Item recommendation based on ratings that are not missing at random

Jacques Hasard Romano Fenzo

December 5, 2018

Abstract

Most of the collaborative filtering algorithms assume that the missing values in a data set are missing at random (MAR) which is a very strong assumption because it can drastically affect the recommendation quality [1]. In this paper we compare the performance of different types of collaborative filtering algorithms with a Multinomial Mixture Model used together with the CPT-v model as proposed by Marlin, Zemel, Roweis and Slaney [2].

1 Introduction

The reason why a missing value in a data set was unobserved can be explained by the nature of it. There are three types of missing data, missing completely at random (MCAR), missing at random (MAR) and missing not at random (MNAR). MCAR states that the probability that a value is observed is independent to any sort of factor, just like flipping a fair coin. MAR assumes that the probability depends on some factors but that are not related to the value of the data itself. On the other hand, MNAR does **not** assume MAR, stating that there is a distribution over possible values of the data that affects if the value is observed or not.

This can be better understood with an example. Lets say that we were collecting data entered by the users voluntarily about their body mass index, and the data is not anonymous. Then, the observed values would probably belong in the majority to the population with a standard value of body mass index, because those that are below or above the mean would presumably not want their information to be published publicly. So this is a clearly example where the value of the data decided either it will be observed or not.

Most of the collaborative filtering algorithms assume MAR, so we will test some of them in a data set that does not hold this assumption. We will also compare the performance of them with the Multinomial Mixture Model used together with a CPT-v proposed by Marlin, Zemel, Roweis and Slaney, that was designed to deal with type of missing data.

We will refer to the authors of that study as the "authors".

Specifically, the performance of UserKnn, ItemKnn, SVD, Slope One and two different imputers will be tested because each one of them assume MAR. We expect to have worse results compared to the model proposed by Marlin et. al., given the nature of the data set to be used.

2 State of the art

Nowadays, most data sets have variables with missing data, and because they may not have the same approach, diagnosing their type is not simple. However, the most commonly used methods for finding a solution are known as deletion and imputation. The first one divides in three methods, listwise, pairwise and deleting columns, each of them removes data with missing values, but is only safe when treating with MAR and MCAR, because removing missing data in MNAR can produce a bias in the model. On the other hand, imputation consists in the substitution of this values for an estimated value based on other available information, although this introduces a bias to the results, it avoids the pitfalls involved with deletion.

3 Data set

Yahoo! Music's Launchcast Radio, was a radio station where the users could rate songs and by doing so change the order in which the songs where played. Yahoo! Webscope provided us with a data set containing ratings from users to songs during the normal interaction with the radio previously mentioned, we will refer to it as "MNAR ratings". This data set also contains information about a survey made to a selected group of the users, we will refer to it as "MCAR ratings".

3.1 MNAR ratings

This data set contains the ratings provided from 10,000 users over 1000 songs (randomly selected), where each user must have at least 10 ratings, but in average they have 21 songs rated. This is a fairly sparse matrix, where the missing values does not satisfy the MAR assumption and are in fact MNAR as we will explain further.

3.2 MCAR ratings

From the users that were present in the MNAR ratings data set, a subset of 5000 accepted to participate in a survey in which they needed to rate obligatorily 10 songs (randomly selected for each user) from the set of 1000. This generated a data set that satisfy the MCAR assumption because the ratings observed were randomly chosen and will be used to test the performance of the collaborative filtering algorithms and imputers, and was

also to train the CPT-v model used by the authors. The 5000 remaining users are used as an extra training set for the models. The **Figure 1** shows the frequency of ratings in the MNAR ratings data set compared to the MCAR ratings data set, showing clearly a different distribution between them.

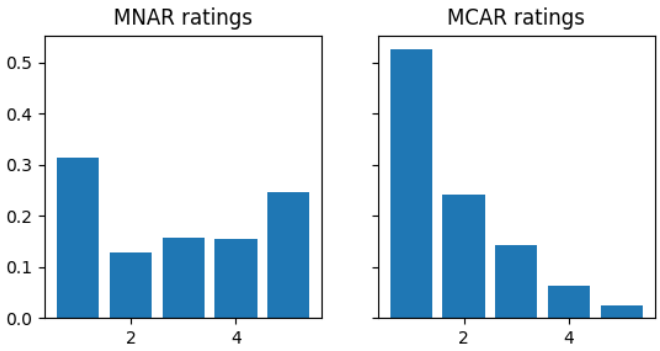

Figure 1: Distribution of ratings in the two different data sets.

The survey also contained several questions used to support the assumption that the MNAR ratings data set has missing values that are indeed MNAR. The users were asked about the frequency in which they rate a song based on their preference to it. So in other words, they were asked to quantify how likely do they rate a song they hated, loved, etc. The results are shown in **Figure 2**.

Rating Frequency	Preference Level				
	Hate	Don't like	Neutral	Like	Love
Never	6.76%	4.69%	2.33%	0.11%	0.07%
Very Infrequently	1.59%	4.17%	9.46%	0.54%	0.35%
Infrequently	1.63%	4.44%	24.87%	1.48%	0.20%
Often	12.46%	22.50%	26.83%	25.30%	5.46%
Very Often	77.56%	64.20%	36.50%	72.57%	93.91%

Figure 2: User reported frequency of rating songs as a function of preference level

4 Methodology

We used two python libraries to make recommendations. For UserKnn, ItemKnn, SVD and SlopeOne we used *pyreclab* [4], and for Iterative Imputer and Knn Imputer we used *fancyimputers* [3].

For each algorithm, we tested with different parameter values in order to get the best performance of each one in the current data set.

To measure the performance, we used MAE and RMSE mainly because the test methods provided by *pyreclab* only return this measures, so in order to do a fair comparison with the imputers, they were also evaluated with them. Next we will provide a general view of the algorithms used in this study.

4.1 Algorithms

We used Collaborative Filtering algorithms in order to make our predictions, specifically: UserKNN, ItemKnn, SVD and SlopeOne. We decided to use these in particular due to the simplicity they offered us in terms of implementation. Since we were looking to prove that treating the data set as a MNAR one instead of MAR would improve our results, there was not much importance on which recommendation algorithm to choose.

4.2 Imputers

For the imputers, the data set needed to be transformed into a sparse matrix, and then, after computing it with each method, normalize the values to match the real rating value range (1 to 5).

4.2.1 Iterative Imputer

It's a strategy for imputing missing values by modeling each feature with missing values as a function of other features in a round-robin fashion. Was inspired by the R MICE package (Multivariate Imputation by Chained Equations) [5], but differs from it by returning a single imputation instead of multiple imputations.

In simpler words, this algorithm completes each column of a sparse matrix with it's corresponding mean, and then iterates though all columns, recomputing it's previously missing values doing a linear regression with the complete data set. This process ends when the output of the linear regression converges, in other words, the difference of the column values did not change more than a threshold after computing the regression.

We chose this imputer to approach the MNAR nature of our dataset and complete the missing data according to this characteristic.

4.2.2 Knn Imputer

Such as a regular Knn algorithm this imputer looked for the nearest neighbours and classified the missing data as the most repeated one among these neighbours.

5 Results

In **Figure 1** whe display the summary of our results.

Metric	UserKnn	ItemKnn	SVD	SlopeOne	Iterative Imputer	Knn Imputer
MAE	1.18	1.22	1.18	1.2	1.38	1.56
RMSE	1.47	1.42	1.4	1.49	1.67	1.97

Table 1: Summary table of the best performance for each algorithm tested

5.1 Authors results

In comparison to the above results, the authors got better results when using the CPT-v model but similar performances when they only used the Multinomial Mixture Model, as seen in **Figure 3**.

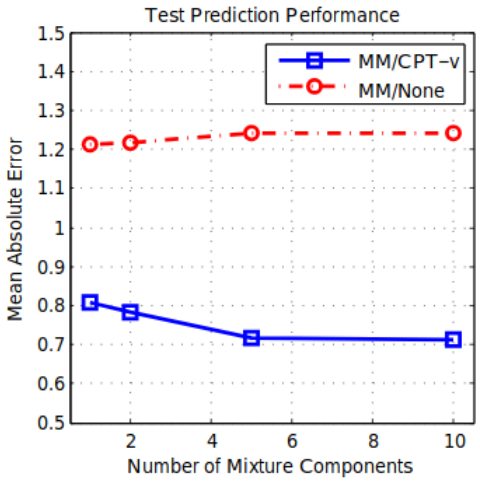

Figure 3: Best case test set prediction error for MM/CPT-v vs MM/None

6 Conclusions and future work

With the best obtained results we managed to get an error as small as the plain MM implemented by the authors, anyway the CPT-v obtained results 40% better. Given a sparse data set, its imperative to understand the nature of the missing data, so in case of being MNAR we can complete the matrix using some models as CPT-v, getting a better quality in the resultant recommendations.

For future work we came up with the idea of considering a mixture of types for the missing data. In this case we considered that our data set was MNAR, so that the missing data was a consequence of the value it would have taken, now if we think deeper we can realize that there are more reasons to the fact that a lot of the data is missing. In the Yahoo! Music data set we could consider certain factors from the users, such as: gender, age, country among others, we could also consider information about the song itself and its author, such

as: release date, amount of reproductions and many more. This way we could really know why the data is missing and predict more accurately which value it would have taken in case the user would have rated it. We think combining MNAR and MAR could improve the results and we look forward to try it out.

References

[1] Marlin et al. Unsupervised learning with non-ignorable missing data. *AISTATS*, 2005.

[2] Marlin et al. Collaborative filtering and the missing at random assumption. *UAI*, 2007.

[3] Alex Rubinsteyn. fancyimputer: A variety of matrix completion and imputation algorithms implemented in python. 2015.

[4] Gabriel Sepulveda, Vicente Dominguez, and Denis Parra. pyreclab: A software library for quick prototyping of recommender systems. August 2017.

[5] Karin Groothuis-Oudshoorn Stef van Buuren. mice: Multivariate imputation by chained equations in r. 2011.