Dauphine | PSL
UNIVERSITÉ PARIS

MASTER IASD

DATA SCIENCE PROJECT

Optimal Transport For Recommendation Systems

Realised by:

CHIKH Yanis

DJECTA Hibat Errahmen

KHEDIM Ibtissem

ESIOT Team

Supervised by:

Mr. NEGREVERGNE Benjamin

Mr. VÉRINE Alexandre

2022/2023

Table of Contents

List of Figures

1 Introduction

The recommendation system is a problem encountered in several domains. It allows filtering the information to present the objects, which can be for example movies, music, books or others, that can interest the user. In order to capture the information, we can rely on the characteristics coming from the objects or the users, or from the social environment, i.e. the links between users and objects.

In this work, we will use a very well known approach used in operational research, which is the transport theory. We will apply this approach or more precisely inverse optimal transport method to predict recommendations. Moreover, we will study the impact of the different hyperparameters on the quality of the results and adapt this method for the recommendation of films.

This report will be divided into four parts, the first one will contain the basic concepts that allow a better understanding of this work, then in the following parts, we will talk about the design, the realization and the results of our work.

2 Basic Concepts

2.1 Recommendation System

A Recommendation System is a process that seeks to predict user preferences. In other words, it is an algorithm that suggests relevant items to users. For instance, in the case of Netflix which movie to watch and in the case of e-commerce which product to buy. Technically, recommendation systems are based on two approaches: collaborative filtering and content based filtering. We are interested in collaborating filtering which is a method that aims to predict a user's preferences by aggregating the ratings and computing similarity between users.

Collaborating filtering has several advantages among them that it works well even if the data is small, the model helps the users to discover a new interest in a given item but the model might still recommend it because similar users are interested in that item. On the other hand, the mean disadvantage of collaborative filtering is the cold start problem i.e it can not handle new items because the model does not get trained on the newly added items in the database.

2.2 Optimal transport

Optimal transport is the general problem of moving one distribution of mass to another as efficiently as possible. Given two populations that we want to match, represented by their probability distribution and given a cost matrix C of moving a unit mass from μ_i to v_j . Optimal transport aims to minimize the distance $d(C, \mu, v)$ by searching of the optimal transport plan pi.

$$d(C, \mu, v) = \min_{\pi \in U(\mu, v)} < \pi, C > \tag{1}$$

Since the optimisation of this distance is computably expensive in large scale settings, in 2013 Culti introduced *Regularized Optimal Transport (ROT)*. ROT consists of minimizing the optimal transportation plan by adding a regularization parameter λ applying Sinkhorn-Knopp's algorithm.

$$d(C, \mu, v) = \min_{\pi \in U(\mu, v)} < \pi, C > -\frac{H(\pi)}{\lambda} \tag{2}$$

2.3 Inverse Optimal Transport

Since the cost of matching is not always available, researchers introduced inverse Optimal transport. By using the Sinkhorn-Knopp's algorithm we can find the minimal matching plan π with respect

to empirical matching π. So the problem takes us back to minimizing the reverse Kullback-Leibler divergence between π and π.

$$\min_A KL(\pi||\pi) \tag{3}$$

2.4 Motivation

The objective of our project is to know how optimal transport can solve recommender system problems. Based on (Li et al. 2018) work, we tried to understand their approach and adapt it to our context which is matching users and filming them through rating prediction. The approach described in the article is based on the application of two main algorithms Sinkhorn-Knopp Algorithm and Solve RIOT which uses the first algorithm. In order to demonstrate the efficiency of these algorithms, we have implemented and tested them by varying several parameters (example: relaxation parameter) and we have calculated the optimal transport plan pi of each algorithm.

3 Conception

3.1 Adapting the problem with Optimal Transport

In our use case, we use MovieLens dataset. We distinguish the two populations: movies and users. So, we will apply optimal transport to match these two populations. More details on manipulation of the dataset is given in section Dataset manipulation.

3.2 Algorithms

As mentioned above, the work is based on two algorithms: Sinkhorn-Knopp and Solve RIOT.

Algorithm 1 Sinkhorn-Knopp Algorithm

Input: marginal distributions $\boldsymbol{\mu}, \boldsymbol{\nu}$, cost matrix C, regularization parameter λ
$K = \exp(-\lambda C)$
$\boldsymbol{a} = \mathbf{1}$
while not converge **do**
  $\boldsymbol{b} \leftarrow \frac{\boldsymbol{\nu}}{K^T \boldsymbol{a}}$
  $\boldsymbol{a} \leftarrow \frac{\boldsymbol{\mu}}{K\boldsymbol{b}}$
end while
$\pi = \mathbf{diag}(\boldsymbol{a}) K \mathbf{diag}(\boldsymbol{b})$
return $\pi, \boldsymbol{a}, \boldsymbol{b}$

Algorithm 2 Solve RIOT

Input: observed matching matrix $\hat{\pi}$, cost matrices C_u, C_v, regularization parameter $\lambda, \lambda_u, \lambda_v$
for $l = 1, 2, \cdots, L$ **do**
  $Z \leftarrow \exp(-\lambda C)$
  $M \leftarrow \delta(\boldsymbol{z}\mathbf{1}^T + \mathbf{1}\boldsymbol{w}^T) \odot Z$
  Initialize $\boldsymbol{\xi}^{(0)}, \boldsymbol{\eta}^{(0)}$
  for $k = 1, 2, \cdots, K$ **do**
    $\theta_1^{(k)} \leftarrow$ root of $p(\theta)$
    $\theta_2^{(k)} \leftarrow$ root of $q(\theta)$
    $\boldsymbol{\xi}^{(k)} \leftarrow \frac{\hat{\boldsymbol{\mu}}}{(M - \theta_1^{(k)} Z)\boldsymbol{\eta}^{(k-1)}}$
    $\boldsymbol{\eta}^{(k)} \leftarrow \frac{\hat{\boldsymbol{\nu}}}{(M - \theta_2^{(k)} Z)^T \boldsymbol{\xi}^{(k)}}$
  end for
  $\boldsymbol{a} \leftarrow \frac{1}{\lambda} \log \boldsymbol{\xi}^{(k)}, \quad \boldsymbol{b} \leftarrow \frac{1}{\lambda} \boldsymbol{\eta}^{(k)}, \quad \theta = \theta_1^{(k)}$
  $\pi \leftarrow \exp(\lambda(\boldsymbol{a}\mathbf{1}^T + \mathbf{1}\boldsymbol{b}^T - C))$
  $\nabla_A \leftarrow \sum_{i,j=1}^{m,n} \lambda[\hat{\pi}_{ij} + (\theta - \delta(z_i + w_j))\pi_{ij}] C'_{ij}(A)$
  $A \leftarrow A - s\nabla_A$
  $\boldsymbol{a}_1 \leftarrow$ Sinkhorn-Knopp$(C_u, \pi\mathbf{1}, \hat{\boldsymbol{\mu}}, \lambda_u)[1]$
  $\boldsymbol{a}_2 \leftarrow$ Sinkhorn-Knopp$(C_v, \pi^T\mathbf{1}, \hat{\boldsymbol{\nu}}, \lambda_v)[1]$
  $\boldsymbol{z} \leftarrow \frac{1}{\lambda_u} \log \boldsymbol{a}_1, \quad \boldsymbol{w} \leftarrow \frac{1}{\lambda_v} \log \boldsymbol{a}_2$
end for

3.3 Class Diagram

In order to implement the adopted model, we have modelled it in the form of object-oriented classes, as illustrated in the figure 1. The main class is matcher, which has RIOT as main function.

The latter uses the traditional optimal transport algorithms, implemented in the class optimal transport.

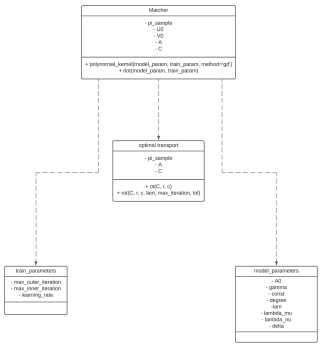

Figure 1: Class Diagram

4 Realisation

4.1 Development environment

In this part we will present the technical requirements of this project and our choices for the implementation of this solution.

We used Python programming language and we worked with Jupyter Notebook under Anaconda. The choice of this language was made due to the numerous libraries available in Python which allow to facilitate the implementation of the solution.

Among the libraries that we used, we can enumerate in particular :

- **numpy :** which is a package for manipulating multidimensional arrays and matrices.
- **pandas :** is a library that allows the manipulation and analysis of data.
- **matplotlib, seaborn :** allow to visualize data in graphical form.
- **scipy.optimize :** provides minimization (or maximization) functions for objective functions. It includes solvers for nonlinear problems, linear programming, constrained and nonlinear least squares, root finding and curve fitting.
- **sklearn :** which is a library that provides tools for data analysis and machine learning.
- **surprise :** which is a package for building and analyzing recommendation systems.

4.2 Dataset Manipulation

4.2.1 Random Dataset

For the previously described work, a small dataset had been generated to test and validate what we have implemented. As we have mentioned before, the main goal of the RIOT is to guarantee the robustness of the traditional inverse optimal transport. In the sake of that, we are going to add some noise to our random dataset to prove this kind of robustness. In the other hand, after adding noise, we will test if calibrating certain parameters will omit this noise or not. This calibration concerns relaxation parameter. To generate the latter dataset, we simulate n users and m movies then we generate randomly ratings between 0.5 and 5 to get this dataset close to the real one.

Furthermore, we need to have two probabilities vectors u0, v0, an interaction matrix, a polynomial kernel , an initial cost matrix , calculated using u,v and A, and finally the optimal transport plan π_0 to which we will add noise.

4.2.2 Movie ratings Dataset

In order to adapt our solution to our dataset, which includes the ratings that users have given to movies, we have transformed the initial matrix that includes sparse data since many users have not rated many movies. For this purpose, we grouped the users into several classes and similarly for the movies.

We opted for the K-means algorithm to perform the partitioning of users and movies. For the users we classified them according to the ratings they gave to the movies before, and for the movies we classified them according to the ratings given to them by the users.

Once the classes were obtained we constructed the new matrix by putting as rating of the class i (of the users) to the class j (of the movies) : the average of the ratings that were assigned from the users of the class i to the movies of the class j.

5 Evaluations and tests

5.1 Random dataset tests

We start this part of evaluation by testing how much RIOT can resist to noise, as shown in figure 2. Here we have a comparison of recovery performance between the traditional inverse transport and the robust version. We have recorded the Kullback Leibler divergence between the learned matching matrixes and the optimal one, with different noises used to biased the optimal matching matrix. This plot (figure 2) shows that RIOT with different relaxation parameter demonstrate the best robustness than IOT does when noise size is large. In contrast, when the relaxation parameter is too large and the noise is too small, the recovery performance is negatively effected. As a result, we can conclude that when information is available about noise, the relaxation parameter can be tuned properly to that.

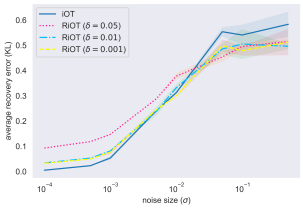

Figure 2: Comparison of recovery performance between IOT and RIOT according to relaxation parameter

In addition to that, we have recorded the recovery performance of the IOT and RIOT variating the regularization parameter to see its effect. According to (figure 3), we figure out that this parameter hasn't a significant effect on the recovery performance of the two approaches. We notice also that the performance of the RIOT always remains better than the IOT's performance.

Figure 4 illustrates matching matrixes (π_0, π_{opt}, π_{IOT}, π_{RIOT}). It's visually clear that the initial matching matrix is significantly changed due to noise. In the other hand, after the execution of the two approaches, it can be seen that the π learned by RIOT is more close to the optimal one than the matrix learned by the traditional IOT. Therefore, this is another proof that RIOT is more robust to noise than IOT.

As in figure 4, figure 5 shows cost matrices (C_0, C_{IOT}, C_{RIOT}). We see that approved approach

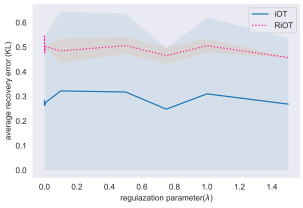

Figure 3: Comparison of recovery performance between IOT and RIOT according to regularization parameter

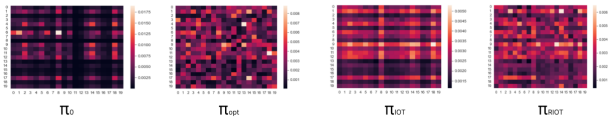

Figure 4: Comparison of matching matrixes

RIOT can learn the cost matrix structure better than the IOT.

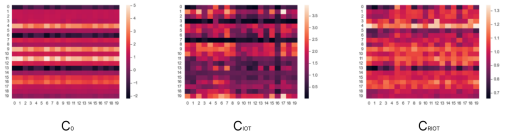

Figure 5: Comparison of cost matrixes

To recapitulate, RIOT has the ability to guarantee better recovery performance with the appropriate tuned relaxation parameter. Also, it has an approved learning capacity by revealing the original structures of cost and matching matrixes.

5.2 Movie ratings dataset tests

The results of the evaluation and comparison of robust inverse optimal transport with other algorithms like : positive matrix factorization, singular value decomposition, random generation and k nearest neighbor, with root mean squared and mean absolute error measures, show that RIOT beats other algorithms as we can see on the graphs (figure 6, figure 7).

The comparison clearly shows that Riot captures information better than the other methods and makes better predictions. In addition, we note that for the other methods, the larger is the dataset, the worse are the results, unlike RIOT which learns better with a larger dataset.

6 Conclusion

Recommendation systems and optimal transport are two concepts that become essential in many industries and, hence, have received always more and more attention in recent years. In this work, we have adopted a framework of matching for recommendation systems proposed in (Li et al. 2018) and which is based on inverse optimal transport. We have verified the performance of the proposed RIOT and our tests confirm the paper results. These results prove the robustness of RIOT and its learning capacity. In addition to the tests performed by the paper, we have tested the effect of the regularization parameter. Moreover, RIOT can be applied on real word dataset, and it gives results better than other algorithms used in the same context. This dataset needs a lot of manipulations before it can be used. Future works can concern the minimization of these manipulations by adopting more sophisticated methods.

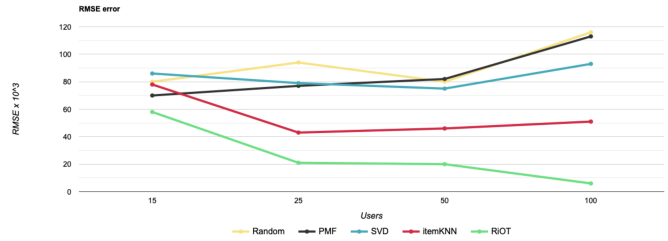

Figure 6: Comparison of RMSE error according to Dataset size

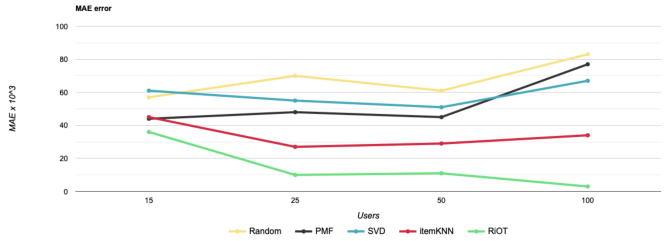

Figure 7: Comparison of MAE error according to Dataset size

Bibliography

Li, Ruilin et al. (Feb. 2018). 'Learning to Recommend via Inverse Optimal Matching'. In.