THERAPY RECOMMENDATION FOR PATIENTS[*]

A recommendation system for patient who needs the treatment - part of Data Mining course[†]

Manh Tuan NGUYEN[‡]
Master Student
Second year of Artificial Intelligence
Avignon University - Trento University
manhtuan.nguyen@studenti.unitn.it

ABSTRACT

The technology is advancing rapidly thus lead to the high demand for life quality and raising attention to living standard. One of the biggest concerns nowadays is healthcare, especially during the Covid crisis. Hospitals are becoming overload due to a lot of intense Covid-19 case around the world. This brings a big trouble to other patients because if they have some sickness but it's really a dangerous situation if they go directly to the hospital for the doctor's examination and may expose to a Covid threat. This project proposes a demonstration of how Data Mining can be applied to make the treatment recommendation to the patients without the need of meeting face to face. This report brings 2 solutions to turn this idea into reality. Both solutions is based on *Collaborative Filtering*. This project is a part of *Data Mining Course* of *Prof.Velegrakis*

[*]Produces the permission block, and copyright information
[†]The full version of the author's guide is available as `acmart.pdf` document
[‡]Manh Tuan NGUYEN

1 INTRODUCTION MOTIVATION

Healthcare has been and is one of the most important areas of life. Throughout history, mankind has achieved certain successes in this field, for example: antibiotics, vaccines, ... However, right now , humanity is facing one of the most dangerous and complicated outbreak - *Covid-19 virus*. This pandemy has affect a lots aspect in every country. One of that is the overload in every healthcare facility such as hospital,... Many and many countries now don't have enough medical team to deliver the treatment not only to the Covid patient but also others. This has become a medical dilemma which needs to be solved. Thanks to the development of technology, we can now apply techniques to build a system that supports diagnosis and treatment.Thus, in the present project, I'm proposing 2 Data Mining solutions. Both of them based on the *Recommendation system lecture*.From now the first solution will be called **General Collaborative Filtering** which based on the idea of Collaborative Filtering and the second solution is **Hybrid Method**. Both methods is built based on the problem that we have to face with some *cold-start items* which we don't have any information.

2 RELATED WORK

During this project, I have applied some techniques/methods which has been seen during the course. This part will briefly describe each techniques/methods I used

Cosine Similarity: This similarity measures the different between 2 vectors. The value of Cosine similarity is a real number between -1 and 1. The bigger the value, the similar the 2 vectors. Inside this project, this will be applied to measures the different between 2 items vector so from that we can make a content-based item-item recommendations.

Utility Matrix: A matrix that re-present the "known" rating between users-items pair. This matrix plays an important roles in both solutions are mentioned above. Inside the context of project, it is a challenge to decide which features should be used to calculate the matrix.

Collaborative Filtering: A method uses to deal with interact data between user and item. This method will fill all the missing value inside utility matrix and then from that we can make the recommendation based on the matrix

Content-Based: A method uses to deal with underact data such as features of items or features of users. From that, we can make the recommendation by finding all the similar user/item to the query one. This method inside project uses to deal with cold-start problem

Hybrid Method: The final solution which combine both *Collaborative Filtering and Content-Based Method.*

3 PROBLEM STATEMENT

Inside this project context, after do the research on the features, I found out that only **Condition** and **Therapy** have important information that can perform the recommendation. From that, I divided the solution just by changing the user-item pair. Both solutions follow strictly this formal model form:

$$\mathbf{X} * \mathbf{S} -> \mathbf{R} \tag{1}$$

Where:

$$\mathbf{X} = \text{Set of users}$$
$$\mathbf{S} = \text{Set of items}$$
$$\mathbf{R} = \text{Set of rating}$$

3.1 General Collaborative Filtering

With this solution, I decided to pick **Condition** (ex: Cond25 - the general condition) as the set X, **Therapy** (ex: Th25) as the set S and the **successful rate** (scale 0-1) as the set R

3.2 Hybrid Method

With this solution, I decided to pick **Condition-PatientsID** (ex: pc1 - the condition of each patient) as the set X, **Therapy** (ex: Th25) as the set S and the **successful rate** (scale 0-1) as the set R

4 SOLUTION

I will present this part for each solutions I have.

4.1 General Collaborative Filtering

With this solution, first I need to create an utility matrix. So first I will received all the **conditions** object and all the **trials** object inside the patients records. Next I will perform the inner merge between two set and group them by the pairs **General Condition - Therapy**. So inside this utility matrix we will have in total 16422 records since we have total 322 Conditions and 51 Therapy. For the rating value, I will calculate the mean of all successful rate by pairs **General Condition - Therapy** (because 1 General Condition can have many different trials). Due to that this method based on the general Condition, this utility matrix is fully filled with the rating which we don't need to apply the rating prediction of Collaborative Filtering. With this Utility Matrix we have, with each test case input inside the system will be look-up to the Conditions table. When we have the General Conditions ID of the test case, query directly to the Utility Matrix and take out top 5 successful rate therapy as the recommendation.
I came up with this method because by simple thinking, normally each condition should only has limited amount of therapy and will not be too depend just by the race, nationality etc of the patient.

4.2 Hybrid Method

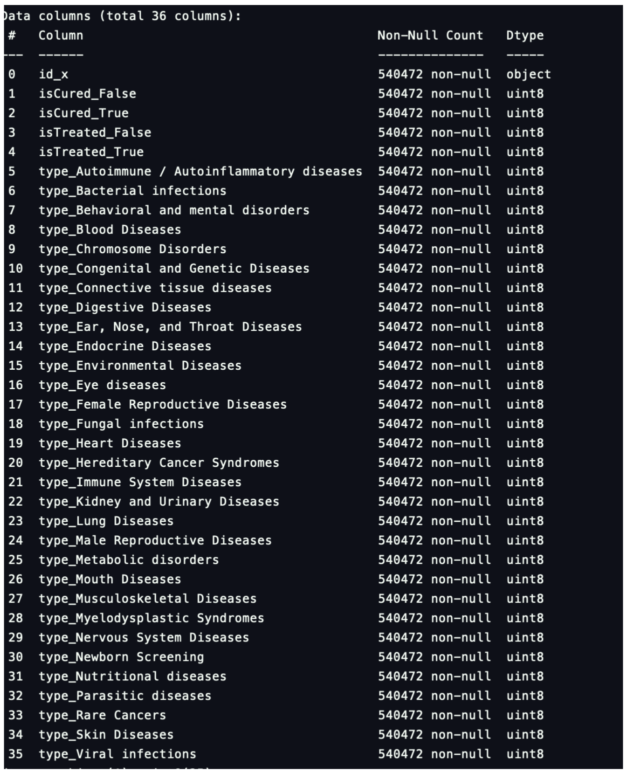
With this method, it requires more processing than the previous method. With this method, we also need to create the utility matrix. A little bit different with above like I already described in **Problem Statement** section, this utility matrix is the pair between specific condition of each patients and the therapy.
So first, I also extracted the *conditions and trials* object inside patients records. From that, I will merge them together and from that we have our matrix which contains userIDs as conditionIDs and itemIDs as Therapy. Because each specific condition has corresponding therapy with the successful rate. So we don't need to calculate anything. From this, I will create a sparse matrix (because for each specific condition, most likely the patient cannot try all the therapy to cure that so we still missing a lot of successful rate - and we call this the sparse matrix).The successful rate - rating will be scale back to [0,1] scale.
Beside of the **sparse rating matrix**, I also created the **sparse features conditions matrix**. To create this, I applied the *one-hot encoding* to the *type* features of each specific conditions. I only take to account the type because with others features such as *isCured or isTreated* will not be good if we put inside this since after we make the recommendation base on this, the system only suggested all the same type and same non-treated conditions (which is not solved the cold-start problem).

After we have all the necessary components, we begin to implement the first part of *Hybrid model* which is *Collaborative Filtering*.There are to 2 ways to implement this part:

- **Memory-based:** this is refer to as neighborhood based collaborative filtering.
- **Model-based:** Where we applied the Machine Learning technique to learn model parameter. This will be our solution since the model will have the capability to learn the features embedding of the Conditions which makes it's easier to develop the hybrid model.

After we have the model, we also need to build the *Content-based* part. By receiving the features embedding from the model, we can perform the cosine similarity to make the item-item recommendations follow this equation:

$$Sim_{A,B} = \frac{A * B}{||A||.||B||} \tag{2}$$



```
ata columns (total 36 columns):
 #   Column                                  Non-Null Count   Dtype 
--   ------                                  --------------   -----
 0   id_x                                    540472 non-null  object
 1   isCured_False                           540472 non-null  uint8
 2   isCured_True                            540472 non-null  uint8
 3   isTreated_False                         540472 non-null  uint8
 4   isTreated_True                          540472 non-null  uint8
 5   type_Autoimmune / Autoinflammatory diseases  540472 non-null  uint8
 6   type_Bacterial infections               540472 non-null  uint8
 7   type_Behavioral and mental disorders    540472 non-null  uint8
 8   type_Blood Diseases                     540472 non-null  uint8
 9   type_Chromosome Disorders               540472 non-null  uint8
 10  type_Congenital and Genetic Diseases    540472 non-null  uint8
 11  type_Connective tissue diseases         540472 non-null  uint8
 12  type_Digestive Diseases                 540472 non-null  uint8
 13  type_Ear, Nose, and Throat Diseases     540472 non-null  uint8
 14  type_Endocrine Diseases                 540472 non-null  uint8
 15  type_Environmental Diseases             540472 non-null  uint8
 16  type_Eye diseases                       540472 non-null  uint8
 17  type_Female Reproductive Diseases       540472 non-null  uint8
 18  type_Fungal infections                  540472 non-null  uint8
 19  type_Heart Diseases                     540472 non-null  uint8
 20  type_Hereditary Cancer Syndromes        540472 non-null  uint8
 21  type_Immune System Diseases             540472 non-null  uint8
 22  type_Kidney and Urinary Diseases        540472 non-null  uint8
 23  type_Lung Diseases                      540472 non-null  uint8
 24  type_Male Reproductive Diseases         540472 non-null  uint8
 25  type_Metabolic disorders                540472 non-null  uint8
 26  type_Mouth Diseases                     540472 non-null  uint8
 27  type_Musculoskeletal Diseases           540472 non-null  uint8
 28  type_Myelodysplastic Syndromes          540472 non-null  uint8
 29  type_Nervous System Diseases            540472 non-null  uint8
 30  type_Newborn Screening                  540472 non-null  uint8
 31  type_Nutritional diseases               540472 non-null  uint8
 32  type_Parasitic diseases                 540472 non-null  uint8
 33  type_Rare Cancers                       540472 non-null  uint8
 34  type_Skin Diseases                      540472 non-null  uint8
 35  type_Viral infections                   540472 non-null  uint8
```

Figure 1: Information about the sparse features conditions matrix

After we have all the combination, first we will use the model to predict the top 5 rating for the desired test case, if that test case is a cold-start one (Which not present inside rating matrix) , we will pass it to Content-based part to return the most similarity condition (which is inside the rating matrix). From that, we perform again the prediction of the model to make the recommendation based on the most similarity condition to the test case.

5 IMPLEMENTATION

To implement the theory to the code, I decided to use the *Python language* along with different libraries for different purposes. I will list the libraries here:

- **scrapy:** This library help to create the data set. It's specialized on crawling data from website by the html or css component. With this, I crawled all the condition and therapy name from the link's provided by professor Velegrakis.
- **json:** This library help to create the data set in the json format
- **Pandas:** This library is used for dealing with the data set. Perform various method such as merge, groupby ,... in order to deliver all the data set in correct form for further processing phase.
- **Scipy:** This library is one of most famous library to perform scientific computing method. I used this specially to create the sparse matrix from the DataFrame type of pandas library.
- **Numpy:** The most famous computation library. I used this to implement the equation of cosine similarity, sort the recommendation list by rating,...
- **Scikit-learn:** A Machine Learning library which includes many interesting function. I used the *MinMaxScaler()* function to scale the successful rate form [0,100] to [0,1].
- **lightFM:** The core library of my project. This library is famous for it speciality in building recommendation system. It provides the option to create the Hybrid model.

5.1 Dataset

This section is indicate about the creating dataset process. To create the dataset, first I have to have enough information such as condition name, therapy name , human name,... As professor Velegrakis mentioned before, to create dataset we don't have to think as specialist just think as data scientist. So for that, the dataset is purely random.
To craw information, I used the library scrapy and intergrate with the css component of the source website, from that I will gather all necessary information and stored it in csv file in case of reusable (if anyone else want to create dataset, they don't need to craw again).
Then, for each object (patients, condtions and therapy) , I created a list of multiple dict that contain all the information follow the description of the project. For Conditions and Therapy, the structure is simple, but for patients it's quite complicated so I will explain it detail here. The patients is a list of many dict (each dict is each object of patients) inside dict of patients, you have all the key, for the key conditions and key therapy, the value will be a list (now call condition list and therapy list) for condition and therapy list, inside that you have many dict (which present a object of a conditions patients have / a therapy they take).
All the choosing component is purely random due to the lack of specialist inside this project.

6 EXPERIMENTAL EVALUATION

We have two methods need to be validated.

6.1 General Collaborative Filtering

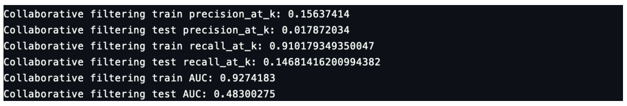
This method is quite not really an recommendation system method due to the matrix is already fulfilled. But from that, I still thinking that I can perform the precision at k metric. With this, I will count all the therapy which cured the desired general conditions and to see how many of them inside the recommendation list and divide it by the size of the list (k = 5). But to be fair, this metric can be applied but after performed it, I can found that it cannot be true with the random dataset since most likely, all the general conditions can have up to 20 therapies to cured them (which is not be the case in real life). Due to that, the precision depend on the size of dataset can get up to 1.0 . But along side with that, our recall metric will be really low since the size of relevant items with the size of recommendation list has a big gap.

But in my opinion, it still can be a good solution if that can be applied in the real life. Because most likely all the sickness can only has some therapies only despite the characteristic of each patient.So this method can be fast applied and also avoid the cold-start problem.

About the time execution, with the help of pandas, the execution to create matrix and query the recommendation only take around 0.3s and can get down to 0.1 after the first run (since the matrix can be save to disk and reuse after without the creation).

6.2 Hybrid Method

With this method, we can apply many metrics to evaluate the model. I already applied precision , recall and also the AUC metrics. The result of precision and recall is not telling much since when I researched the data, the average interaction of specific conditions to a therapy is 1.83 (that means 1 conditions normally patients only try around 2 therapy to treat that). That will make the relevant set is too small to perform the precision and recall at k (the bigger the k, the lower the precision). For the AUC metrics, it's the probability that a randomly chosen positive example has a higher score than a randomly chosen negative example.This will be a good metrics to evaluate the model in my opinion. With this metrics, I got 0.92 for train set and 0.48 for the test set (train-test ratio is 80-20) .



```
Collaborative filtering train precision_at_k: 0.15637414
Collaborative filtering test precision_at_k: 0.017872034
Collaborative filtering train recall_at_k: 0.910179349350047
Collaborative filtering test recall_at_k: 0.14681416200994382
Collaborative filtering train AUC: 0.9274183
Collaborative filtering test AUC: 0.48300275
```

Figure 2: Evaluation result Hybrid method

The time execution of this method is also fast, in total every phase from creating matrix, train model , test model and making the prediction for the test case, it only took 26.74s

7 ACKNOWLEDGMENTS

I would like to express my gratitude to all those who gave me possibility to complete this project in specific as well as academic supporting in general that I received during the time of our studying at the Trento University. I wish to express my sincere thanks to Professor Yannis Velegrakis for providing us with all the necessary facilities for the research.

ACKNOWLEDGMENTS

To Robert, for the bagels and explaining CMYK and color spaces.

REFERENCES