and then we can search this algorithm on an open library.

Surprise Library was designed based on MovieLens datasets and that is why we use this library to run all four algorithms. This make our job much easier and safe, since many of the parameters and default metrics were built based on the dataset used in this study.

3.3.1. K-nearest neighbor (K-NN)

As it was mentioned, the movie-based K-NN mainly focus on the similarities between movies. It is important to notice that in this step, different methods can be used to calculate similarity. In our study, the choice will be the same one taken by "Big Chaos" team: the Pearson correlation coefficient (Piotte & Chabbert, 2009). The Pearson correlation is a measure of the linear association between two variables and it obtained by dividing the covariance of both by the product of their standard deviations[22].

Surprise library offers four ways to calculate the KNN: the basic mode (the one that we choose), the KNN with means (which takes into account the mean ratings of each user), the KNN with Z Scores (which takes into account the z-score normalization of each individual) and the KNN baseline (which takes into account a baseline rating). Below is the formula of item-based model.

$$(i)\ \hat{r}_{ui} = \frac{\sum_{j \in N_u^k(i)} \text{sim}\,(i,j) \cdot r_{uj}}{\sum_{j \in N_u^k(i)} \text{sim}\,(i,j)}$$

Notation
i = reference movie
j = different movie from i
r_{uj} = rating of user on movie j
$\hat{r}_{ui}$ = predicted rating of user on movie i
i = reference movie

[22] https://en.wikipedia.org/wiki/Pearson_correlation_coefficient

j = different movie from i
r_{uj} = rating of user on movie j
$\hat{r}_{ui}$ = predicted rating of user on movie i
k = The (max) number of neighbors to take into account for aggregation. Default is 40
r_{ui} = rating of user on movie i
μ_i = average rating (of all users) on movie i
μ_j = average rating (of all users) on movie j
sim = similarity measure, which is going to be chosen the Pearson Correlation. $$\text{pearson\_sim}(i,j) = \frac{\sum_{u \in U_{ij}} (r_{ui} - \mu_i) \cdot (r_{uj} - \mu_j)}{\sqrt{\sum_{u \in U_{ij}} (r_{ui} - \mu_i)^2} \cdot \sqrt{\sum_{u \in U_{ij}} (r_{ui} - \mu_j)^2}}$$

3.3.2. Singular Value Decomposition (SVD)

This algorithm will be tested with the support of Surprise library. The model is based on the one proposed by members of "BellKor" team (Koren, Bell, & Volinsky, 2015). The minimization will follow the stochastic gradient descent method, which was also use by the winners of Netflix Prize (Toscher & Jahrer, 2009a).

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

Notation
$\hat{r}_{ui}$ = predicted rating of user on movie i
μ = average rating
b_u = bias of user from the average
b_i = bias of movie from the average
q_i = each item i is associated with a vector $q_i \in \mathbb{R}^f$, where is latent factor space of dimensionality f. For a given movie, the elements of q_i measure the extent to which the film possesses those factors, positive or negative.
p_u = each user u is associated with a vector $p_u \in \mathbb{R}^f$. For a given user u, the elements of p_u measure the extent of interest the individual has in items that are high on the corresponding factors, positive or negative.
The dot product captures the interaction between user u and item i and the user's overall interest in the item's characteristics.

3.3.3. SVD++

It is going to be used Surprise library to run this algorithm. The model follows the one proposed in the "Recommender Systems Handbook" (Ricci, Rokach, Shapira, Kantor, & Ricci, 2011), and the

optimization technique will be the stochastic gradient descent. The difference in the formula from SVD is that now the user is modelled by the term in parenthesis.

$$\hat{r}_{ui} = \mu + b_u + b_i + q_i^T \cdot \left(p_u + |I_u|^{-\frac{1}{2}} \sum_{j \in I_u} y_j\right)$$

Notation
I(u) is a set that contains the items rated by user u
y_i = captures the implicit ratings

3.3.4. Restricted Boltzmann Machines (RBM)

RBM model will be based on the one proposed in the book "Signal Processing and Networking for Big Data Applications" (Han et al., 2017), with a binary visible units and binary hidden units (Bernoulli or Boolean). The pair of Boolean vectors (v, h) is defined as:

$$E(v,h) = -\sum_i a_i v_i - \sum_j b_j h_j - \sum_i \sum_j v_i w_{ij} h_j$$

Or, in matrix notation as:

$$E(v,h) = -a^T v - b^T h - v^T W h$$

Notation
v = visible units (the number of nodes in the visible layer is defined as the number of items multiplied
h = hidden units (each hidden unit can learn to model the dependency between the ratings of different movies)
a = bias weights related to visible units
b = bias weights related to hidden units
W = matrix of weights associated with the connection between hidden and visible units, and the biases weights

4. EXPLORATORY DATA ANALYSIS

4.1. MOVIELENS LATEST DATASETS - SMALL

The rating-score histogram indicates a higher concentration of data in the highest rates rather than in the lowest, with 4-stars being the mode. The average rating is 3,5 stars (with standard deviation of 1,04) and is close to mode value.

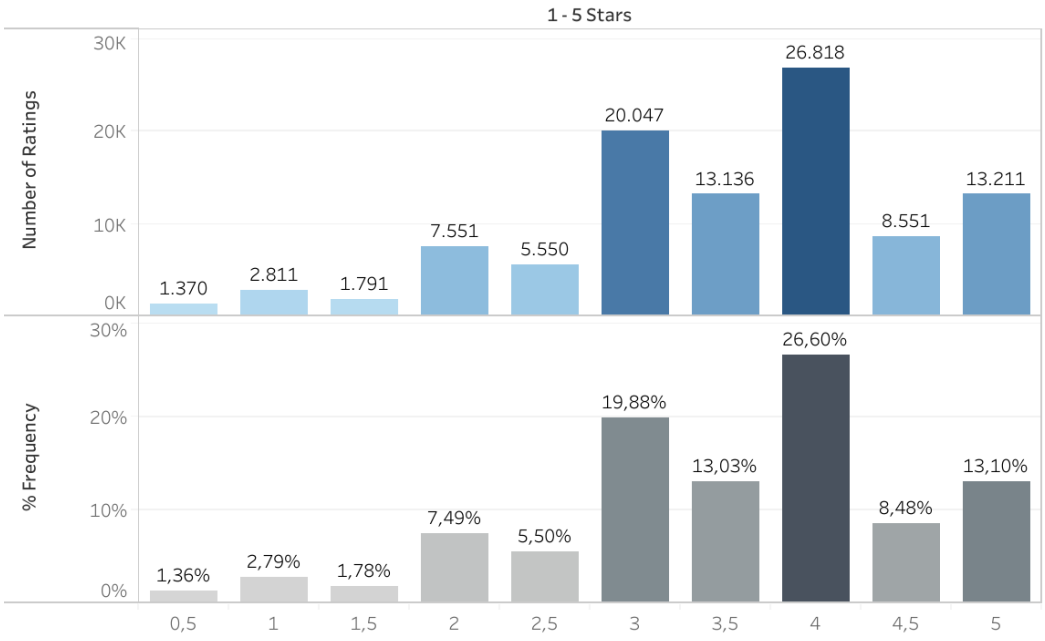


Figure 8 – Star rating histogram

If we plot the number of rating given by person and we count how many users gave that amount of reviews, we get the result below, where 2.698 is the maximum number of reviews given by an individual[23]. The fifth largest evaluator contributed with 1.346 ratings, what is almost half of 2.698. This express the size of the significant drop throughout the first positions, which means that few people rated this magnitude.

It is important to remember that the maximum number of rating that a reviewer can give is 9.724 (which is the total number of movies in the database) and the minimum is 20 (restriction imposed by MovieLens). Hence, the average number of reviews by user (165) certainly does not represent a good measure of central tendency because it is influenced by the upper outliers. We prefer to use the median, equal to 71, instead. That means that from 9.724 eligible movies, people tend to choose 71 to give an opinion, which indicates that our database suffers from a substantial reduction in the potential data that could be at the disposal of the research.

[23] Figure 8 can be read as follows: 14 individuals rated 20 films. 15 users evaluated 21 movies. And so on.

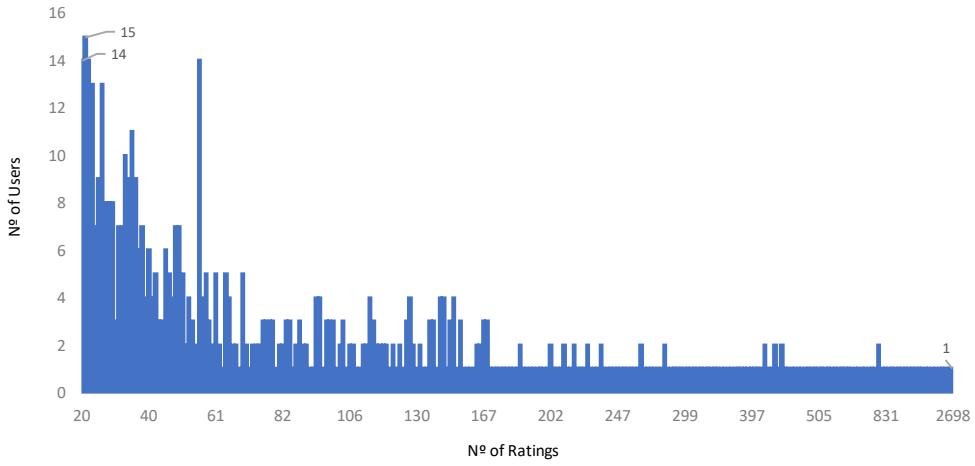


Figure 9 – Nº of ratings per user

Something similar occur with the number of ratings by movie. It can be observed that the majority of the films received few or only 1 review[24].

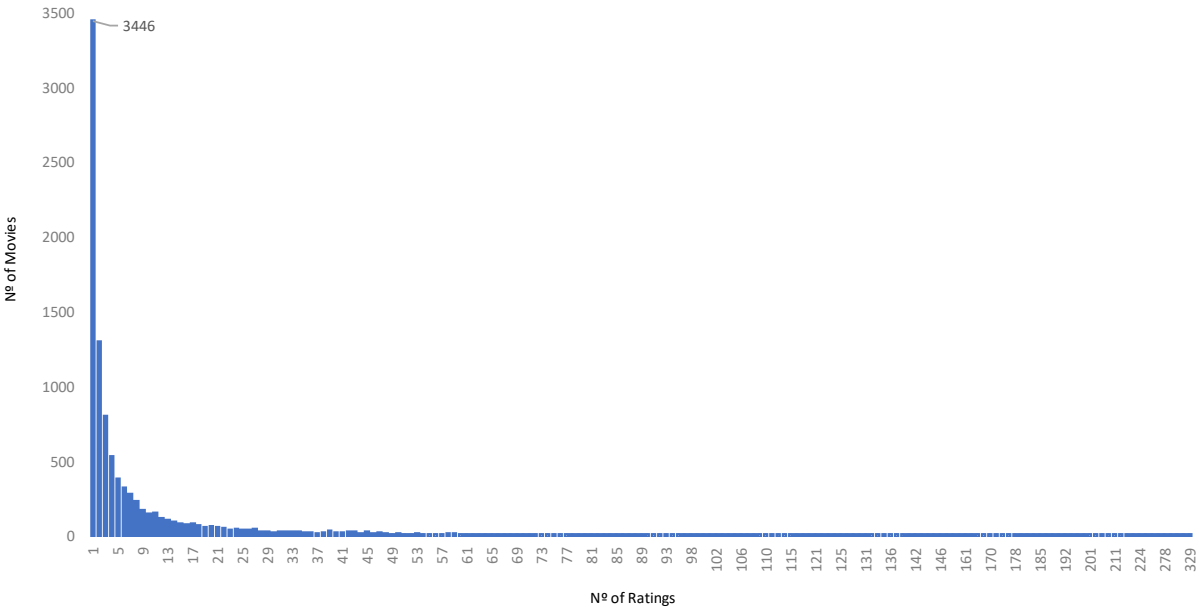


Figure 10 – Nº of ratings per movie

The most popular film of the base was reviewed by 54% or 329 of all individuals. Note that the maximum number of rating a movie can receive is 610 (which is the total number of individuals) and the minimum is 1 (MovieLens' restriction). This is to say, a little more than half of the people rated this film, even though the users' selection to compose the dataset was random. This number is more surprising if we remember that the reviewers of the dataset tend to select very few movies to

[24] Figure 10 can be read as follows: 3.446 movies had only 1 review.

evaluate, specifically 71 from 9.724 films. The median number of rating received by movie is 3 that is, from 610 available users to rate a movie, usually a film receives 3 votes.

You might be curious to know the popular movies, so below we printed the top 10.

Rank	Title	Nº of Ratings
1	Forrest Gump	329
2	The Shawshank Redemption	317
3	Pulp Fiction	307
4	The Silence of the Lambs	279
5	The Matrix	278
6	Star Wars: Episode IV - A New Hope	251
7	Jurassic Park	238
8	Braveheart	237
9	Terminator 2: Judgment Day	224
10	Schindler's List	220

Table 1 – Top 10 popular movies

This implies that there is a high probability that users have a tendency to select the popular films or a tendency that such movies had a higher prominence on MovieLens´ platform (as pointed out in Chapter 3).

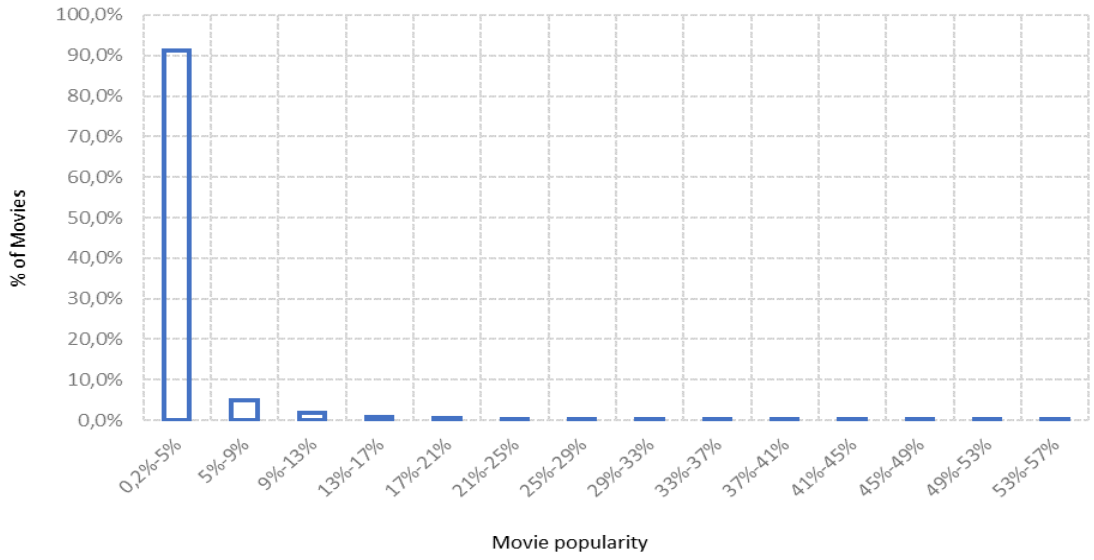


Figure 11 – Popularity distribution

The popularity of the movies falls rapidly as we move down the popularity list. If we select the movies that had a popularity greater than 30%, we only select 0,29% of all moving picture in the dataset (28 films). This reminds what we have studied in Chapter 2 as the long-tail problem. If we move further on that list, we observe that only 3,4% of all movies had popularity greater than 10%. In the figure above it can be noted that 91% of the movies were not popular, since they had at most only 5% of the users watching it. As we move to the other extreme of the chart, there are located the most popular movies of the dataset, which are very few compared to the non-popular ones.

So, the sparsity of MovieLens Small dataset seems to be high. As we studied before, sparsity is pointed out as a factor that deteriorates the performance of recommendation algorithms (S. Wang, Tang, Wang, & Liu, 2018). But sparsity is a very common problem faced both in the academic and real-world. In fact, there are many studies conducted with MovieLens dataset and sparsity is a constant problem present in all datasets. Our attention, once again, must reside in the fact that with sparsity, recommendation algorithms face limitations to suggest movies in the long-tail (Yin et al., 2012). It will be important to always keep this in mind when analysing the results of our research.

In Chapter 3, we created three synthetics datasets based on MovieLens Small. The first two synthetic datasets maintained constant the films' popularity. However, in Synthetic Dataset 3 the popularity of the movies was no longer preserved. As a result, we obtained a distribution that had a similar shape to a normal distribution.

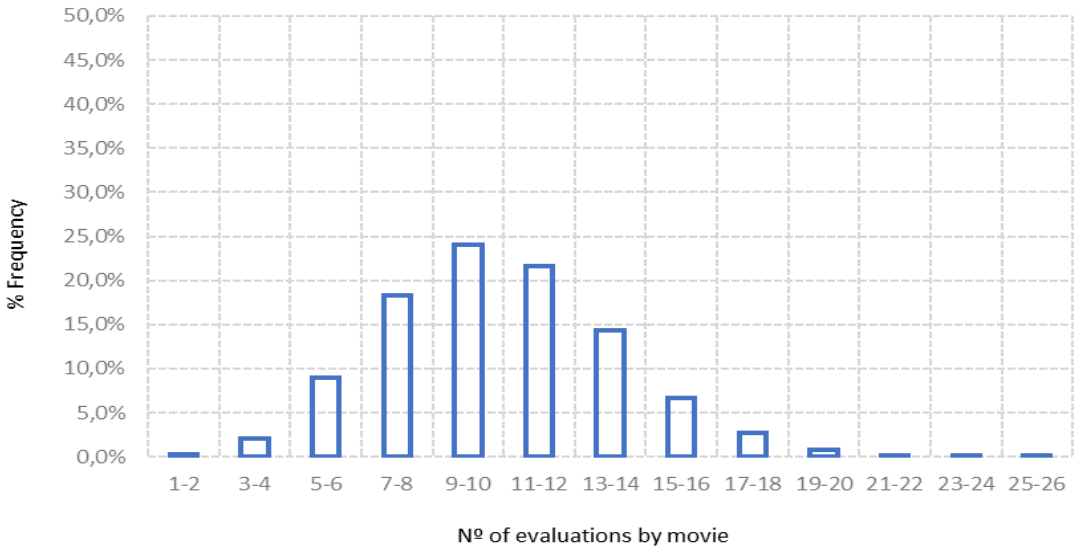


Figure 12 – Distribution of the nº rating by movie (Synthetic Dataset 3)

4.2. Sample of MovieLens 1M Dataset

The rating-score histogram of the dataset have a higher concentration of data in the highest scores, with 4-stars being the mode. The average rating is 3,9 stars (very close to the mode value) with standard deviation of 1-star.

In our sample there are relative few films with the worst score. As we selected the movies with a good relative number of reviewers, at first glance it could mean that in general there is a greater consensus among movies considered good than those considered bad. However, on a second thought we can compare the histogram of the whole MovieLens 1M dataset with our sample histogram and verify if there had been a significant change on it. It was verified that movies with 1-

star rating were already very few on MovieLens 1M dataset[25]. So, this indicates that in general people tend to rate more the films that they liked than those they didn't (this conclusion can be extended to the histogram of MovieLens Small dataset).

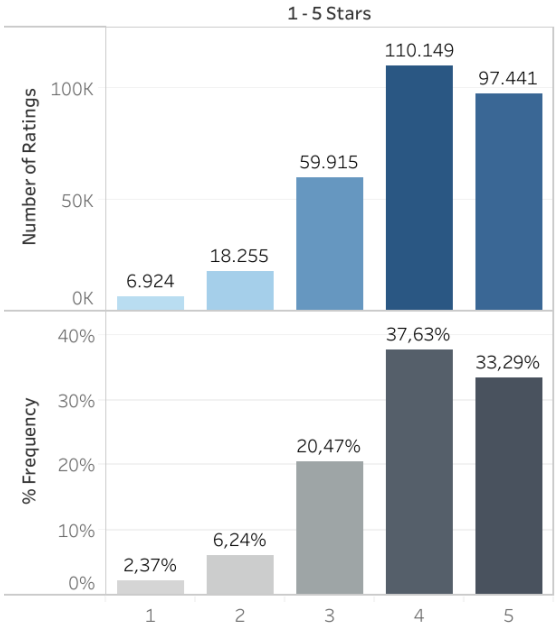


Figure 13 - Star rating histogram

The maximum number of rating given by person in this database was 203. Remember that the minimum number of movies that a person can choose to evaluate is 16 and the maximum is 207. Figure 8 shows a better homogenization between the number of classifications per user, revealing that there is a greater number of individuals with similar number of reviews. Hence, since 203 represents 98,1% of the 207 films at the disposal to be evaluated, we managed to make that an individual classified almost all the movies from our sample and also that a greater number of individuals selected a higher percentage of films from the catalogue to classification. So now, at the median, reviewers tend to choose 48 movies to give an opinion from 207 eligible ones.

[25] The frequency occurs in this sequence: 1.57% of the films receive 1-star, 6.0% of the movie have 2-star, 21.87% obtain 3-star, 38.97% reach 4-star and 31.59% earn 5-star.

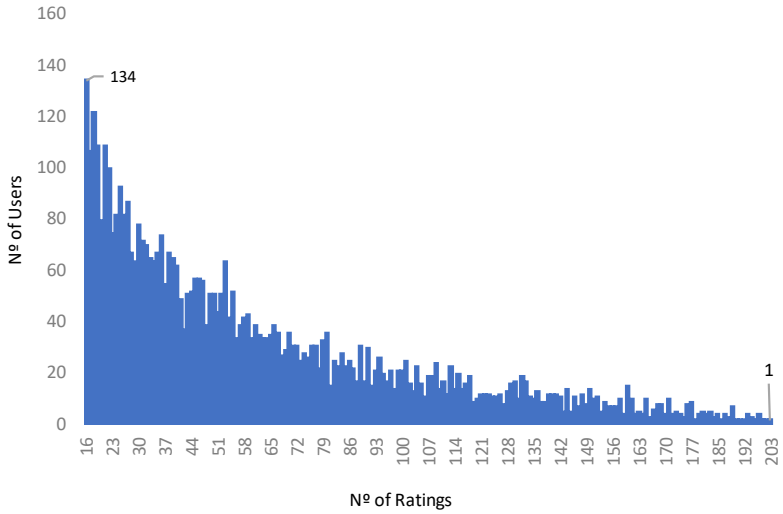


Figure 14 – Nº of rating per user

Regarding to the movies, the minimum number of rating a movie can receive is 912 and the maximum is 4.820 reviews. At the median, a movie received 1.295 classifications from 4.820 users (which is much better than the previous dataset). The maximum number of evaluations collect by a movie was 3.015, that is, 62,6% of the individuals of the sample rated it. It can be seen in the figure below that there is a greater homogenization between films in relation to their number of ratings.

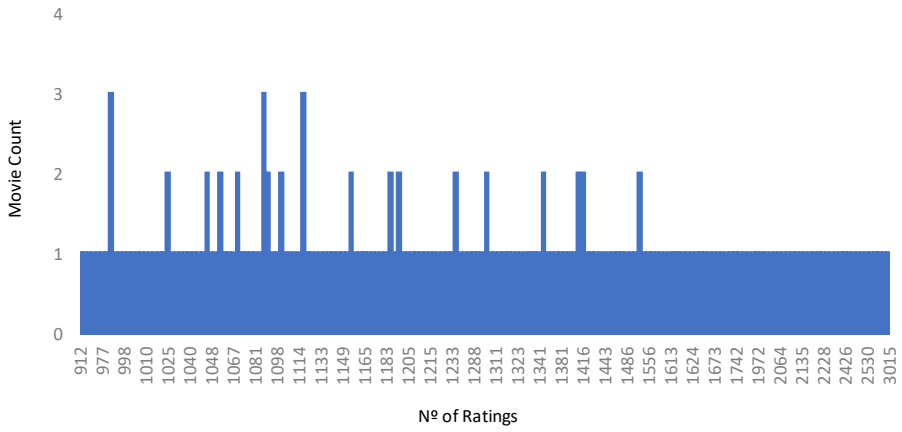


Figure 15 – Nº of rating per movie

Below we printed the top 10 popular films:

Rank	Title	Nº of Ratings
1	American Beauty	3.015
2	Star Wars: Episode V - The Empire Strikes Back	2.854
3	Star Wars: Episode IV - A New Hope	2.832
4	Star Wars: Episode VI - Return of the Jedi	2.666
5	Terminator 2: Judgment Day	2.532
6	The Matrix	2.530
7	Saving Private Ryan	2.488
8	Jurassic Park	2.483
9	Back to the Future	2.460
10	Men in Black	2.446

Table 2 – Top 10 popular movies

If we select the movies that had a popularity greater than 30%, we pick 31,9% of all in the catalogue (which is terrific better than MovieLens Small). In this sample all the films have a popularity greater than 18,9%, which is a huge improvement comparing with MovieLens Small where only 1,0% of them had a popularity greater than 18,9%). But we still have a long-tail distribution as can see in the figure below. We have few popular movies compared with the least popular ones (almost 45% of them had a popularity between 18% and 25%).

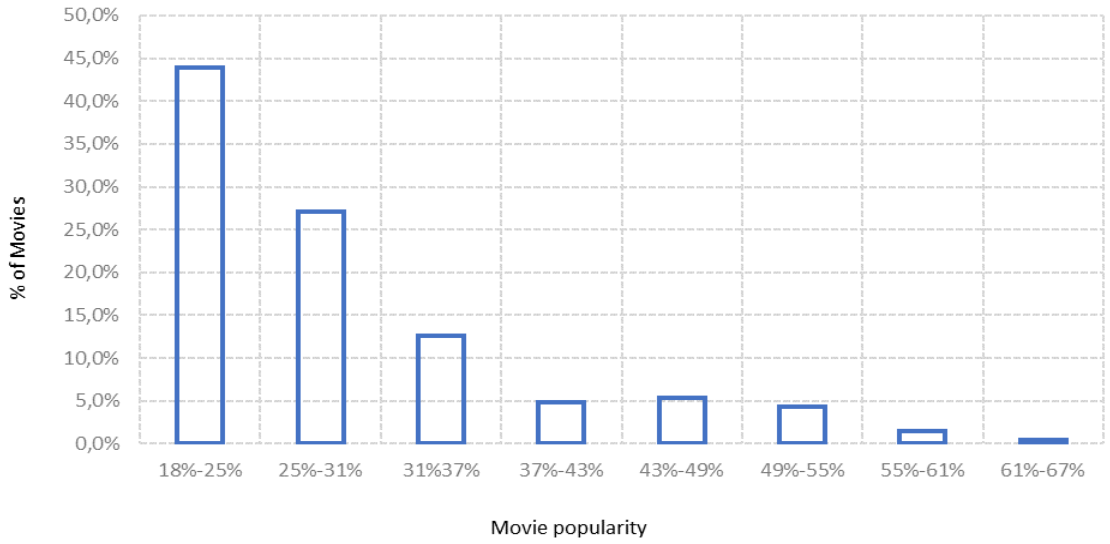


Figure 16 – Popularity distribution

Another interesting thing is to look at the distribution of the number of ratings per movie in our sample. That is, we have a relative high frequency of films located in the class range with the least number of ratings (where are located the least popular movies). At the other extreme, we have a low concentration of the popular ones. In Chapter 5, we revisit this graph and compare it with the results of the recommendations.

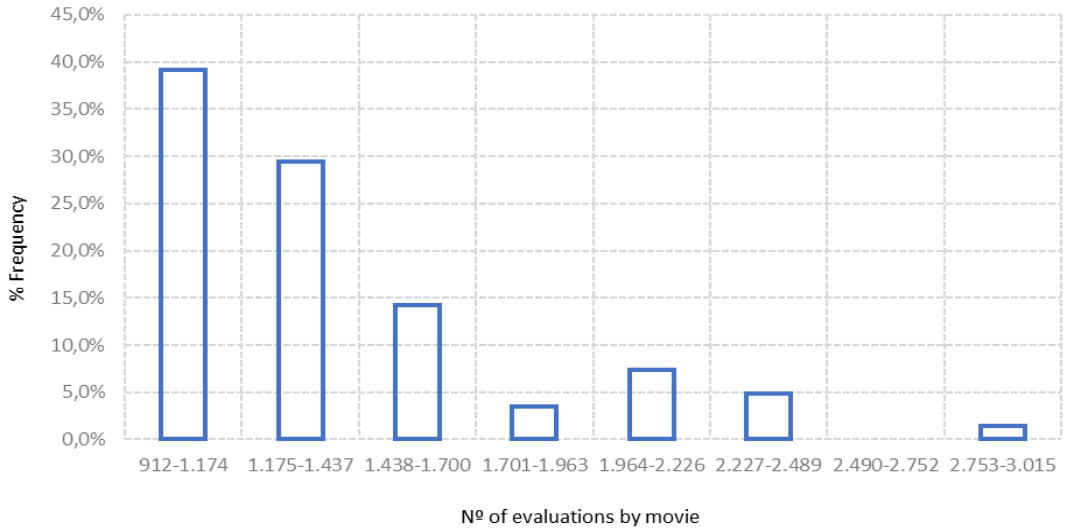


Figure 17 – Rating distribution by movie

Finally, we see how the graph above was modified when the last randomization procedure was applied on our MovieLens 1M sample. That is, Synthetic Dataset 3 had a shape very similar to a normal distribution. Here it is eliminated the long-tail problem. The greater amount of the evaluations is in the middle classes. So, neither the popular films (located in the last interval) or the least popular movies (situated in the first range) concentrate a high number of reviews. It is also good to note that the total amplitude has been reduced, that is, now the minimum number of ratings that a film can have is 1.310 votes and the maximum number is 1.524, which implies that there is less disparity between the films that are in different classes, in terms of number of evaluations. In Chapter 5 it will be important to remember this main difference between the datasets.

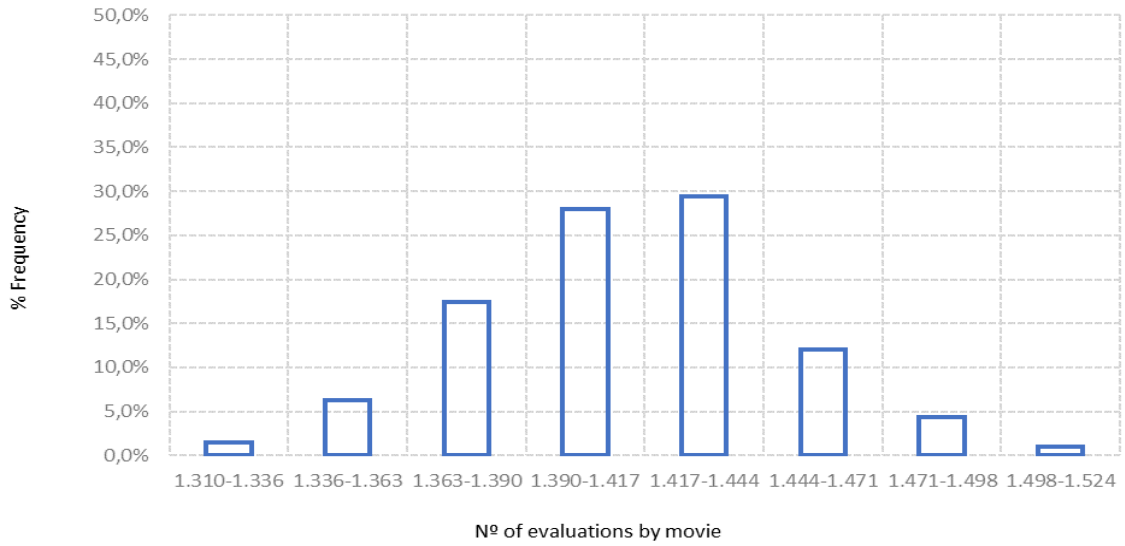


Figure 18 – Distribution of the nº rating by movie (Synthetic Dataset 3)

5. RESULTS AND DISCUSSION

5.1. ANALYSIS OF RECOMMENDATIONS

5.1.1. MovieLens latest dataset – small

Below we can see the results of each algorithm run on the original base. All algorithms generated 10 recommendations by users, totalling 6.100 recommendations. The tables can be read as follows (we will take the Item-KNN as an example):

- 1.044 movies were recommended between 1 to 9 times;
- 86,1% of the recommendations were between 1 to 9 users;
- In total, 1.212 films were suggested;
- The maximum number of times (users) that a movie was recommended was 92 and the minimum was one[26].

Nº of recommendations	Amplitude	Nº of movies	% of recommendations
1-9	**9**	1.044	86,1%
10-18	**9**	97	8,0%
19-27	**9**	41	3,4%
28-36	**9**	16	1,3%
27-45	**9**	6	0,5%
46-54	**9**	4	0,3%
55-63	**9**	1	0,1%
64-72	**9**	1	0,1%
73-81	**9**	0	0,0%
81-90	**9**	1	0,1%
91-92	**9**	1	0,1%
Total		**1.212**	**100,0%**

Table 4 - Item-KNN: Frequency distribution table

Nº of recommendations	Amplitude	Nº of Movies	% Frequency
1-26	26	351	83,80%
27-52	26	34	8,10%
52-78	26	19	4,50%
79-104	26	6	1,40%
105-130	26	4	1,00%
131-156	26	2	0,50%
157-182	26	1	0,20%
183-208	26	1	0,20%
209-229	26	1	0,20%
Total		**419**	**100,00%**

Table 3 - SVD: Frequency distribution table

All algorithms had one thing in common in their outcomes: a high concentration of films in the smallest range of recommendations and practically a null frequency in the highest ranges of recommendations. This suggests that there was a bias in all outputs as there was a tendency for few movies to be suggested several times and a tendency for the vast majority of movies to have a low number of recommendations. Briefly, all algorithms presented an asymmetric right-skewed

[26] The upper boundary of the last class always is reported with the highest number of recommendations. This is done to make it easier to visualize what was the maximum number of suggestions obtained by a movie, since this number is more important to us than the value of the upper boundary itself. If you want to know the real upper boundary of the class, just add the lower-class interval with its amplitude.

distribution[27]. So, we confirmed our research's expectation: there are some movies that have an anomalous vantage among other in terms of number of recommendations. In other words, it was found evidence of presentation bias.

Nº of recommendations	Amplitude	Nº of Movies	% Frequency
1-43	43	334	91,0%
44-86	43	19	5,2%
87-129	43	6	1,6%
130-172	43	2	0,5%
173-215	43	2	0,5%
215-258	43	2	0,5%
259-301	43	0	0,0%
301-344	43	1	0,3%
345-381	43	1	0,3%
Total		**367**	**100,0%**

Table 6 - SVD++: Frequency distribution table

Nº of recommendations	Nº of Movies	% Frequency
1	3.352	73,1%
2	1.002	21,8%
3	197	4,3%
4	33	0,7%
5	3	0,1%
6	1	0,0%
Total	**4.588**	**100%**

Table 5 - RBM: Frequency distribution table

Of the four tested algorithms, SVD++ was the one that achieved the maximum number of recommendations that a film could have, equal to 381, number that exceeded the maximum number of evaluations reached by a movie in the original base (in Chapter 4 we showed that the most popular movie had 329 votes). So, through this view, SVD++ was the one with the greater disparity in respect of the films that were more or less suggested. SVD++ also was the one with the highest concentration of movies in the smallest range of recommendations and by far the one with the lowest coverage, equal to 3,8%.

The result of RBM must be seen separately, as it was the most different of all algorithms. We can say that this algorithm, in a way, limited the disparity between the number of recommendations by establishing a very small range in which all movies were suggested: from 1 to a maximum of 6 times. It was as if this algorithm had all the recommendations in the first class of the others. Due to that, we could view the outputs discreetly, without the need to group them by class[28] . We can say that this algorithm presented the most standardized (least biased) results since 94,9% of the films were suggested 1 or 2 times and 99,2% were indicated 1 to 3 times. In other words, there was no significant advantage between the movies, that being understood as its appearance on the screen device. RBM was also, by far, the algorithm that had the highest coverage, equal to 47,5%.

Now we analyse the outputs of each algorithm run on Synthetic Dataset 1.

Contrary to our expectations, the introduction of randomness in the database only reduced the bias in one of the four algorithms: Item-KNN. The percentage of movies with a relative low number of

[27] A right-skewed distribution is a distribution whose tail is longer on the right and in which the crowd of data is concentrated on the left at the lower scales.

[28] That is why the "Amplitude" column is no required here.

recommendations had decreased from 86,1% to 73,7%. However, there was a worsening of coverage from 12,5% to 8,3%.

We can say that RBM did not exhibit a relevant difference. Concerning to SVD and SVD++, however, both had a considerable increase in the percentage of movies with low recommendation and, at the same time, an increase in the maximum number of times that a movie was suggested. That is, there was a worsening in the discrepancy between films with low and high number of recommendations. An expressive majority of movies was at a disadvantage compared to a minority of them. However, there was an increase in coverage.

Therefore, the expectation that the first randomization procedure should induce a reduction in the asymmetry was only confirmed for Item-KNN. For SVD and SVD++ the opposite was true: there was a greater asymmetry between the frequency along the interval-classes and there was a greater discrepancy between the number of recommendations.

Nº of recommendations	Amplitude	Nº of Movies	% Frequency
1-8	8	593	73,7%
9-18	8	83	10,3%
19-24	8	54	6,7%
25-32	8	40	5,0%
33-40	8	15	1,9%
41-48	8	11	1,4%
49-56	8	2	0,2%
57-64	8	2	0,2%
65-72	8	1	0,1%
72-76	8	4	0,5%
Total		**805**	**100,0%**

Table 7 - Item-KNN: Frequency distribution table

Nº of recommendations	Amplitude	Nº of Movies	% Frequency
1-32	32	1.070	97,4%
32-64	32	18	1,6%
65-96	32	2	0,2%
97-128	32	3	0,3%
129-160	32	1	0,1%
161-192	32	2	0,2%
193-224	32	1	0,1%
225-256	32	0	0,0%
257-288	32	0	0,0%
289-320	32	0	0,0%
321-348	32	2	0,2%
Total		**1.099**	**100,0%**

Table 8 - SVD: Frequency distribution table

Nº of recommendations	Amplitude	Nº of Movies	% Frequency
1-49	49	579	96,2%
50-98	49	11	1,8%
99-147	49	3	0,5%
148-196	49	1	0,2%
197-245	49	1	0,2%
246-294	49	3	0,5%
295-343	49	2	0,3%
344-392	49	0	0,0%
393-441	49	0	0,0%
441-486	49	2	0,3%
Total		**602**	**100,0%**

Table 10 - SVD++: Frequency distribution table

Nº of recommendations	Nº of Movies	% Frequency
1	3.285	72,1%
2	1.019	22,4%
3	225	4,9%
4	23	0,5%
5	2	0,0%
Total	**4.554**	**100%**

Table 9 – RBM: Frequency distribution table

Now let´s look the results of each algorithm run on Synthetic Dataset 2.

Item-KNN seemed to have an improvement since there was a greater reduction in the maximum number that a film was recommended and since there was a decrease in the percentage of films with less recommendation. The drop of coverage in relation to the original base (from 12,5% to 11,1%), now was smaller than the drop noticed in Synthetic Dataset 1.

SVD and SVD++ showed a slightly better result than Synthetic Dataset 1, but still worse than the original base. RBM´s results did not indicate a significant difference compared to previous outcomes.

Again, Item-KNN was the only algorithm whose result confirmed our expectation. Again, SVD and SVD++ outcomes were opposite to our expectations since they exhibited a greater skewness.

Nº of recommendations	Amplitude	Nº of Movies	% Frequency
1-5	5	748	69,3%
6-10	5	146	13,5%
11-15	5	78	7,2%
16-20	5	52	4,8%
21-25	5	29	2,7%
26-30	5	14	1,3%
31-35	5	5	0,5%
36-40	5	4	0,4%
41-45	5	1	0,1%
46-50	5	2	0,2%
51-51	5	1	0,1%
Total		**1.080**	**100,0%**

Table 11 – Item-KNN: Frequency distribution table

Nº of recommendations	Amplitude	Nº of Movies	% Frequency
1-19	19	972	93,6%
20-38	19	36	3,5%
39-57	19	14	1,3%
58-76	19	8	0,8%
77-95	19	3	0,3%
96-114	19	0	0,0%
115-133	19	1	0,1%
134-152	19	1	0,1%
153-171	19	2	0,2%
172-190	19	1	0,1%
191-208	19	1	0,1%
Total		**1.039**	**100,0%**

Table 12 – SVD: Frequency distribution table

Nº of recommendations	Amplitude	Nº of Movies	% Frequency
1-48	48	555	95,4%
49-96	48	9	1,5%
97-144	48	11	1,9%
145-192	48	2	0,3%
193-240	48	2	0,3%
241-288	48	0	0,0%
289-336	48	0	0,0%
337-384	48	1	0,2%
385-432	48	1	0,2%
433-475	48	1	0,2%
Total		**582**	**100,0%**

Table 14 - SVD++: Frequency distribution table

Nº of recommendations	Nº of Movies	% Frequency
1	3.191	70,8%
2	1.070	23,8%
3	213	4,7%
4	26	0,6%
5	4	0,1%
6	1	0,0%
	4.505	**100%**

Table 13 - RBM: Frequency distribution table

Finally, let´s check the results of the algorithms applied in Synthetic Dataset 3.

To our surprise, the outcome of Item-KNN, SVD and SVD++ on Synthetic Dataset 3 were worse than the results coming from the original base. These outputs suggested that the algorithms tend to skew the number of recommendations towards few movies. The removal of randomness of the original base did not favour to reduce the observed bias. On the contrary, it reinforced the bias if we consider that Synthetic Dataset 3 had a symmetrical distribution of the number of ratings by movie. Thus, the algorithms ended up generating a bias that was not present in the original shape of the data.

Nº of recommendations	Amplitude	Nº of Movies	% Frequency
1-7	7	1.595	90,6%
8-14	7	86	4,9%
15-21	7	29	1,6%
22-28	7	17	1,0%
29-35	7	9	0,5%
36-42	7	5	0,3%
43-49	7	5	0,3%
50-56	7	6	0,3%
57-63	7	2	0,1%
64-70	7	2	0,1%
71-74	7	4	0,2%
Total		**1.760**	**100,0%**

Table 16 - Item-KNN: Frequency distribution table

Nº of recommendations	Amplitude	Nº of Movies	% Frequency
1-42	42	640	95,2%
43-84	42	12	1,8%
85-126	42	10	1,5%
127-168	42	2	0,3%
169-210	42	3	0,4%
211-252	42	0	0,0%
253-294	42	2	0,3%
295-336	42	0	0,0%
337-378	42	1	0,1%
379-419	42	2	0,3%
Total		**672**	**100,0%**

Table 18 - SVD++: Frequency distribution table

Table 15 – SVD: Frequency distribution table

Nº of recommendations	Amplitude	Nº of Movies	% Frequency
1-22	22	1.022	95,4%
23-44	22	30	2,8%
45-66	22	8	0,7%
67-88	22	3	0,3%
89-110	22	3	0,3%
111-132	22	1	0,1%
133-154	22	1	0,1%
155-176	22	1	0,1%
177-198	22	0	0,0%
199-220	22	0	0,0%
221-242	22	2	0,2%
Total		**1.071**	**100,0%**

Nº of recommendations	Nº of Movies	% Frequency
1	3.232	71,4%
2	1.044	23,1%
3	216	4,8%
4	29	0,6%
5	2	0,0%
6	1	0,0%
Total	**4.524**	**100%**

Table 17 – RBM: Frequency distribution table

This maybe was because the introduction of randomness in 100% of the database had worsen the model's ability to make good recommendations. But the unexpected outcomes made us reflect on what else could be wrong and we remembered an important fact. In Chapter 3, it was demonstrated that all the selected algorithms took into account the ratings values: the rating was the only measure used relative to the users' opinion about a film. Also, in Chapter 3 we saw that at least three of them (Item-KNN, SVD and SVD++) used the ratings average in their calculations. This can be a problem if we go back to Chapter 4 (Figure 10) and remember that from 9.724 films, 3.446 (35,4%) had only one review and that 1.298 (13,3%) had two evaluations. That is, almost 50% of movies had one or two individuals evaluating it.

This is much more a problem regarding the number of observations than a sparsity problem. We saw in Chapter 2 that sparsity occurs when there are movies with not enough evaluations. We also saw that collaborative filtering methods serve to address sparsity problem. However, one thing is not having enough information, and another very different is a scenario of practically absolute lack of information. In our view, it cannot be done a reliable research with half of the movies being rated by only one or two persons. In addition, we must remember that in Chapter 2 it was mentioned that performance of Item-NN worsened a lot under the presence of too much sparsity, which would lead us any way to invalidate its results.

What all these algorithms do when they are working is to consider the average rating as an indicative of the “true” score of the movie. In theory, when we have a large sample of people, the average rating approaches in probability that “true” score. When we have only one vote, we have a lot of uncertainty in relation to the score of the film, that is, if that review represents the real evaluation of the movie or only the opinion of a single person, which, in turn, may differ from most people's opinion. And this problem the algorithms are not taking into account. Therefore, as our base is composed of many movies that may be far from their “true” rating. This may have been the reason why the algorithms performed so differently than expected.

However, we also should remember that our database already contained a small number of data, so if we simply select the movies films with a minimum number of ratings, we will have a very limited dataset for research purposes. The solution found was to use a sample of a larger base from MovieLens, since the only reason we didn't use them was that they were too big for the computers available for this research. But we will no longer have this problem if we use a small sample of one of those larger bases. So, the next step will be to reapply the algorithms in the selected sample and check the outcomes.

5.1.2. Sample of MovieLens 1M dataset

All algorithms generated 10 recommendations by users, totalling 48.200 recommendations.

Let´s start looking the results on the original sample and first step of the analysis.

Nº of recommendations	Amplitude	Nº of Movies	% Frequency
23-119	97	55	26,6%
120-216	97	60	29,0%
217-313	97	39	18,8%
314-410	97	26	12,6%
411-507	97	14	6,8%
508-604	97	5	2,4%
605-701	97	4	1,9%
702-794	97	4	1,9%
Total		**207**	**100%**

Table 19 - Item-KN: Frequency distribution table

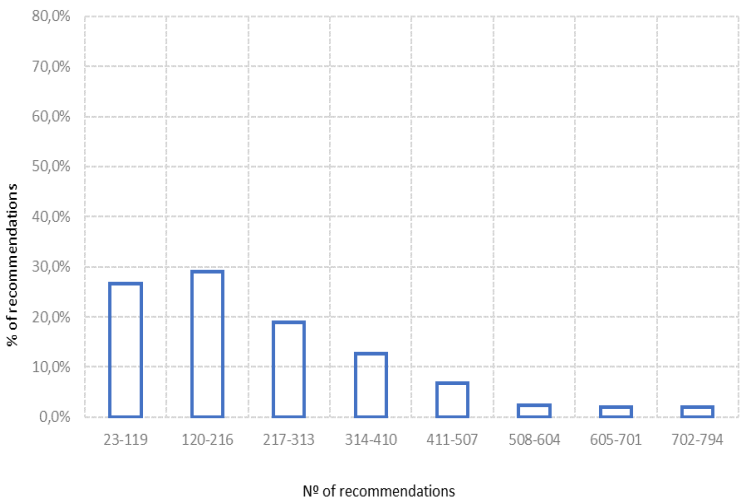


Figure 19 - Item-KNN: Histogram (original data)

SVD			
Nº of recommendations	**Amplitude**	**Nº of Movies**	**% Frequency**
1-265	265	149	74,1%
266-530	265	24	11,9%
531-795	265	12	6,0%
796-1.060	265	5	2,5%
1.061-1.325	265	4	2,0%
1.326-1.590	265	4	2,0%
1.591-1.855	265	1	0,5%
1.856-2.115	265	2	1,0%
Total		**201**	**100%**

Table 20 - SVD: Frequency distribution table

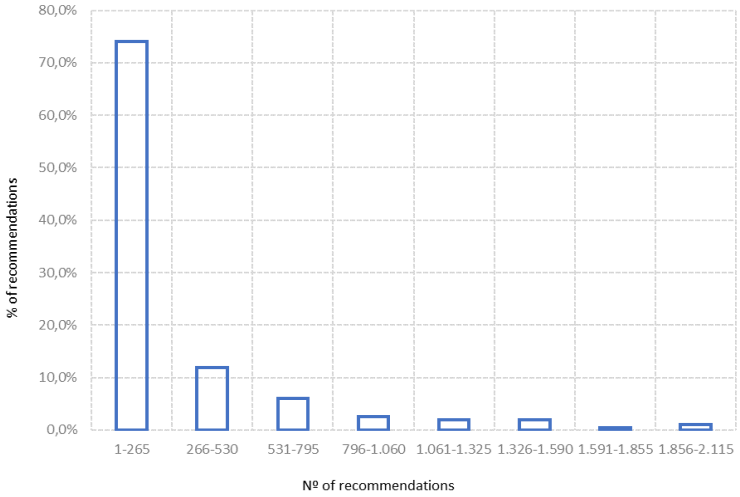


Figure 20 - SVD: Histogram (original data)

Compared with the previous section, now it can be noticed a huge differentiation between the algorithms' outputs coming from the original base. Although each algorithm produced a distinct outcome, it was confirmed the expectation stated in Chapter 3 for Item-KNN, SVD and SVD++. Namely: it was observed a relevant asymmetry between the frequency of its histograms, which indicated that few movies were more recommended than most films.

Item-KNN had a bias for the first two ranges of recommendations, especially if compared to the latest ranges. Item-KNN and RBM had 100% of coverage, evidencing that all movies from catalogue were suggested. For RBM we had the opposite of the expectation stated in Chapter 3.

We concluded that SVD and SVD++ were the algorithms that most favored a bias in the recommendations. Both presented a very high concentration in the first class of recommendation (which was the range with the movies with smaller visibility). The other classes exhibited low or

almost null concentration, especially the latest interval (which was the band with the movies better positioned). Also, both models where the ones that generated the greatest discrepancy between the number of times that movies were recommended. For example, there was a movie suggested for one person at the same time that there was another indicated for 2.318 users. These models had the least coverage (SVD with 97,1% and SVD++ with 97,6%), although now all algorithms expressed an excellent level of coverage.

Nº of recommendations	Amplitude	Nº of Movies	% Frequency
1-290	290	156	77,2%
291-580	290	16	7,9%
581-870	290	15	7,4%
871-1.160	290	4	2,0%
1.161-1.450	290	6	3,0%
1.451-1.740	290	2	1,0%
1.741-2.030	290	0	0,0%
2.031-2.318	290	3	1,5%
Total		**202**	**100%**

Table 21 - SVD++: Frequency distribution table

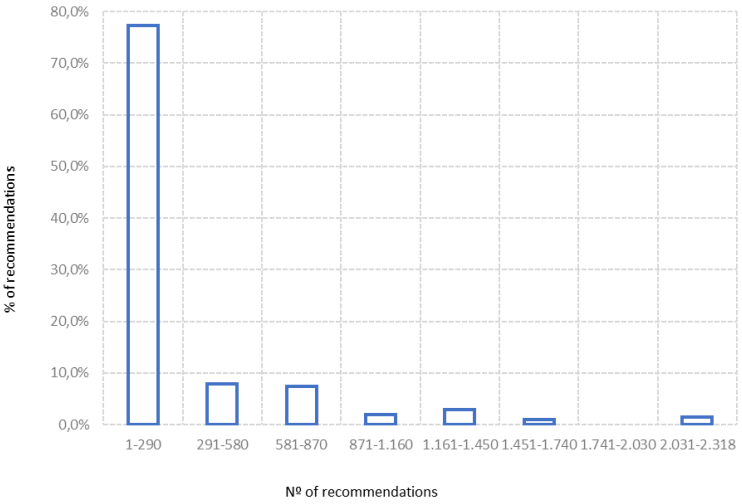


Figure 21 – SVD++: Histogram (original data)

RBM was the algorithm that presented the most differentiated result. First of all, a movie was not suggested for less than 111 people, which was an excellent result. It was also the algorithm with the lowest maximum number of recommendations, and it was by far the one with a minor discrepancy between the number of recommendations achieved by movies. In the extreme case, there was a film indicated for 111 people and another pointed out for 320 individuals, which was not such a wide discrepancy range when compared to the other outputs. In addition to the discrepancy not being high, the lowest frequencies were situated in the first and in the last class, showing that there were few cases where this disparity occurs.

Nº of recommendations	Amplitude	Nº of Movies	% Frequency
111-137	27	7	3,4%
138-164	27	8	3,9%
165-191	27	16	7,7%
192-218	27	32	15,5%
219-245	27	55	26,6%
246-272	27	67	32,4%
273-299	27	21	10,1%
300-320	27	1	0,5%
Total		**207**	**207**

Table 22 - RBM: Frequency distribution table

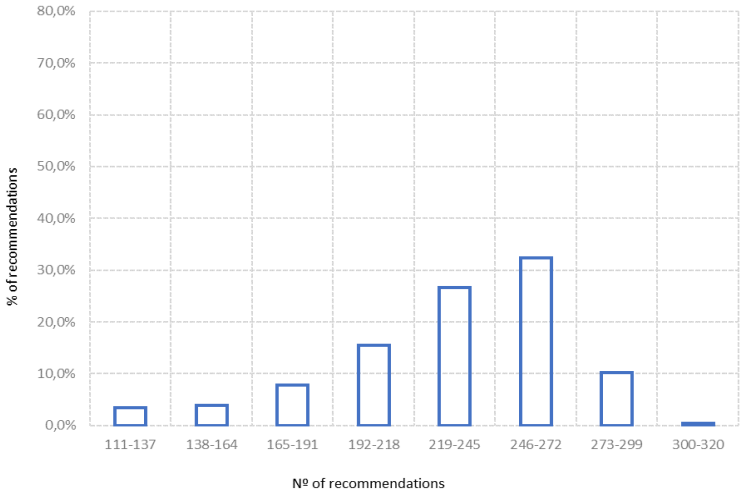


Figure 22 - RBM: Histogram (original data)

Finally, RBM was the only algorithm with a different histogram curve, as the concentration of movies tended to increase as the bands with the highest levels of number of recommendations increased (which was totally opposite to that observed in the other algorithms, since all were right-skewed, while this was left-skewed). Here, the maximum frequency was reached in the third from last class. Despite being left-skewed, it was the distribution that most closely resembled the shape of a normal distribution. The bias was toward promoting more films among the most recommended. We consider that RBM presented the best result, being contrary and much better than our initial expectation. It is interesting to have it as a counterpoint in the analysis of the outputs from the synthetic bases.

Here it is important to make an additional observation. Remember Figure 17. We can say that SVD and SVD++ intensified the pattern observed in data regarding the distribution of rating by movie, but that RBM did not mirror it. That is, the existence of long-tail seemed to be enhanced by SVD and SVD++, while it seemed not to influence RBM as this model improved the balance between popular and not-popular movies. Looking at it in another way, it cannot be stated that all algorithms behave in the same way under the presence of long-tail.

In second step of the analysis, we start to analyse the results run on the synthetic bases, beginning with Synthetic Dataset 1.

Nº of recommendations	Amplitude	Nº of Movies	% Frequency
127-159	33	9	4,3%
160-192	33	36	17,4%
193-225	33	49	23,7%
226-258	33	58	28,0%
259-291	33	28	13,5%
292-324	33	18	8,7%
325-357	33	6	2,9%
358-375	33	3	1,4%
Total		**207**	**100%**

Table 23 - Item-KNN: Frequency distribution table

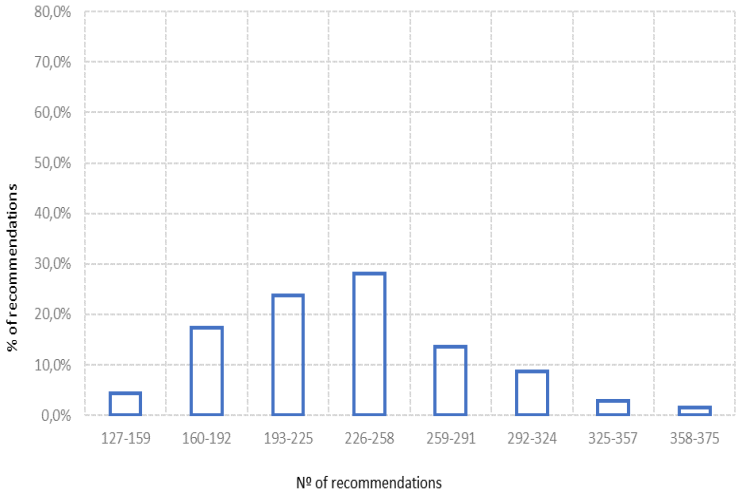


Figure 23 – Item-KNN: Histogram (Synthetic Dataset 1)

We have interesting outcomes here. A quick look at the figures and comparing them to the ones on the first step, they showed that randomization influenced all the results produced by the algorithms. Let's start with Item-KNN. First, we must note that the total amplitude (or the discrepancy between the highest and lowest number of a movie recommendation) had reduced considerably: if before we had a minimum of 23 and a maximum of 794, now we have a minimum of 127 and a maximum of 375. That is, no film was indicated for less than 127 consumers, and the extreme case was much less disadvantageous than before. Secondly, there was a significant improvement in the distribution format, which was much more symmetric, resembling the shape of a normal distribution.

Regarding SVD, it was the one with the best distribution curve, a surprisingly terrific result since in the original sample its chart resembled a negative exponential curve, having a huge elevated concentration in the first class. The curve now is very similar to a normal distribution, with a slight right skewness. The discrepancy between the least and the most recommended films also dropped a lot, showing an excellent improvement: before there was a difference from 1 to 2.115 times and now the difference is from 68 to 448 times. Also, now, together with SVD++, SVD achieved 100% coverage, being identical to the other algorithms in this score.

Nº of recommendations	Amplitude	Nº of Movies	% Frequency
68-115	48	19	9,2%
116-163	48	24	11,6%
164-211	48	39	18,8%
212-259	48	52	25,1%
260-307	48	34	16,4%
308-355	48	21	10,1%
356-403	48	12	5,8%
404-448	48	6	2,9%
Total		**202**	**100%**

Table 24 - SVD: Frequency distribution table

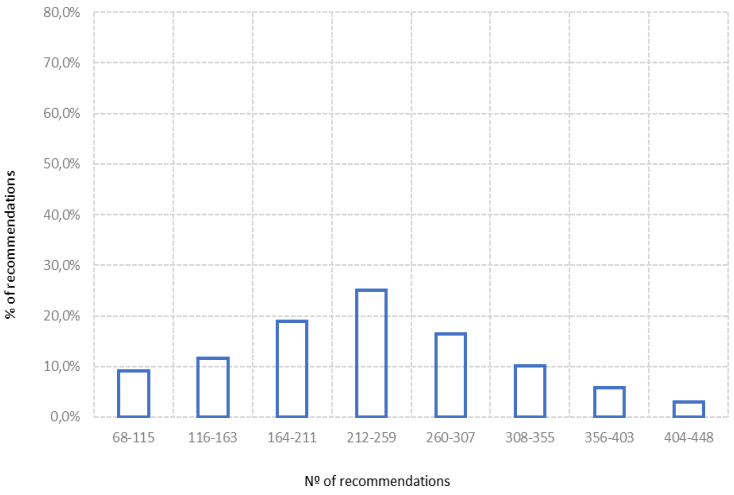


Figure 24 - SVD: Histogram (Synthetic Dataset 1)

Nº of recommendations	Amplitude	Nº of Movies	% Frequency
8-102	95	41	19,8%
103-197	95	59	28,5%
198-292	95	46	22,2%
293-387	95	32	15,5%
388-482	95	18	8,7%
483-577	95	4	1,9%
578-672	95	4	1,9%
673-761	95	3	1,4%
Total		**202**	**100%**

Table 25 - SVD++: Frequency distribution table

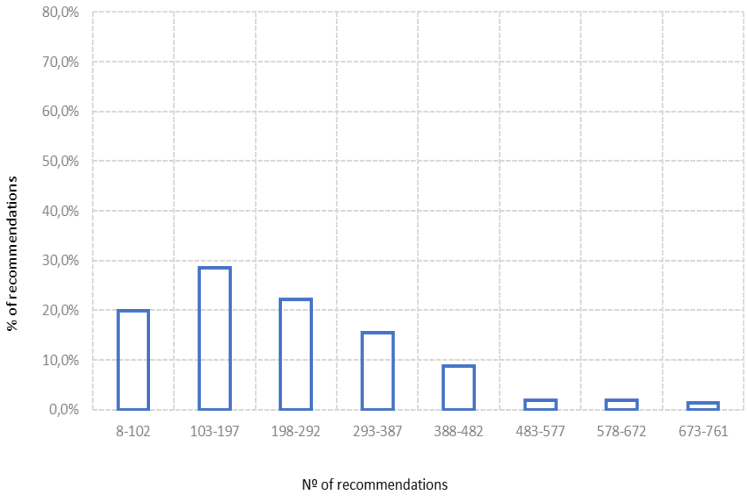


Figure 25 - SVD++: Histogram (Synthetic Dataset 1)

What draws the attention of SVD++ was that, although it had an extremely skewed outcome, similar with SVD, in the original dataset, on Synthetic Dataset 1, SVD++ was the algorithm that presented the worst distribution, in opposition to SVD, which had the best. However, if we compare all the outputs from the first and second steps, the improvement obtained by SVD++ was one of the best. The asymmetry decreased considerably and now, the right-skewed distribution had the highest

concentration in the second class, followed by the third class (ceasing to be the first class the focus of the recommendations). The divergence between the films recommended more or less times was reduced: the range of 1 to 2.318 (the worst achieved on the first step) is now from 8 to 761 time (still being the worst on the second step).

RBM			
Nº of recommendations	**Amplitude**	**Nº of Movies**	**% Frequency**
110-134	25	4	1,9%
135-159	25	12	5,8%
160-184	25	16	7,7%
185-209	25	19	9,2%
210-234	25	43	20,8%
235-259	25	53	25,6%
260-284	25	49	23,7%
285-303	25	11	5,3%
Total		**207**	**100%**

Table 26 – RBM: Frequency distribution table

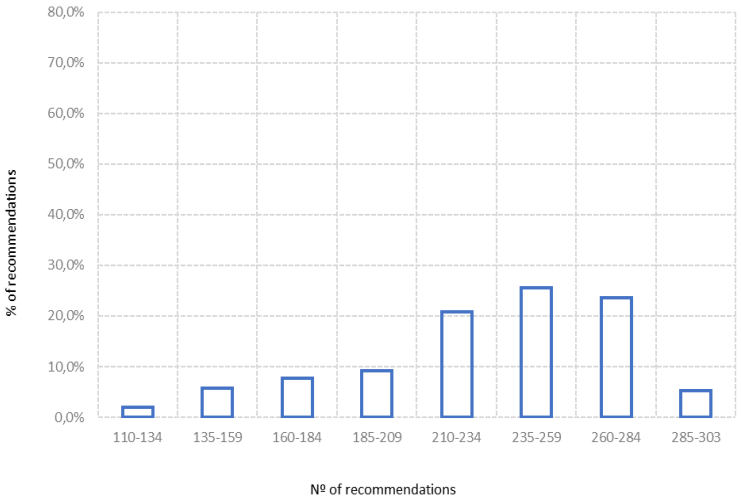


Figure 26 - RBM: Histogram (Synthetic Dataset 1)

RBM, which had the best output in the first step, suffered the slightest change in its histogram shape compared to that result. The negative asymmetry decreased and softened, but if we compare this improvement with that obtained by SVD or even by SVD++ we could expect that the result could have been much better. The drop between the minimum and maximum number of recommendations obtained by a movie was also very low if compared to the drop seen in the other algorithms: the total range that was from 111 to 320 users has been reduced from 110 to 303 times.

We can consider that the introduction of randomness in the individuals´ preference (Synthetic Dataset 1) had a greater effect in the algorithms the higher was the bias exhibited on the original base. That is, SVD and SVD++ were the algorithms whose outputs were more biased in the first step, and they were the ones which had the best improvement. Then, Item-KNN had the third best progress, and finally RBM, which was the least biased algorithm in the first step, therefore was the least impacted in the second stage.

From another perspective, we can also think that the preference of individuals had a greater influence on SVD and SVD++ algorithms. However, it may emerge a curiosity to this examination. In general, it can be inferred that the introduction of randomness in individual preference reduced the presentation bias. But we must remember that what differentiated SVD++ from SVD was the inclusion of implicit feedback. Therefore, if SVD++ added information related to the user's choice in its calculation, the introduction of randomness in those choices should had produced a greater impact on SVD++ than on SVD. However, it was on SVD that we observed a clearly better result. As mentioned earlier, SVD obtained the best and SVD++ the worst result among all the algorithms in the second step. The only possible explanation for this apparent contradiction is that on Synthetic Dataset 1 we did not change the shows watched by each user. And, implicit feedback was measured precisely by these choices. So, as SVD++ included that choices into its calculation, it means that we

did not change so much its inputs as much as we changed them to SVD. SVD ended up giving a much greater weight to the average score of the films (which was modified in Synthetic Dataset 1) than the choices made by the individuals. Therefore, the introduction of randomness ends up affecting SVD much more than SVD++ precisely because SVD did not consider these choices so much in its calculation. The opposition between the results of SVD and SVD++ in the first and second stages encourages us to affirm that much of the bias observed in the first stage was a reflection of the consumers' own choice, and that is why it was still present on SVD++ in the second step.

In any case, without doubt it can be concluded that the introduction of randomness in individual´s taste improved the result of all algorithms in terms of reducing presentation bias. That is, much of the skewness observed in the results coming from the original base was a reproduction of a pre-existing bias in the base resulting from individual's preference. More than that, we can note that the outcomes of the second step no longer suggest the strong evidence of a position bias as it existed before, specially by the strong contrast between the extreme interval´s frequencies of the histograms in the first stage.

We must now remember Figure 17 again. On Synthetic Dataset 1, we preserved the film popularity, so Figure 17 is still valid for this dataset. Hence, we are still under the same influence of the long-tail distribution. We can conclude that no algorithm was affected solely by the format of the distribution (or by the movie popularity), otherwise we should have the same result for the original and Synthetic Dataset 1. In this step, we no longer had any of the algorithms intensifying or reflecting precisely the Figure 17 shape. Thus, this was another evidence that much of the bias observed in the results of the algorithms run on the original sample was a reflection of the bias contained in the individuals' preferences.

Let's start with the analysis of the second synthetic base and third step of the analysis (Synthetic Dataset 2).

Nº of recommendations	Amplitude	Nº of Movies	% Frequency
133-164	32	7	3,4%
165-196	32	32	15,5%
197-228	32	56	27,1%
229-260	32	63	30,4%
261-292	32	35	16,9%
293-324	32	9	4,3%
325-356	32	4	1,9%
357-382	32	1	0,5%
Total		**207**	**100%**

Table 27 - Item-KNN: Frequency distribution table

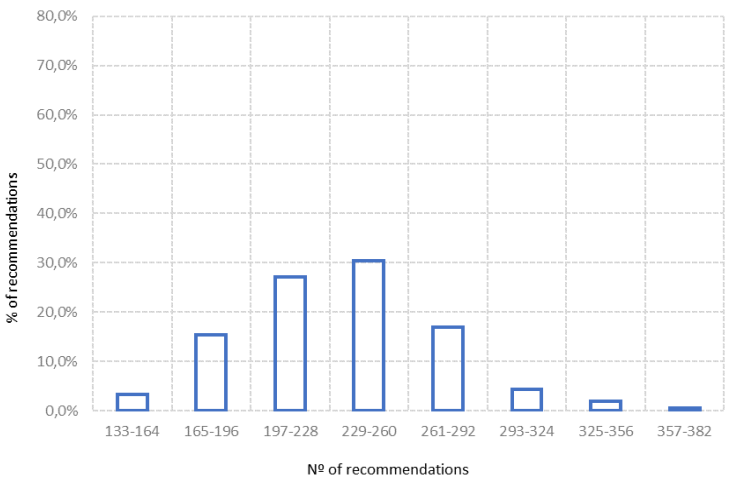


Figure 27 - Item-KNN: Histogram (Synthetic Dataset 2)

Compared to the original sample, it is clear that all distributions showed improvement in terms of reducing the asymmetry or bias. However, in relation to the first randomization, there were both improvement and worsening in the algorithms' outputs.

Nº of recommendations	Amplitude	Nº of Movies	% Frequency
60-128	69	26	12,6%
129-197	69	59	28,5%
198-266	69	57	27,5%
267-335	69	42	20,3%
336-404	69	11	5,3%
405-473	69	6	2,9%
474-542	69	3	1,4%
543-607	69	3	1,4%
Total		**207**	**100%**

Table 28 - SVD: Frequency distribution table

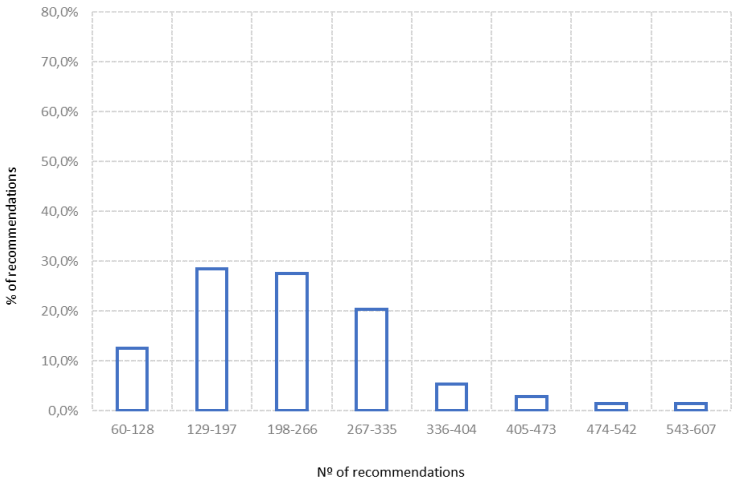


Figure 28 - SVD: Histogram (Synthetic Dataset 2)

Nº of recommendations	Amplitude	Nº of Movies	% Frequency
1-96	96	52	25,1%
97-192	96	40	19,3%
193-288	96	47	22,7%
289-384	96	35	16,9%
385-480	96	17	8,2%
481-576	96	7	3,4%
577-672	96	5	2,4%
673-761	96	4	1,9%
Total		**207**	**100%**

Table 29 - SVD++: Frequency distribution table

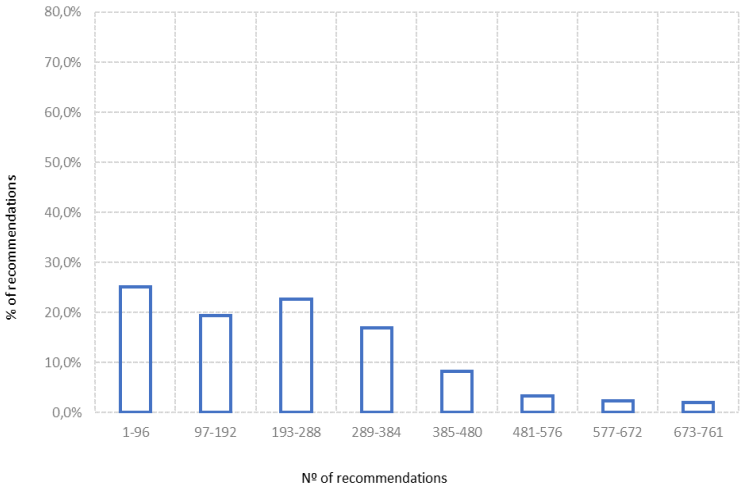


Figure 29 - SVD++: Histogram (Synthetic Dataset 2)

Item-KNN had a good improvement in relation to the original base, although in relation to the first randomization there was no substantial variation. The right skewness persisted, although much more smoothly than the one noted in the original sample.

SVD, which in the first randomization obtained the best distribution curve, now had one of the worst. However, if we compare it with the original base, we still had a great improvement and a notable positive impact (both in terms of the reduction of asymmetry and total amplitude).

SVD++ showed a right skewness that lasted the first four intervals, having a larger decay from the fifth range onwards. We can say that, in terms of number of recommendations, the films positioned

among the first four ranges did not present significant bias among themselves, and that the bias prevailed in the comparison of all those class with the successive intervals.

Finally, RBM was the only algorithm that showed a visible improvement compared to the original sample and to the first randomization. First, we must note that the total amplitude had been reduced even further: the smallest number of recommendations for a film is now 169 users and the maximum number is 296 (noticing that the ranges where these extremes occur are the least populous). In other words, there was not much difference or advantage between a film that had been suggested too much or too little. Lastly, it should be noted that the distribution is more centralized, resembling more a normal distribution, making the left skewness less pronounced and less noticeable.

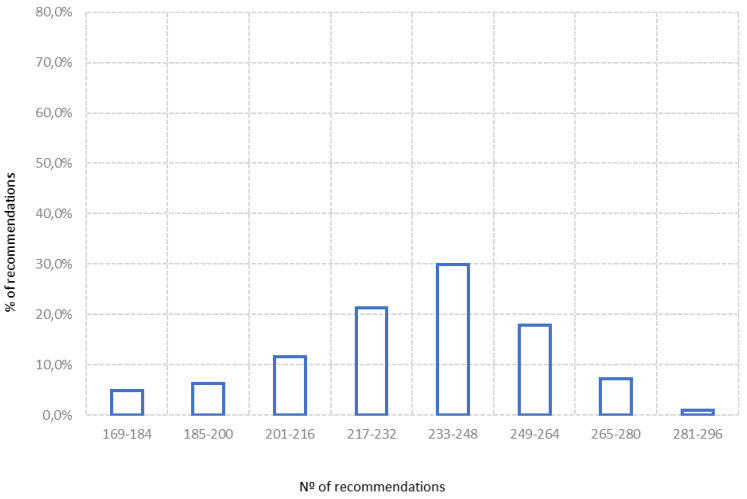


Figure 3 - RBM Histogram (Synthetic Dataset 2)

Nº of recommendations	Amplitude	Nº of Movies	% Frequency
169-184	16	10	4,8%
185-200	16	13	6,3%
201-216	16	24	11,6%
217-232	16	44	21,3%
233-248	16	62	30,0%
249-264	16	37	17,9%
265-280	16	15	7,2%
281-296	16	2	1,0%
Total		**207**	**100%**

Table 30 - RBM: Frequency distribution table

So, the greater flexibility in the randomization of individual´s preferences done in Synthetic Dataset 2 did not totally agree with what we concluded about the results of Synthetic Dataset 1. That is, it could not be stated that the more biased were the outputs on the first step, the more impacted the algorithms will be by the introduction of randomness in individual choice. If that was the case, we would expect that that SVD and SVD++ would continue to improve when compared with Synthetic Dataset 1. That is, the introduction of randomness in the original base regarding to user´s tastes contributed to make the distributions less asymmetric (just compare the first with the third stage). However, in general, the deepening of randomization in consumer´s choice did not result in a greater symmetry of the histograms (just compare the third with the second stage).

In theory, the more randomization there is in the data, the more it will be reflected in the results, which means that the histograms will tend to be more symmetric. Why then is this theory was confirmed when we compared the outcomes of the step 3 with the step 1, but not when we compared the outputs of the step 3 with the step 2? Well, if we have a less randomized base and a much more randomized base and the results don't change so much, it is a sign that probably the algorithms are responsible for that bias. So, considering the results of third stage, we can no longer confirm what was observed in the first randomization: the introduction of randomness seemed to remove the strong evidence of presentation bias. In the third step, it was evident that there were still some relationship between its distribution's shapes with those of the original sample. Those algorithms that were more biased remained the most biased (SVD and SVD++) and the least

remained the least skewed (Item-KNN and RBM). If even with randomness, the algorithms continue to reflect the pattern observed in the results of the original base, but there was not necessarily a considerable improvement with respect to Synthetic Dataset 1, we can consider this as an evidence that presentation bias may be being partly produced by the algorithm itself.

Lastly, let's analyse the results of the third randomization done and last step of the analysis (Synthetic Dataset 3). In this step we must also take in mind Figure 18.

In the third base it was where Item-KNN presented its best result in terms of a symmetric distribution. We can visualize that this was the best graph among all previous Item-KNN outcomes. It's still possible to visualize a right skewness, but much smoother. It must be highlighted that the total amplitude had been reduced a lot: the biggest difference between the number of recommendations is between 201. This difference was the second smallest obtained from all algorithms run on all bases.

Nº of recommendations	Amplitude	Nº of Movies	% Frequency
150-175	26	7	3,4%
176-201	26	37	17,9%
202-227	26	51	24,6%
228-253	26	52	25,1%
254-279	26	33	15,9%
280-305	26	15	7,2%
306-331	26	10	4,8%
332-351	26	2	1,0%
Total		**207**	**100%**

Table 31 - Item-KNN: Frequency distribution table

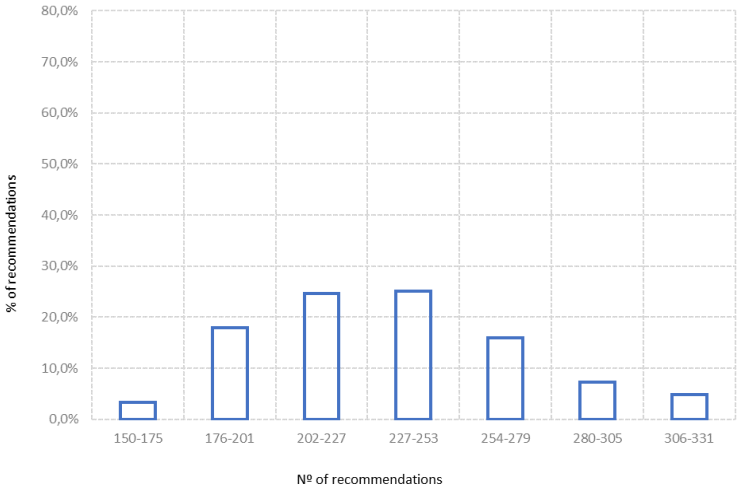


Figure 31 - Item-KNN: Histogram (Synthetic Dataset 3)

Regarding SVD, there was a slight improvement compared to Synthetic Dataset 2, most notable by the total amplitude of the recommendations. But, undoubtedly, it was on Synthetic Dataset 1 that SVD achieved its best result.

Nº of recommendations	Amplitude	Nº of Movies	% Frequency
64-124	61	18	8,7%
125-185	61	54	26,1%
186-246	61	56	27,1%
247-307	61	40	19,3%
308-368	61	24	11,6%
269-429	61	7	3,4%
430-490	61	4	1,9%
491-546	61	4	1,9%
Total		**207**	**100%**

Table 32 - SVD: Frequency distribution table

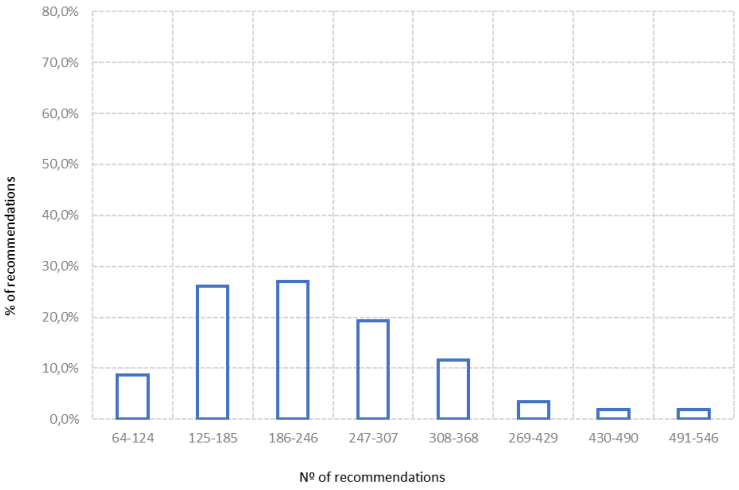


Figure 32 - SVD: Histogram (Synthetic Dataset 3)

Nº of recommendations	Amplitude	Nº of Movies	% Frequency
14-104	91	54	26,1%
105-195	91	46	22,2%
196-286	91	35	16,9%
287-377	91	33	15,9%
378-468	91	16	7,7%
469-559	91	15	7,2%
560-650	91	5	2,4%
651-735	91	3	1,4%
Total		**207**	**100%**

Table 33 - SVD++: Frequency distribution table

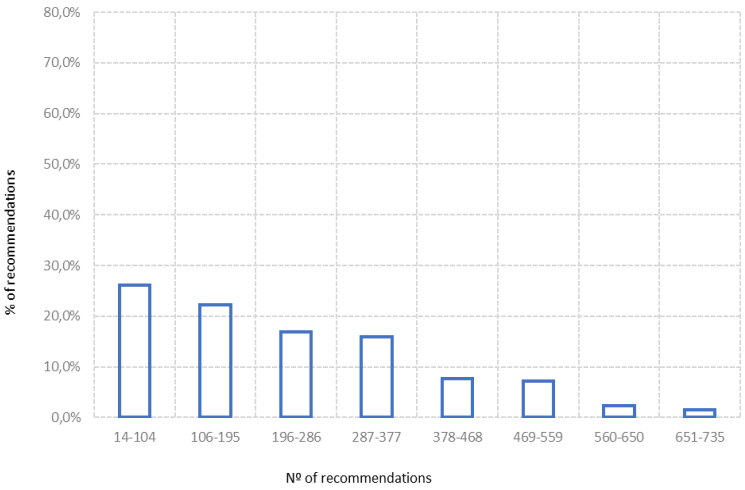


Figure 33 - SVD++: Histogram (Synthetic Dataset 3)

Nº of recommendations	Amplitude	Nº of Movies	% Frequency
195-204	10	3	1,4%
205-214	10	24	11,6%
215-224	10	37	17,9%
225-234	10	49	23,7%
235-244	10	46	22,2%
245-254	10	30	14,5%
255-264	10	14	6,8%
265-268	10	4	1,9%
Total		**207**	**100%**

Table 34 - RBM+: Frequency distribution table

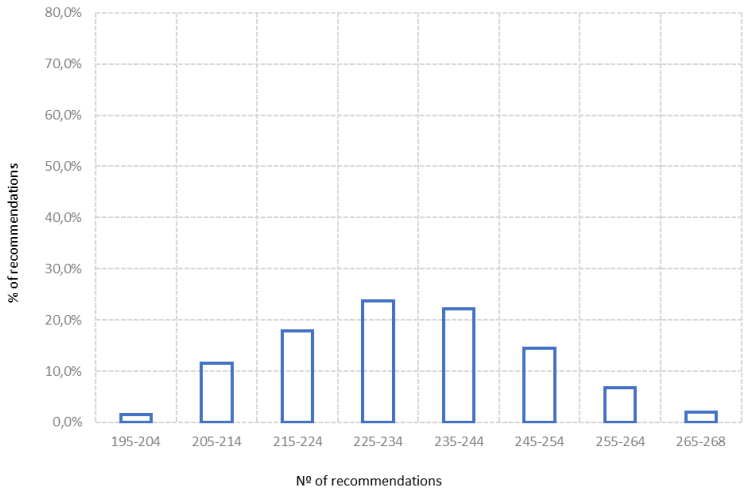


Figure 34 - RBM Histogram (Synthetic Dataset 3)

Concerning SVD++ outcomes, a common characteristic in all four tested bases was that it was always the worst when compared to the other algorithms. In Synthetic Dataset 3, the asymmetry remained visible, with a bias towards the first class.

The result of RBM in Synthetic Dataset 3 was the best result of all outcomes of this research. It should be noted that the highest difference of the number of recommendations between two films was the smallest of all: equal to 74. Thus, the work done by the algorithm was very good, since there is almost no big advantage in the number of recommendations. Plus, the higher concentration of movies was in the intermediate ranges, being minimal in the extreme bands, meaning that most of the numbers are close to the average. The standard deviation was low and equal to 15,5 times, revealing that, on average, a film was recommended for 15,5 individuals more or less than another

film (remembering that our sample had 4.800 individuals). The minimum number of times a film was suggested, equal to 195 times, was the greatest of all, which was a positive achievement too.

We must remember that the distribution of the number of evaluations per film in Synthetic Dataset 3 had a symmetrical shape (Figure 18), especially when compared to the long-tail shape of the other bases (Figure 17). With that in mind, we can infer that SVD++, followed by SVD, tended to create a non-existent bias on the Synthetic Dataset 3 by generating a disproportionate concentration of the number of recommendations on the left side of the distribution. That is, it can be observed the existence of a skewness that is incompatible with the symmetric shape of that base. Perhaps some will remember that in Synthetic Dataset 3 we standardized the distribution of the number of reviews obtained by the films, but not the distribution of rating´s value. Therefore, it could be questioned whether the outcome would be an indication that in both SVD and SVD++ models, comparing to the other models, the score value obtained by a film had a greater weight than the quantity of users evaluating it. In the first place, this could not be true due to the antagonism among their results showed on the second step of this Chapter. But besides that, in Chapter 4 we saw that the rating distribution mode was 4-stars and the mean was 3,9 stars (Figure 13), which would not justify why these algorithms tended to suggest few films at the expense of a large majority with a low number of ratings, since the distribution of the number of evaluations per film in Synthetic Dataset 3 was symmetrical (Figure 18).

Now, if both SVD and SVD++ tended to create a non-existent bias in the dataset, this would justify why they were, by far, the ones that gave the worst results when applied on the original base. In fact, with the exception of SVD in the first synthetic base, all the four results coming from SVD and SVD++ were the most skewed compared with the other two algorithms when applied on the same base. It is concluded, therefore, that SVD++ is the worst algorithm of all, followed by SVD, in terms of presentation bias.

With respect to Item-KNN, we can conclude that the algorithm tended to reproduce a pre-existing bias (or information) in the data. That is, when applied purely to the original sample, the bias occurred around the second-class interval. However, when we introduced randomness in the preferences of individuals, the distribution became more symmetrical. Finally, when we introduced randomness in the selection of films as well, that is, when Item-KNN was applied on a dataset whose distribution was much more symmetrical it is when we had the most symmetrical outcome.

RBM appeared to be the most robust model of all for the following factors. First, it was the model whose results suffered less with the modifications made on the base. Secondly, because it was the only algorithm that never created a bias around the first- or second-class interval. On the contrary, there was always a tendency to provide a greater number of recommendations for a greater number of films. Thirdly, it was the one that best reproduced the symmetry between the number of evaluations per film in Synthetic Dataset 3 (that is, it did not present a substantial evidence that presentation bias exists). Fourthly, because it was the algorithm that always produced the smallest total amplitude (smallest discrepancy between the minimum and maximum number that a film was suggested). Fifthly, because, with the exception of Item-KNN in Synthetic Dataset 1, RBM always produced the highest minimum number of recommendations that a film could have. Sixthly, because it was the only algorithm whose histogram of all cases reminded more of a normal distribution.

It is interesting to note that the introduction of randomness (here are referring to all synthetic datasets in relation to the original base) had the effect of centralizing more the distribution of the number of recommendations (or reduce its asymmetry). RBM´s histogram, which was left-skewed, always was centralized (the frequencies were moved to the left side), and Item-KNN, SVD and SVD++, which were right-skewed, always had the right skewedness reduced towards the right side of the distribution.

Finally, it is assumed that the tests performed on the of MovieLens Small dataset ended up being very influenced by the large number of elements (movies) with too little observation (users' evaluation), which ended up causing the algorithms to behave in an unexpected way. Thus, we conclude that to have a greater reliability in the recommendations generated by the algorithms, there must be a minimum number of reviewers evaluating each movie. This value cannot be very low, as it was the case of MovieLens Small dataset, where almost 50% of the movies had only one or two ratings. This point may seem obvious, but it is important because as we saw in Chapter 3, the datasets of MovieLens are constituted taking into account a minimum number of ratings by person (in general, only users that rated at least 20 films are selected), but there is no such concern regarding to the movies. Therefore, MovieLens datasets, which had been widely used in research due to their reliability, in our view has this negative feature: the existence of a significant range of films with only one, two or very few evaluations, which can cause a substantial influence on the algorithms output. Due to this conclusion, the second part of our research will take into account only the sample of MovieLens 1M.

5.2. ANALYSIS OF RMSE

Item-KNN was the only algorithm whose performance of one of the three synthetic bases differed. SVD, SVD++ and RBM RMSEs did not show a substantial difference between each synthetic base.

With Item-KNN, the first synthetic base had a lower RMSE in relation to the other two synthetic bases, for all levels of randomization. Up to 50% of randomization, the performance of Item-KNN worsened (indicated by the increase in RMSE). Between 50%-70% the RMSE remained almost constant. Between 70%-90% it seemed to have a slow increase in RMSE followed by slightly decrease until 100%.

The second and third synthetic bases exhibited close RMSE for all percentages of randomization, and their RMSE appeared to oscillate together, being the third base with a RMSE slightly lower in most cases. Up to 60% of randomization, it is easy to conclude that greater randomization led to a worsening of the performance of this algorithm in both bases. After that point, it seemed to have a slow increase in RMSE until 90% and slightly decrease until 100%.

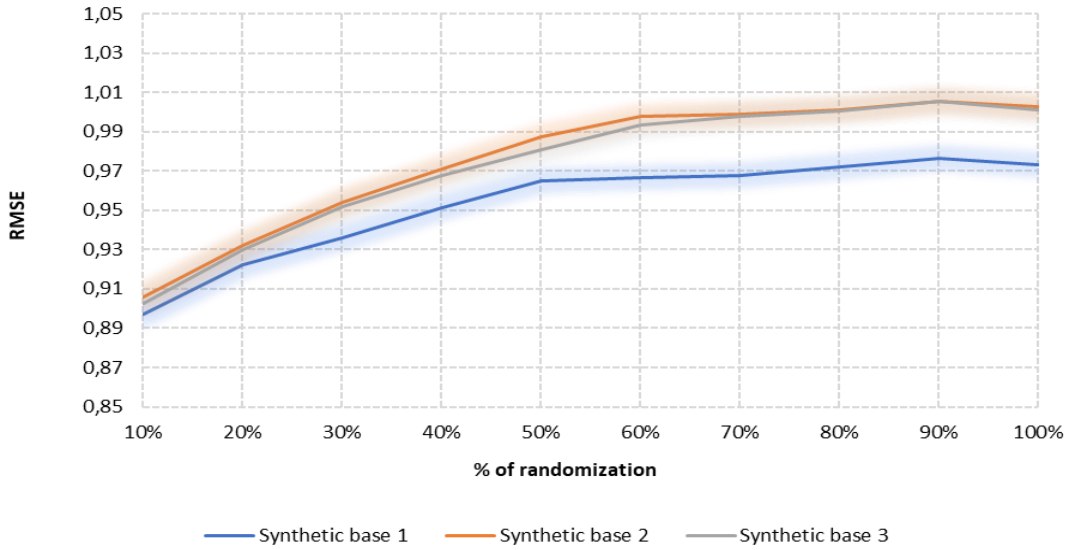


Figure 35 – Item-KNN: RMSE

Therefore, there was no significant evidence indicating that randomization could improve the performance of Item-KNN. Regarding the percentage of randomization, it is possible to infer the opposite: there was a tendency to worsen the performance as the percentage increased. Regarding the three randomization procedures conducted at the original base, on the one hand we view that smaller randomization, expressed by the lower RMSE of the first base in relation to the other synthetic bases, represented a better performance. However, on the other hand, as the second and the third bases did not show much difference, it becomes uncertain to say to what extent a more radical randomization procedure influences Item-KNN performance.

SVD and SVD++ presented a similar behaviour in their performance. The three synthetic bases exhibited similar RMSE (having the first base a slightly smaller RMSE until 40% of randomization) and their RMSE were impacted with almost the same magnitude as we changed the percentage of randomization. Therefore, it is inferred that there was no considerable difference between the RMSE of the three bases. On both models an increase on the percentage of randomization led to a continuous worsening of their efficiency. On the other side, the three different procedures of randomization done on the base did not seem to have an impact on the performance. That is, if we compare the third base – which at 100% level had a symmetrical distribution of the number of ratings per film – with the first and with the second base – that continue to be under strong presence of long-tail –, we see that the distribution of the data did not affect the RMSE (or the SVD and SVD++ performance). In this respect, it can be said that both algorithms were more robust than Item-KNN, as the value of RMSE did not depend on the randomization procedure. That is, it is as if the performance did not depend on the type of bias that we removed from the base, but it depended only on the percentage in which it occurred.

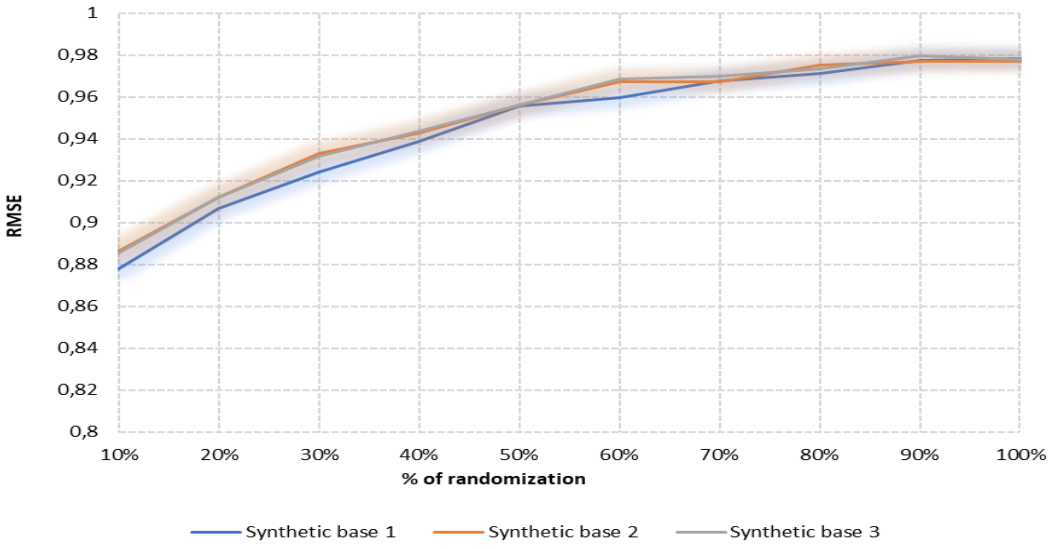


Figure 36 – SVD: RMSE

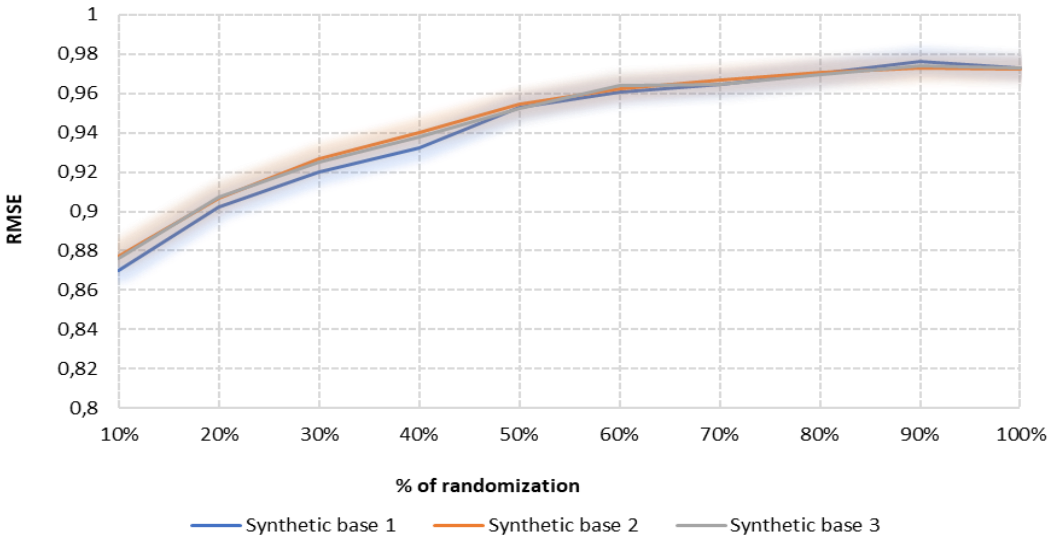


Figure 37 – SVD++: RMSE

RBM was undoubtedly the most robust model of all. Firstly, because for any percentage of randomness of all bases, the performance of the algorithm remained nearly the same, which was evidenced by the flat RMSE curves. That was an exclusive result achieved by RBM, which characterizes an extreme scenario in which its performance did not depend on the level of randomization. Secondly, because we see the same phenomenon observed in SVD and SVD++: the ineffectiveness of the different procedures of randomness introduced in each synthetic base over the performance of the algorithm.

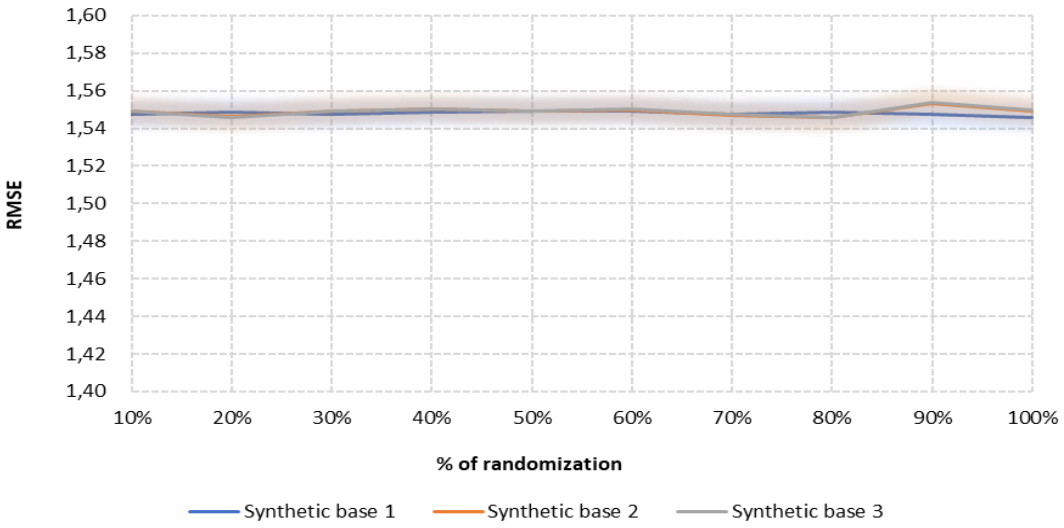


Figure 38 – RBM: RMSE

That is, the performance of RBM was always the same, regardless the randomization procedure done at the base, regardless the level of randomization applied over the base, regardless the kind of bias that are removed from the base or regardless the distribution curve of the number rating by movie. However, it was the algorithm with the highest RMSE for all percentages of randomization.

The results of the second part of our research are not total in line with those of the first part. In the first part, for example, we saw that RBM presented a continuous improvement as we increased the randomization procedure done on the original sample. Thus, we could expect its RMSE to be different for the three synthetic bases, being higher for the first and lower for the last. Also, in the first part, we noticed that SVD had the best result in the first synthetic base. Therefore, we could expect it RMSE curve to be below the curves of the other bases. We don't see any of that. Taking into consideration all the results of both parts of our research, we are unable to explain why Item-KNN is the only algorithm whose RMSE curve of a synthetic dataset (synthetic 1) differed by being below the other curves.

However, what is common in both parts of our study is that RBM resulted in being the most robust algorithm, since it was the one that suffered the least shock with the introduction of randomness.

Another interesting fact is to note that the increase in the level of randomness, in general, impacted the RMSE of the three bases in the same magnitude. For all algorithms, the curves practically move together. What draws attention is that it would be expected that the curve of the synthetic base 3 should detach as we approach 100%, since as we increase the percentage, we change the shape of its long-tail distribution to a normal distribution. Thus, it would be expected that this fact should have a visible impact on the performance of the algorithms. However, the performance of Item-KNN, SVD and SVD++ on the third dataset worsens in the same proportion as in the first and in the second datasets. And the performance of RBM is indifferent of the database used. On the other hand, in the first part of our research it was also expected that the recommendations from the third base would have a symmetrical distribution, or, mainly, that the distribution would be much more symmetrical

than the other bases. However, this is not what we saw. This only occurred for RBM and Item-KNN and not with as much difference as expected. The distributions of the third synthetic dataset was much closer than distant to the others synthetic bases. In this sense, this is more in line with the correspondence of the RMSE curves (the RMSE of the third base with the RMSE of the others dataset) seen in the second part of the research.

5.3. ANALYSIS OF DIVERSITY

The research was impaired in terms of diversity analysis. We saw that selection of a full MovieLens dataset (MovieLens Small) produced outputs that lacked reliability, in our judgment. Thus, we could not measure the diversity of the recommendations resulted from MovieLens Small if we believe that the algorithms have been strongly influenced by an imbalance in the database, being unable to produce a trustworthiness output. Even if the measured Coverage reflects something, we believe that this measure is insufficient to conclude about the behaviour of the algorithms in relation to the diversity of its suggestions.

As we saw in Chapter 3, our objective in selecting a sample from MovieLens 1M base was to establish a reasonable minimum number of evaluations per film in order to have a reliable result for the first and, consequently, second part of the research. However, by doing so, the diversity analysis was sacrificed. By forming the sample the way we did, we simply eliminated the less popular movies, not ensuring that it was a representative sample of MovieLens 1M dataset. In fact, we can see that in almost all cases, all the algorithms reached 100% coverage (the 207 films in the sample were indicated to someone). But we suspect that these scores were much more a reflection of the sample than a reflection of the algorithm itself. That is, we presume that there is a difference in each algorithm behaviour in relation to Coverage, as it happened in MovieLens Small dataset. Thus, everything indicates that not even looking at the measured Coverage we can conclude something about Diversity.

6. CONCLUSIONS

The four main movie recommendation algorithms are: Item based neighborhoods (Item-KNN), Singular Value Decomposition (SVD), SVD++ and Restricted Boltzmann Machines (RBM). Besides they were quoted as the most popular or as with the best performance or as with the best fit for the Netflix company usage, we ensured that they were representative of distinct methods of machine learning for recommender system. The first three of them were from collaborative filtering method: Item-KNN from memory-based model and the SVD and SVD++ from model-based technique, specifically, from matrix factorization. The difference between those last two had the purpose to include implicit feedback in this study. Lastly, from deep learning method, RBM was selected.

To conduct the research, it was necessary to subsample MovieLens 1M dataset. Each algorithm was applied separately in data in order to deliver the top 10 highest predictions for each user as a recommendation. It was examined the distribution of the number of recommendations obtained per movie. The outputs showed a strong evidence of presentation bias in at least two algorithms: SVD and SVD++, which intensified a pre-existing (inputs) asymmetry in the number of evaluations per film (or movie popularity).

Then, it was started the process of data randomization. First, randomness was introduced in the individuals‘ preferences, which were shuffled between their own choices. The movie popularity was preserved, keeping intact the inputs asymmetry in relation to the number of evaluations received per movie. All the algorithms had a visible improvement in their histograms, suggesting that part of the previous observed bias was a reflection of individual´s taste, and not exactly a reflection of the movie popularity. In other words, was found indication that part of the bias derived from the human being himself, however, it was not all exclusive to human´s behaviour, since there was still a bias remaining in some algorithms. SVD was the one which experienced the most drastic advance, practically ceasing to exhibit evidence of presentation bias.

Next, the randomization of consumer´s preference was intensified, which were mixed among each other. We maintained untouched the movie popularity. Although there was a reduction in asymmetry in comparison with the results from the original database, when compared with the results from the first randomization procedure there was a considerable worsening for SVD and no significant improvement for Item-KNN and SVD++. The greater randomization did not promote a proportional improvement in the symmetry of the distributions, and the outcomes signalized that probably part of the presentation bias could be being generated by the algorithms themselves.

Finally, it was made a random draw of the films from the catalogue among the individuals, being therefore randomized the movie popularity and the individuals´ choices. In this last procedure, we changed the input distribution, which went from having a long-tail format to having a shape similar to that of a normal distribution. Item-KNN and RBM presented a symmetrical distribution, while SVD and SVD++ continued to be right-skewed. This showed evidence that some algorithms were not intensifying or reproducing the input patterns and continued to exhibit a tendency towards a few movies in the number of recommendations, inconsistent with the popularity of films or the humans' choice. With that, it was concluded that part of presentation bias could be generated by some algorithms.

In terms of presentation bias, SVD++ was always the one with the worst results, being considered the worst of all algorithms because it expressed a tendency to generate this bias in all steps. That is, in general, few films had a high number of recommendations, while a large majority tended to be at disadvantage by being suggested to few users. In sequence, SVD was considered the second worst, as it also exhibited a tendency to generate presentation bias.

RBM was the most robust model which delivered substantially better results. In all procedures RBM tended to favour a greater number of films to have a larger public when compared to the other algorithms. In all stages, RBM tended to increase the minimum number of recommendations obtained by movie, to reduce the discrepancy between the maximum and minimum number of suggestions and tended to be the algorithm with the better achievements.

Based on all results, it had been found evidence of presentation bias. In part this bias was generated or intensified by some algorithms, but also in part the bias was mere reflection of a pre-existing bias in the human's preferences. In any case, it was concluded that the algorithms did not behave in a single way, such as RBM, which never exhibited presentation bias in its recommendations, or Item-KNN, which tended to reproduce a pre-existing bias (or information) in the data.

In general, the randomization procedures done on the database contributed to reduce the asymmetry and to centralize the frequency distributions. However, when compared to each other, the intensification of randomization performed on the database did not have a proportional effect on the results, by not conducting a higher centralization of the frequencies distributions.

By an exclusive analysis of the algorithm's performance, it was concluded that usually an increase in the percentage of data randomized led to a worsening of RMSE. However, there was no evidence that the intensification of the different randomization procedures applied at the original base had any effect on RMSE. Typically, the algorithm's performance depended more on the percentage at which the randomization was done than on the type of randomization done on the dataset. Comparing the algorithm's performance between themselves, RBM once again proved to be the most robust model because it presented a high stability on its RMSE curves, although it was the one that presented the worst RMSE value.

All that said, it was concluded that a higher accuracy (lower RMSE) was not correlated with a better performance in terms of the recommendation's distributions. More specifically, with regard to presentation bias. Indeed, this study brought more evidence to the line of thought that believe that rating predictive accuracy metrics may not be the most suitable for a ranking purpose. Our analysis leads us to believe that both algorithm's performance and machine learning process should consider other metrics rather than RMSE. We even remain open to the possibility that part of the observed presentation bias may be a reflection of the usage of improper metrics.

7. LIMITATIONS AND RECOMMENDATIONS FOR FUTURE WORKS

The biggest limitation was the computational capacity available to do the research. The ideal scenario would be to run the selected algorithms on the second largest dataset available on MovieLens website (MovieLens 25M), which is the most recent. However, it was only on the second smallest database (MovieLens Small) that we managed to apply the models. Yet, as it was revealed, the minimum number of reviews by movie on those datasets was extremely low and, in general, the relative number of films with few evaluations was significant. It was demonstrated that this ended up having a great impact on the functioning of the algorithms, losing the reliability of the results, in our view. For these reasons, it is suggested to reapply this study not on one of those largest bases, but on a sample of them with the intention to correct this problem, seeking to minimize the loss of information, especially in relation to diversity.

Our research was mainly done with Surprise library. That is, we followed some pre-established definitions in this library regarding the selected algorithms, as well as the similarity metrics. Therefore, it would be interesting to apply the research in other validated models for Item-KNN, SVD, SVD++ and RBM and check if there are relevant changes in the results found. Moreover, even for each of these algorithms there was a huge number of configurations that could be tested against each other to verify if there would be any considerably better.

This study was impaired in diversity analysis, which could not be done for the reasons explained. We believe that this is an important, complementary and inherent aspect of our research theme. So, it is essential to proceed with further investigations of which would be the best method to accomplish this analysis.

A critical limitation of our work, somewhat complex to be faced, was presented in Chapter 2. The algorithms used by the recommendation systems works predominantly based on the rating prediction accuracy. But this not necessarily perform in relation to the main task: the production of a top-N recommendation list (McNee, Riedl, & Konstan, 2006). For example, when we fixed that our analysis would be based on the top 10 recommendations by user, what we have was the 10 films with the highest rating predicted for each person by the algorithms. But the ratings were predicted taking into account the accuracy metric, and not a ranking criterion. So, we had a result concerning about RMSE and not with the quality of top-N suggestions. Therefore, in the interest of deepen the understanding of presentation bias, one should not seek to measure it in the most used algorithms, but rather to create new models, or adapt the existing algorithms, that were relied on ranking criteria (naturally, along with that, it should be investigated what kind of database is ideal for such purpose).

Moreover, we saw that sparsity (which was present in our datasets and very common in many databases) impacts negatively accuracy measures. However, we do not know how much RMSE is affected by it and how much it may have influenced the algorithm´s outcome.

Another limitation, already presented in Chapter 2, concerns the infeasibility of conducting online tests. However, in relation to this issue, it would only be possible to circumvent it by having access to a real and dynamic database, which, even in an affirmative case, it would be hardly to have a wide academic dissemination.

A constraint, previously mentioned in Chapter 3, was related to the high possibility that the databases used in this research were already influenced by presentation bias. Since the datasets were affected by collaborative filtering applied on MovieLens website to order the movies in front of users, the origin per se of this problem could not be independently observed. That is, the inputs were already a product of filtering mechanisms and therefore contain some information related to the presentation bias[29]. However, this is a difficult problem to be overcome. Even if we put the films in alphabetical order, if we take a relatively small base like MovieLens Small, we had 9.724 movies. It is to be questioned whether the initial films would tend to be much more rated than the ones in other positions. That is, in a catalogue with a reasonable number of movies, it is impossible to eliminate the ordering issue as there must be always some movies shown before others. Our suggestion to work around this problem is to get a sample of users large enough to ensure that all films from the catalogue could has the same probability to be arranged firstly to each reviewer, and then, for each person the films would be ordered randomly. Thus, although in the individual sphere the ordering problem could not be eliminated, having a large sample of individuals for which this ordering was random it would be a way to get rid of presentation bias, as we would ending up having a sample where the order was random.

[29] Even though all users selected to compose each dataset had been chosen randomly by MovieLens, the movies evaluated by them reflects this filtering mechanism.

8. BIBLIOGRAPHY

Abdollahi, B., & Nasraoui, O. (2016). *Explainable Restricted Boltzmann Machines for Collaborative Filtering*. (Whi). Retrieved from http://arxiv.org/abs/1606.07129

Abdollahpouri, H., Burke, R., Mansoury, M., & Mobasher, B. (2019). *The Unfairness of Popularity Bias in Recommendation*.

Abdollahpouri, H., Burke, R., & Mobasher, B. (2019). *Managing Popularity Bias in Recommender Systems with Personalized Re-ranking*.

Adomavicius, G., & Tuzhilin, A. (2005). Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. *The Philosophical Review*, *12*(5), 548. https://doi.org/10.2307/2176678

Aggarwal, C. C. (2016). *Recommender Systems* (Vol. 40). https://doi.org/10.1145/245108.245121

Ahmed, M., Imtiaz, M. T., & Khan, R. (2018). Movie recommendation system using clustering and pattern recognition network. *2018 IEEE 8th Annual Computing and Communication Workshop and Conference, CCWC 2018*, *2018-Janua*, 143–147. https://doi.org/10.1109/CCWC.2018.8301695

Amatriain, X., & Basilico, J. (2012a). Netflix Recommendations: Beyond the 5 stars (Part 1). *Netflix Technology Blog Apr*, (Part 1).

Amatriain, X., & Basilico, J. (2012b). Netflix Recommendations: Beyond the 5 stars (Part 2). *Netflix Technology Blog*, (Part 2), 1–11.

Anderson, C. (2006). *The long tail - Why the future of business is selling less of more*. 238. Retrieved from http://dl.motamem.org/long_tail_chris_anderson_motamem_org.pdf

Ayyaz, S., Qamar, U., & Nawaz, R. (2018). HCF-CRS: A Hybrid content based fuzzy conformal recommender system for providing recommendations with confidence. In *PLoS ONE* (Vol. 13). https://doi.org/10.1371/journal.pone.0204849

Bar-Ilan, J., Keenoy, K., Levene, M., & Yaari, E. (2008). Presentation Bias is significant in determining user preference for search result - A User Study. *Journal of the American Society for Information Science*, *59*(1), 126–135. https://doi.org/10.1002/asi

Bell, R. M., & Koren, Y. (2007a). *Improved Neighborhood-based Collaborative Filtering*. 7–14.

Bell, R. M., & Koren, Y. (2007b). Lessons from the Netflix prize challenge. *ACM SIGKDD Explorations Newsletter*, *9*(2), 75. https://doi.org/10.1145/1345448.1345465

Bell, R. M., Koren, Y., & Volinsky, C. (2007a). *Modeling relationships at Multiple Scales to improve cccuracy of large Recommender Systems*.

Bell, R. M., Koren, Y., & Volinsky, C. (2007b). *The BellKor solution to the Netflix Prize*. 15. Retrieved from https://www.netflixprize.com/assets/ProgressPrize2007_KorBell.pdf

Bell, R. M., Koren, Y., & Volinsky, C. (2009). The BellKor 2008 Solution to the Netflix Prize. *Pathologie-Biologie*, (12), 1–21. https://doi.org/10.1016/j.patbio.2009.09.005

Bellogı, A., Castells, P., & Cantador, I. (2017). *Statistical biases in Information Retrieval metrics for recommender systems*. 606–634. https://doi.org/10.1007/s10791-017-9312-z

Bennett, J., & Lanning, S. (2007). The Netflix Prize. *Proceedings of KDD Cup and Workshop*, 3–6. https://doi.org/10.1145/1562764.1562769

Bougiatiotis, K., & Giannakopoulos, T. (2018). Enhanced movie content similarity based on textual, auditory and visual information. *Expert Systems with Applications*, *96*, 86–102. https://doi.org/10.1016/j.eswa.2017.11.050

Celma, Ò., & Cano, P. (2008). *From hits to niches? or how popular artists can bias music recommendation and discovery*.

Chandrashekar, A., Amat, F., Basilico, J., & Jebara, T. (2017). Artwork Personalization at Netflix. *Netflix Technology Blog*, (6), 67–72.

Chen, H. (2017). *Weighted-SVD: Matrix Factorization with Weights on the Latent Factors*. (2010), 1–19.

Chen, L., Zhang, G., & Zhou, H. (2017). *Improving the Diversity of Top-N Recommendation via Determinantal Point Process*.

Choi, S. M., Ko, S. K., & Han, Y. S. (2012). A movie recommendation algorithm based on genre correlations. *Expert Systems with Applications*, *39*(9), 8079–8085. https://doi.org/10.1016/j.eswa.2012.01.132

Corduneanu, A. D., & Kim, H. (2015). *Modifying search result ranking based on implicit user feedback and a model of presentation bias*. *1*(12).

Cremonesi, P., Koren, Y., & Turrin, R. (2010). Performance of recommender algorithms on top-N recommendation tasks. *RecSys'10 - Proceedings of the 4th ACM Conference on Recommender Systems*, 39–46. https://doi.org/10.1145/1864708.1864721

Deldjoo, Y., Elahi, M., & Cremonesi, P. (2016). *Using visual features and latent factors for movie recommendation*.

Deshpande, M., & Karypis, G. (2004). *Item-Based Top-N Recommendation Algorithms*. *22*(1), 143–177.

Economist, T. (2019). Disney, AT&T and Comcast v Netflix, Amazon and Apple. *The Economist*, 1–7. Retrieved from https://www.economist.com/business/2019/03/30/disney-at-and-t-and-comcast-v-netflix-amazon-and-apple

Farine, D. R. (2017). *A guide to null models for animal social network analysis*. 1309–1320. https://doi.org/10.1111/2041-210X.12772

Fleder, D., & Hosanagar, K. (2009). *Blockbuster culture's next rise or fall: The effect of recommender systems on sales diversity*. (2007).

Fontaine, G., & Simone, P. (2017). VOD distribution and the role of aggregators. *European Audiovisual Observatory*, (May).

Funk, S. (2006). Netflix Update: Try This at Home. Retrieved from https://sifter.org/~simon/journal/20061211.html

Ghazanfar, M. A., Iqbal, H., Azam, M. A., Aljohani, N. R., & Alowibdi, J. S. (2018). Building scalable and accurate hybrid kernel mapping recommender systems. *2017 Intelligent Systems Conference, IntelliSys 2017*, *2018-Janua*(September), 488–493. https://doi.org/10.1109/IntelliSys.2017.8324338

Ghazanfar, M. A., & Prugel-bennett, A. (2010). A scalable, accurate hybrid recommender system. *2010 Third International Conference on Knowledge Discovery and Data Mining*, (2), 94–98. https://doi.org/10.1109/WKDD.2010.117

Gomez-Uribe, C. A., & Hunt, N. (2015). The Netflix Recommender System. *ACM Transactions on Management Information Systems*, *6*(4), 1–19. https://doi.org/10.1145/2843948

Goodfellow, I., Bengio, Y., & Courvill, A. (2016). Deep learning. In *Cambridge, Massachusetts, London, England: The MIT Press*. https://doi.org/10.1007/s10710-017-9314-z

Hallinan, B., & Striphas, T. (2016). The Netflix Prize and the production of algorithmic culture. *New Media and Society*, *18*(1), 117–137. https://doi.org/10.1177/1461444814538646

Han, Z., Wang, D., & Hong, M. (2017). *Signal Processing and Networking for Big Data Applications*.

Harper, F. M., & Konstan, J. A. (2015). The MovieLens Datasets: History and Context. *ACM Transactions on Interactive Intelligent Systems (TiiS)*, *5*(4), 1–19. https://doi.org/https://doi.org/10.1145/2827872

He, J., & Chu, W. W. (2010). *A Social Network-Based Recommender System (SNRS)*. 47–74. https://doi.org/10.1007/978-1-4419-6287-4_4

Himabindu, T. V. R., Padmanabhan, V., & Pujari, A. K. (2018). Conformal matrix factorization based recommender system. *Information Sciences*, *467*, 685–707. https://doi.org/10.1016/j.ins.2018.04.004

Hu, Y., Koren, Y., & Volinsky, C. (2008). *Collaborative Filtering for Implicit Feedback Datasets*. https://doi.org/10.1109/ICDM.2008.22

Huang, H., Zhao, Y., Yan, J., & Yang, L. (2015). *Improve the "Long Tail " Recommendation through Popularity-Sensitive Clustering*.

Hurley, N., & Zhang, M. I. (2011). *Novelty and diversity in Top- N recommendation – Analysis and evolution*. https://doi.org/10.1145/1944339.1944341

Hwang, K., & Chen, M. (2017). *Big-Data Analytics for Cloud, IoT and Cognitive Computing*.

Jannach, D., & Jugovac, M. (2019). *Measuring the Business Value of Recommender Systems*. Retrieved from http://arxiv.org/abs/1908.08328

Jung, J. J. (2012). Attribute selection-based recommendation framework for short-head user group: An empirical study by MovieLens and IMDB. *Expert Systems with Applications*, *39*(4), 4049–4054. https://doi.org/10.1016/j.eswa.2011.09.096

Kane, F. (2019). *Building Recommender Systems with Machine Learning and AI*.

Karypis, G. (2001). *Evaluation of Item-Based Top-N Recommendation Algorithms*. 1–13.

Kesharwani, M. A., Rakesh, M., Tech, B. M., Abdul, A. P. J., & Prof, A. (2017). Movie Rating Prediction Based on Twitter Sentiment Analysis. *Journal of Advanced Computing and Communication Technologies*, *5*(1), 2347–2804. Retrieved from https://apps.twitter.com/app/new.

Khan, M. W., Chan, G., Chua, F., & Haw, S. (2017). *Ontology-Based Hybrid Recommender System for Internet Protocol Television*. https://doi.org/10.1007/978-3-319-70010-6

Konstan, J. A., & Riedl, J. (2012). *Recommender systems: from algorithms to user experience*. 101–

123. https://doi.org/10.1007/s11257-011-9112-x

Koren, Y. (2008). Factorization Meets the Neighborhood: a Multifaceted Collaborative Filtering Model. *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 426–434.

Koren, Y., Bell, R., & Volinsky, C. (2015). *Matrix Factorization Techniques for Recommender Systems*. 199–207. https://doi.org/10.1002/9781118770368.ch6

Lee, D., & Hosanagar, K. (2014). Impact of recommender systems on sales volume and diversity. *35th International Conference on Information Systems "Building a Better World Through Information Systems", ICIS 2014*, 1–15.

Levene, M. (2010). *An Introduction to Search Engines and Web Navigation*.

Li, G., & Chen, Q. (2016). Exploiting explicit and implicit feedback for personalized ranking. *Mathematical Problems in Engineering*, *2016*. https://doi.org/10.1155/2016/2535329

Liu, C., Last, M., & Shmilovici, A. (2019). Identifying turning points in animated cartoons. *Expert Systems with Applications*, *123*, 246–255. https://doi.org/10.1016/j.eswa.2019.01.003

Lops, P., Musto, C., Narducci, F., & Semeraro, G. (2019). *Semantics in Adaptive and Personalised Systems - Methods, Tools and Applications*.

Louppe, G. (2010). *Collaborative filtering*.

Luhaniwal, V. (2019). Why Gradient descent isn ' t enough: A comprehensive introduction to optimization algorithms in neural networks. *Towards Data Science*, 1–18.

McNee, S. M., Riedl, J., & Konstan, J. A. (2006). Accurate is not always good: How Accuracy Metrics have hurt Recommender Systems. *Search*, 1097–1101. Retrieved from http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Accurate+is+not+always+good:+How+Accuracy+Metrics+have+hurt+Recommender+Systems#1

Moreno, M. N., Segrera, S., López, V. F., Muñoz, M. D., & Sánchez, Á. L. (2016). Web mining based framework for solving usual problems in recommender systems. A case study for movies' recommendation. *Neurocomputing*, *176*, 72–80. https://doi.org/10.1016/j.neucom.2014.10.097

Musto, C., Narducci, F., Lops, P., de Gemmis, M., & Semeraro, G. (2019). Linked open data-based explanations for transparent recommender systems. *International Journal of Human Computer Studies*, *121*(May 2017), 93–107. https://doi.org/10.1016/j.ijhcs.2018.03.003

Nikolakopoulos, A. N., Berberidis, D., Karypis, G., & Giannakis, G. B. (2019). Personalized difusions for Top-N recommendation. *RecSys 2019 - 13th ACM Conference on Recommender Systems*, 260–268. https://doi.org/10.1145/3298689.3346985

Park, Y., & Tuzhilin, A. (2008). *The Long Tail of Recommender Systems and How to Leverage It*. 11–18.

Patel, A. A. (2019). Hands-On Unsupervised Learning Using Python. In *O'Reilly*. O'Reilly Media.

Piotte, M., & Chabbert, M. (2009). The Pragmatic Theory solution to the Netflix Grand Prize. *Working Paper*, (August), 1–92. https://doi.org/10.1.1.162.2118

Quadrana, M., Piazzolla, P., Elahi, M., Cremonesi, P., Deldjoo, Y., & Garzotto, F. (2016). Content-Based Video Recommendation System Based on Stylistic Visual Features. *Journal on Data Semantics*, *5*(2), 99–113. https://doi.org/10.1007/s13740-016-0060-9

Resnick, P., Bergstrom, P., Riedl, J., & Iacovou, N. (1994). *GroupLens: An Open Architecture for Collaborative Filtering of Netnews*. 175–186.

Resnick, P., Varian, H. R., & Editors, G. (1997). Recommender Systems. *Communications of the ACM*, *40*(3), 56–58. Retrieved from http://portal.acm.org/citation.cfm?id=245121

Ricci, F., Rokach, L., Shapira, B., Kantor, P. B., & Ricci, F. (2011). *Recommender Systems Handbook*. https://doi.org/10.1007/978-0-387-85820-3

Ruder, S. (2016). *An overview of gradient descent optimization algorithms*. 1–14. Retrieved from http://arxiv.org/abs/1609.04747

Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2001). *Item-based Collaborative Filtering Recommendation Algorithms*.

Steck, H. (2018). *Calibrated recommendations*. 154–162. https://doi.org/10.1145/3240323.3240372

Su, X., & Khoshgoftaar, T. M. (2009). *A Survey of Collaborative Filtering Techniques*. *2009*(Section 3). https://doi.org/10.1155/2009/421425

Tewari, A. S., Singh, J. P., & Barman, A. G. (2018). Generating Top-N items recommendation set using Collaborative, Content Based Filtering and rating variance. *Procedia Computer Science*, *132*(Iccids), 1678–1684. https://doi.org/10.1016/j.procs.2018.05.139

Toscher, A., & Jahrer, M. (2009a). *The BigChaos Solution to the Netflix Grand Prize*. 1–52.

Toscher, A., & Jahrer, M. (2009b). The BigChaos Solution to the Netflix Prize 2008. *Netflix Prize Documentation*, (August), 1–17. https://doi.org/10.1.1.162.2118

Truyen, T. T., Phung, D. Q., & Venkatesh, S. (2009). Ordinal Boltzmann machines for collaborative filtering. *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence, UAI 2009*, 548–556.

Wang, H., & Zhang, H. (2018). Movie genre preference prediction using machine learning for customer-based information. *2018 IEEE 8th Annual Computing and Communication Workshop and Conference, CCWC 2018*, *2018-Janua*, 110–116. https://doi.org/10.1109/CCWC.2018.8301647

Wang, S., Tang, J., Wang, Y., & Liu, H. (2018). Exploring hierarchical structures for recommender systems. *IEEE Transactions on Knowledge and Data Engineering*, *30*(6), 1022–1035. https://doi.org/10.1109/TKDE.2018.2789443

Wang, Z., Guo, Y., & Du, B. (2018). Matrix completion with preference ranking for top-n recommendation. *IJCAI International Joint Conference on Artificial Intelligence*, *2018-July*, 3585–3591. https://doi.org/10.24963/ijcai.2018/498

Wu, J., & Li, T. (2008). *A Modified Fuzzy C-Means Algorithm For Collaborative Filtering*. 2–5.

Xian, Z., Li, Q., Li, G., & Li, L. (2017). *New Collaborative Filtering Algorithms Based on SVD ++ and Differential Privacy*. *2017*.

Yang, H., & Qiu, R. (2018). *Advances in Service Science - Proceedings of the 2018 INFORMS International Conference on Service Science*.

Yin, H., Cui, B., Li, J., Yao, J., & Chen, C. (2012). *Challenging the Long Tail Recommendation*. *5*(9), 896–907.

Yue, Y., Patel, R., & Roehrig, H. (2010). Beyond position bias: Examining result attractiveness as a source of presentation bias in clickthrough data. *Proceedings of the 19th International Conference on World Wide Web, WWW '10*, 1011–1015. https://doi.org/10.1145/1772690.1772793

Zolaktaf, Z., Babanezhad, R., & Pottinger, R. (2018). A generic Top-N recommendation framework for trading-off accuracy, novelty, and coverage. *Proceedings - IEEE 34th International Conference on Data Engineering, ICDE 2018*, 149–160. https://doi.org/10.1109/ICDE.2018.00023