International Journal of Computer Applications (0975 - 8887)
Volume - No.,

Therapy Recommendation Systems for Patients

Data Mining Course by Prof. Velegrakis

Ali Akay [224414]
Trento University
MSc.Artificial Intelligence Systems
ali.akay@studenti.unitn.it

ABSTRACT

In this project, I designed a recommendation systems solution using Collaborative filtering (CF) predicts patients preferences in item selection based on the known patients therapy successful rate in order to help doctors to suggest next therapy for a patient to try a specific condition he or she may have for the dataset that we created for the project.
8

1. INTRODUCTION

Recommendation systems (RS) which use data mining and information filtering techniques to provide products, services and information to potential customers have attracted a lot of attention of researchers. The recommendation systems can be divided into three main branches according to different strategies – collaborative filtering, contend-based and the combination of both [1]. Among of them, the collaborative filtering is the most widely used since it has the advantage that does not require the creation of explicit profiles. CF can be divided into two main branches: memory-based and model-based. Most of the present researches improve the accuracy of Memory-based algorithms only by improving the similarity measures. But few researches focused on the prediction score models which we believe are more important than the similarity measures. The most well-known algorithm to model-based is the matrix factorization. Compared to the memory-based algorithms, matrix factorization algorithm generally has higher accuracy. However, the matrix factorization may fall into local optimum in the learning process which leads to inadequate learning. CF approaches are usually designed to provide products to potential customers. Therefore the accuracy of the methods is crucial.

2. RELATED WORK

2.1 Collaborative Filtering Method

Collaborative filtering predicts user preferences in item selection based on the known user ratings of items. As one of the most common approach to recommender systems, CF has been proved to be effective for solving the information overload problem. CF can be divided into two main branches: memory-based and model-based. Most of the present researches improve the accuracy of Memory-based algorithms only by improving the similarity measures.
The goal of a collaborative Filtering algorithm is to suggest new items or to predict the utility of a certain item for a particular user based or item based on the user's/items previous likings and the opinions of other like-minded users/items. In a typical CF scenario, there is a list of m users $U = u1; u2....um$ and a list of n items $I = i1; i2....in$. Each user ui has a list of items Iui, which the user has expressed his/her opinions about.It makes sense to normalize the utilities by subtracting the average value for a patient.That way, we get negative weights for items with a below-average successful rate, and positive weights for items with above-average successful rate.

Memory-based Collaborative Filtering Algorithms

Memory-based algorithms utilize the entire user-item database to generate a prediction. These systems employ statistical techniques to find a set of users, known as neighbors, that have a history of agreeing with the target users or item.(i.e., they either rate different items similarly or they tend to get similar set of items). Once a neighborhood of users is formed, these systems use different algorithms to combine the preferences of neighbors to produce a prediction or top-N recommendation for the active user.
One fundamental difference between the similarity computation in user-based CF and item-based CF is that in case of user-based CF the similarity is computed along the rows of the matrix but in case of the item-based CF the similarity is computed along the columns, i.e., each pair in the co-rated set corresponds to a different user. Computing similarity using basic cosine measure in item-based case has one important drawback|the differences in rating scale between different users are not taken into account.

User Based Collaborative Filtering

User-user method tries to identify users with the most similar "interactions profile" (nearest neighbours) in order to suggest items that are the most popular among these neighbours (and that are "new" to our user). This method is said to be "user-centred" as it represents users based on their interactions with items and evaluate distances between users.

Item-to-Item Collaborative Filtering

Item-item method is based on the search of similar items in terms of user-item interactions. To make a new recommendation to a user, the idea of item-item method is to find items similar to the ones the user already "positively" interacted with. Two items are considered to be similar if most of the users that have interacted with both of them did it in a similar way. This method is said to be "item-centred" as it represent items based on interactions users had with them and evaluate distances between those items.

Advantages over user-based CF;

Stability : Items ratings are more stable than users ratings. New ratings on items are unlikely to significantly change the similarity between two items, particularly when the items have many ratings
Scalability : with stable item's ratings, it is reasonable to pre-compute similarities between items in an item-item similarity matrix (similarity between items can be computed offline). This will reduce the scalability concern of the algorithm. [2]
The algorithm that defines item-based CF is described as follow [3]
First identify the k most similar items for each item in the catalogue and record the corresponding similarities. To compute similarity between two items we can user the *Adjusted Cosine Similarity* that has proven to be more efficient than the basic *Cosine similarity* measure used for user-based collaborative as described in :

$$w_{u,v} = \frac{\vec{r}_u \cdot \vec{r}_v}{\|\vec{r}_u\|_2 * \|\vec{r}_v\|_2} = \frac{\sum_{i \in I} r_{u,i} r_{v,i}}{\sqrt{\sum_{i \in I}(r_{u,i})^2}\sqrt{\sum_{i \in I}(r_{v,i})^2}} \quad (1)$$

To produce top-N recommendations for a given user that has already purchased a set of items, do the following :

—Find the set C of candidate items by taking the union of all

$$S^{(i)}, \forall i \in I_u \quad (2)$$

and removing each of the items in the setI_u

—

$$C = \bigcup_{i \in I_u} \{S^{(i)}\} I_u \quad (3)$$

—Compute similarity between c and the set as follows:

$$w_{c,I_u} = \sum_{i \in I_u} w_{c,i}, \forall c \in C \quad (4)$$

The last step is to calculate the items rating. The rating is computed by a weighted average of the ratings by the neighbors.

$$\hat{r}_{u,i} = \frac{\sum_{i \in S^{(i)}} r_{u,j} \cdot w_{i,j}}{\sum_{j \in S^{(i)}} |w_{i,j}|} \quad (5)$$

2.2 Evaluation Metrics

In order to verify the accuracy of the algorithms,Mean Absolute Error (MAE) and Root means quared Error (RMSE) metrics is used as evaluation metrics. The MAE and RMSE were two quantity used to measure how close forecasts or predictions are to the eventual outcomes in statistics.In our case it is successful rate of the therapy on a condition.

$$rmse = \sqrt{(\frac{1}{n}) \sum_{(u,i)=1}^{n} (r_{ui} - r'_{ui})^2} \quad (6)$$

3. PROBLEM DEFINITION AND DATASET

In this section,Firstly I would like to talk about the formal definition of the project then I mention about how I collect dataset and how pre-processed the dataset.

3.1 Problem Definition

Let set up following definition:

—P = Set of Patients
—C= Set of Conditions
—T = Set of Therapy trails for the spesific condition of the patient
—r = Successfull rate of the trails

The idea is to create a recommendation systems that identify some therapies based on specific condition of the patient may have a trial.
In other words;

(1) pi < C [c1,c2,...cn] > A set P of patients with their conditions
(2) pici < t1,t2,...tn > A set T of trials that have tried on spesific condition of the patient.
(3) Successful rate of the trials

As a output of the problem is

—Set of Therapy

As input,we have patient id and patient conditions and list of trails of the conditions.It might be more than one trial for a certion condition of the patient.

Every trails of the condition for each patient also has successful rate r which I use also in utility function.

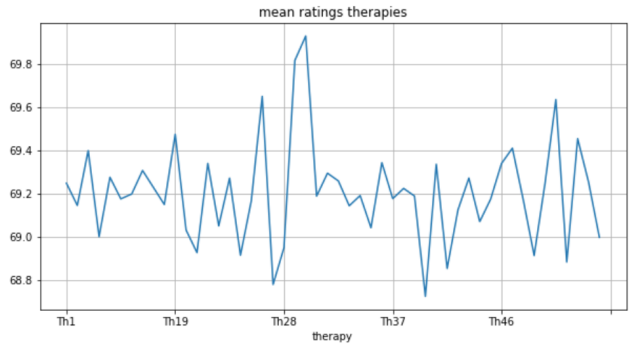

Fig. 1. Mean Rating of Therapies

Given input such as patient id and condition, the model has to compute an ordered list of ten recommended therapies based on trails successful rate for the conditions of patient.

So our utility function is;

$U : Pi[ci] \times ti -> ri$

3.2 Dataset

First of all,the dataset of the project is artificial and dataset has three table in json file which are conditions,therapies and patients.Conditions and therapies extracted using web scrapper tool which is extension of google chrome then exported to the excel file.You can reach the **website** for condition and **website** for therapies.After therapies and conditions are extracted,*patients* data created from artificial names database in order to combine therapy and condition dataset with patients.

—<20000> Number of Patient
—<1210> Number of Therapy
—<318> Number of Condition is used to create dataset.

For each patients in json file which you can see example of the patient.

```
{'Conditions': [{'id': 'Cond1',
   'name': 'Abdominal aortic aneurysm',
   'type': 'Blood Diseases'},
  {'id': 'Cond2', 'name': 'Acne', 'type': '
      Newborn Screening'}}
{'Patients': [{'age': 57,
   'birthdate': '19640714',
   'blood_group': 'O+',
   'conditions': [{'cured': 'Null',
     'diagnosed': '20080609',
     'id': 'pc1',
     'isCured': False,
     'isTreated': False,
     'kind': 'Cond46'},
    'trials': [{'condition': 'pc3',
     'end': '20120109',
     'id': 'tr1',
     'start': '20111219',
     'successful': 86,
     'therapy': 'Th49'}}
'Therapies': [{'id': 'Th1', 'name': 'abortive
    therapy', 'type': 'Physio'},
  {'id': 'Th2', 'name': 'antibody therapy', 'type
      ': 'Physio'}}
```

condition	end	id_x	start	successful	therapy	cured	diagnosed	isCured	isTreated	kind	id
pc3	20120109	tr1	20111219	86.0	Th49	20120404	20111218	True	True	Cond240	0
pc3	20120217	tr2	20120203	10.0	Th45	20120404	20111218	True	True	Cond240	0
pc3	20120404	tr3	20120330	100.0	Th45	20120404	20111218	True	True	Cond240	0
pc4	19650727	tr4	19650714	100.0	Th17	19650727	19650601	True	True	Cond39	0
pc5	19731019	tr5	19730919	100.0	Th47	19731019	19730915	True	True	Cond309	0

Fig. 2. Pre-processed dataset

As you can see here we have that list of trials of the condition of the patient has features [Fig. 1.];

—id = Patient id
—kind = Condition id
—diognosed,cured = Condition start and end date
—isCured = Wheather condition is cured or not
—therapy = Therapy id
—successful = Successful rate of the trail for the condition id
—start,end = Trail start and end date
—isTreated = Wheather condition is cured or no

In baseline model and my algoritm,I use only *[id,kind,Therapy,Succesful]* columns. [Fig. 1.]

3.3 Pre-Processing

Pre-processing is basically converting dataset json file to the dataframe that I will use for the modelling part.Patient dataset is pre-processed in for loop for each patients in order to make it more structured for the model.

It is basically combine each patient record for conditions and therapy to the dataframe.Here is the for loop that I used for pre-processing.

Algorithm 1 Convert Json to Dataframe

Require: $df[]$
for <i in len(patients)> **do**
<p=patients[i:i+1][['id']] >
<t=pd.DataFrame(patients["trials"][i])>
<c=pd.DataFrame(patientss["conditions"][i])>
<tc=t.merge(c, lefton="condition", righton="id")>
<ptc=pd.concat([tc,p], axis=1)>
<df=pd.concat([df,ptc],axis=0)>
end for

Here in the for loop, I define a empty dataframe and for each patient id, I get their therapy and condition records

and concatenate it with the dataframe that I created in axis 0.
In the end of the for loop, I got my pre-processed dataset with shape of $(1027030, 12)$ which you can see in [Fig. 1.]

4. SOLUTION

In this section,I present the user-based KNN inspired baseline model by using python toolkit and implementation of my solution which is item-based model from scratch.

4.1 Baseline Model

There are quite a few libraries and toolkits in Python that provide implementations of various algorithms that I can use to build a recommender. But I prefer to build baseline model with Surprise toolkits in order to also understand recommendation systems.
The entire process of user-to-user CF algorithm is described as follow:
For an active user u_i,

(1) First identify the set G_u of k most similar patient. G_u is the group patients similar to the patient . The similarity between two patients u and v can be measured by the cosine similarity.$w_{u,v}$ is the degree of similarity between patients u and v. This term is computed for all , where is the set of all patients.
(2) Find the set C of candidate therapy, already tried by the group and not triend by the active patient u. Candidate therapies have to be the most frequent therapies already tried by the group of patients.
(3) Aggregate succescull rates of patients in G_u to make predictions for patient u on therapies he/she has not already tried. Several aggregation approaches are often used such as average, weighted sum, ajusted weighted sum. By using weighted sum, the predicted rating of therapy u on therapy i denoted by $\hat{r}_{u,i}$ is computed as follow :

$$\hat{r}_{u,i} = \bar{r}_u + \frac{\sum_{v \in G_u} (r_{v,i} - \bar{r}_v) \cdot w_{u,v}}{\sum_{v \in G_u} |w_{u,v}|}. \quad (7)$$

(4) Ratings of similar patients are weighted by the corresponding similarity with the active patient. Summation are made over all the patients who have rated therapy i . Subtracting the patient's mean rating $\bar{r}_v$ compensates for differences in patients' use of the rating scale as some patients will tend to give higher ratings than others [2]. This prediction is made for all therapies $i \in C$ not tried by patient
(5) The Top-10 recommendations are obtained by choosing the 10 therapies which provide most satisfaction to the patient according to prediction.

4.2 Item based Model

The first step of collaborative filtering algorithm is to obtain the therapy profile, which can be represented as a ratings matrix with each entry the succesful rate of a therapy given to patients.A ratings matrix consists of a table where each row represents a therapy, each column represents a patients, and the number at the intersection of a row and a column represents the patient successfull rate value.

In our problem,let be p the patient and the referenced therapy i, which is in the literature and papers users and items are used which in our case **patient is user and therapy is item**. I also used users and items in the formulas as literature.

—If p succesfully recovered with therapies similar to i , he/she will probably get therapy i
—If he/she doesn't recovered with therapy similar to i , he/she will also hate therapy i_j
—The idea is therefore to look at how an patient p rated therapy similar t to know how he/she would have rated therapy i

4.2.1 Step 1. Find similarities for each of the therapies. To compute similarity between two therapy i and j , I follow this path:

(1) Find all patients who rated both of them
(2) Normalize their successful rate on therapy i and j
(3) Apply the cosine metric to the normalized successful rate to compute similarity between i and j

4.2.2 Step 2. Similarities computation. Similarities between items can be measured with the Cosine distance. The Nearest Neighbors class from the sklearn library in python simplifies the computation of neighbors.

The above method, the following nearest neighbors method uses the created model to kNN items. It returns nearest neighbors as well as similarities measures for each items.

Nearest neighbors returns :

—similarities : Distances of the neighbors from the referenced patient / numpy array of shape (n, k)
—neighbors : Neighbors of the referenced patient in decreasing order of similarities / numpy array of shape (n, k)

where n is the total number of therapy and k is the number of neighbors to return, specified when creating the kNN model.

4.2.3 Finding candidate items. To find candidate therapy for patient , I follow this path:

(1) Find the set I_u of therapy already rated by patient u
(2) Take the union of similar therapy as C for all therapy in I_u
(3) exclude from the set C all therapy in I_u, to avoid recommend to a patient therapies he/she has already occured.

In other words my solution can be define as in this several steps

(1) Normalize each successful rate by substracting the mean successful rate of the corresponding patient
(2) Represent each therapy by a vector of its normalized successful rate in Rating Matrix
(3) Create the nearest neighbors model with the cosine similarity metric and get similarities and neighbours vectors.(20 Neighbours)
(4) Find the set I_u of therapy already rated by patient in order to find candidate therapies C
   (a) Take the union of similar therapy for all therapies in I_u to form the set of candidate therapies
   (b) Exclude from the set C all therapies in I_u.
(5) Compute similarity between an candidates therapies C and a set of therapy I_u. For each therapy i in I_u, get similarity between i and c, if c exists in the set of therapy similar to therapy i.
(6) Rank candidate therapies according to their similarities with i_u
(7) Get the first N row of ranked candidates to build the top N recommendation list

4.2.4 Step 3. Rating prediction. The most important step in a collaborative filtering system is to generate the output interface in terms of prediction.Once I isolate the set of most similar therapy based on the cosine similarity measure,the next step is to look into the target patient ratings and use a technique to obtain predictions.
In our case,as stated earlier the predicted rating for a given patients on a therapy is obtained by aggregating successful rate given by on therapy similar to.
Here is the steps for rating prediction;

(1) Get therapies similar to therapy i with their corresponding similarities
(2) Get Successful rate of patient with id patient id
(3) Find similar therapies $W_{i,j}$ rated by therapy i of the patient
(4) Calculate the predicted successful rate according to rating formula (5)

5. EXPERIMENTAL EVALUATION

In order to validate solutions,two kind of experiment have been run.Firstly,Experiment have been done with;

(1) Full Data
(2) Patients selected who had at least one trial for same condition and whose trial had ended with a higher successful rate

Second experiment have been run in order to understand how successful trial effect to algorithms.Because for some patients there are some cases that for same condition-trial with different successful rates.So it means that the order of the trail could be also important for the recovery of the patient.

As you can see in the result table,Item-Based model gives the worst result for the second experiment.To quantify the performance,RMSE metric is used.

Dataset	UB KNN Baseline	UB KNNwithMeans	Item-Based
A	34.03	**34.00**	36.38
B	34.06	**34.04**	80.35
	SVD	SVD++	Item-Based
A	**33.50**	45.66	36.38
B	**34.48**	46.66	80.35

In order to compare baseline and my implementation results, we randomly split dataset into portion of 0.04 of test set which is 41082 data extracted.For the baseline models,used-based(UB) KNN-inspired model with mean rating and baseline rating used.

KNNBaseline:A basic collaborative filtering algorithm taking into account a baseline rating.
KNNwithMeans:A basic collaborative filtering algorithm, taking into account the mean ratings of each user.
For the second comparison I also implemented matrix-factorization based and slone one algorithms using Suprise toolkit.In this table you can see the results with baseline parameters.
SVD:The famous SVD algorithm, as popularized by Simon Funk during the Netflix Prize. SVD++: An extension of SVD taking into account implicit ratings.
According to results, we can see that baseline SVD model gives the lowest RMSE metric.Cold start It could be problem for the item-to-item collaborative filtering approach because model cannot extract any rating for the patient condition if there is no trial applied yet and this means that there are is no successful rate for that reason,I predict successful rate as the mean value of the therapy in my item-based implementation.

6. FUTURE WORK

User-based and Item-based CF are memory based algorithms. They directly act on the user-item interactions to compute recommendation. To the contrary, model-based algorithms are mathematical models trained on the user-item interactions and used to predict recommendation.Model-based algorithms are different with Memory-based algorithms. It first uses the database to estimate or learn a model and then apply this model for prediction. Generally speaking, the Model-based algorithms usually have higher accuracy than the Memory-based algorithms. We could get better score and recommendation using model-based algorithms.

Hybrid recommender systems by combining each strategy together can provide better performance rather than either strategy alone [4]. The most famous is the BellKor's solution winning the Netflix prize [5], which combines

predictions from 107 different baseline recommender systems.

7. CONCLUSION

In this project,memory-based CF which is also called Neighbor-based models implemented and evaluated with the dataset that extracted from scrapping on the internet for patients illness and their therapies.It is easy to be implemented and scaled well with co rated items as mentioned above.Then recommendation predicted using patient and his/her condition.

8. REFERENCES

[1] Kantor P B, Rokach L, Ricci F *et al. Recommender systems handbook[M]. Springer.*. 2011.

[2] *Sarwar et al.,Michael D. Ekstrand, et al. 2011.*

[3] *B. Sarwar et al.,George Karypis 2001*

[4] *Yuan-hong, Wu, and Tan Xiao-qiu. "A real-time recommender system based on hybrid collaborative filtering." Computer Science and Education (ICCSE), 2010 5th International Conference on. IEEE, 2010.*

[5] *Bell, Robert M., Yehuda Koren, and Chris Volinsky. "The BellKor solution to the Netflix prize." KorBell Team's Report to Netflix (2007).*