A REPORT ON

Recommender Systems based on Social Networks

BY :

 Ruddhi Prasad Panda | 2016A7PS0021P

Navjot Bansal | 2016A7PS0070P

Nikhil Dilip Agrawal | 2016A4PS0248P

COURSE : Information Retrieval
INSTRUCTOR : Poonam Goyal
GUIDED BY : Chandramani Chaudhary

1.Problem Statement

To propose a social regularization approach that incorporates social network information to enhance user experience and to provide personalised recommendations for the same.

2. Background of the problem

a. Description of the selected application domain

Recommender systems have attracted much attention in the past decade. A recommender system is a software tool that supports users in identifying the most interesting items. There has been much work done on developing new approaches to recommender systems. The research topic is still popular because of the abundance of practical applications that help users to deal with information overload and their great commercial value. Examples of the applications include recommending books, movies and some other commercial systems.

b. Motivation of the problem

The traditional recommender systems tend to ignore social interactions among users which can improve recommendations.Mainly we tend to consult our friends for the best recommendations, since they are those who we can reach for immediate advice. Hence, in order to provide more accuracy and personalized recommendation results, the social network information should be incorporated. Based on the above viewpoints, a few trust-based recommender systems have been incorporated in social media.

c. Technical issues included

Extreme cases were included the situation where new user has no reviews.
The traditional recommender systems, even social-based approaches, cannot make a recommendation to the new user since the approaches cannot find the similar users for him/her. However, in this paper, when a new user join the system, he/she must pay attention to the existing users in the system, we treat the interactions as the initial friendships of this new user to generate recommendations. We simply use friends' rating distribution on the target item as the result from friend inference.
Even for existing users the UxI matrix generated is extremely sparse.

3. Related Work: Literature survey

3.1. Traditional recommender systems

The major traditional approaches are usually classified into three categories: collaborative filtering (Bellogin et al., 2013, Gunes et al., 2013, Chang and Hsiao, 2011, Bergner et al., 2012), content-based filtering (Illig et al., 2011, Barragáns-Martínez et al., 2010), and hybrid filtering (Adomavicius and Tuzhilin, 2005).

3.2. Trust-based recommender systems

The development of the traditional approaches is very mature, but they are all based on the assumption that users are independent. However, there are actually friendships among users. Based on the above assumption, many researchers have paid attention to trust-based recommender systems which combine the trust social information to further improve traditional approaches.

3.3. Social-based recommender systems

In this paper, the concept "social recommender systems" is defined as combining the social network information which can affect personal behaviors on the web, such as the interactive information among users and the information of tags, to improve recommender systems. With the development of social network in recent years, how to utilize social network information has become a hot issue and been studied in many applications.

4. Research Gap

The methodology suggested by the paper picks up ideas from models ranging from 2003 to 2015. Although it takes in account the fallacies of previous papers, it fails to acknowledge the limits of computational power. The biclustering of the sparse dataset is computationally expensive and pairwise calculation of similarity function using tag weights makes the process slower. Our solution simplifies certain aspects of computation using primitive algorithms which do not reduce the precision by much.

5. System Description
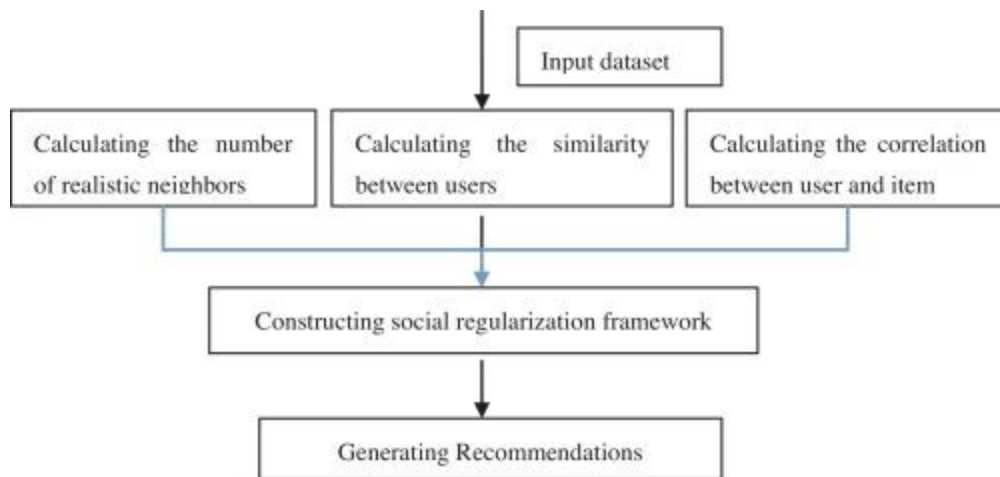
Precision Calculation of the given dataset

```python
user_interest1 = ui(key1,key2)
user_interest2 = ui(key2,key1)
#print(sim)
if(user_interest1 > beta or user_interest2 > beta):
    #this friend is recommended to user (key1)
    S.append((key1,key2))

    tr += 1      #total recommendations
    recom = 1    #current friend is recommended
    #print("tr %s"%(tr))
    #To check if it is a positive contact
    if( ( (key1,key2) in contacts ) or ( (key2,key1) in contacts)):
        tp += 1
        flag = 1

    #To check if recommended bookmarks are serendipitous
    for i in users[key2].bookmarks:
        check = 1
        for j in users[key1].bookmarks:
            sim = book_sim(i,j)
            if(sim>0.5):
                check = 0
                break
```

Satisfied Users and Novelty

```
            #To check if recommended bookmarks are novel
            for b in users[key2].bookmarks:
                if(not b in users[key1].bookmarks):
                    novel += 1
                novel_denom += 1
    if(recom == 1):
        satisfied_denom += 1
    if(flag == 1 and recom == 1):
        satisfied += 1
```

The biclustering algorithm

```
def biclustering(matrix):
    model = SpectralBiclustering()
    model.fit(matrix)
    fit_data = data[np.argsort(model.row_labels_)]
    fit_data = fit_data[:, np.argsort(model.column_labels_)]
    return fit_data
```

Making a matrix of the following bookmarks and tags

```
UxI = np.zeros((U.shape[0],I.shape[0]), dtype='int32')
for x in range(0,data.shape[0]):
    UxI[uid[index[x][0]]][bid[index[x][1]]]=1

f_data=biclustering(UxI)
```

Read data reads the data in form of list of list [User is a data Structure]

```
def read_data():
    for line in open('dataset/user_taggedbookmarks.dat'):
        fields = line.split('\t')
        uid = fields[0]
        bid = fields[1]
        tid = fields[2]

        if not uid in users:
            users[uid] = User(uid)

        if not tid in users[uid].tags:
            users[uid].tags[tid] = 0
        users[uid].count += 1
        users[uid].tags[tid] += 1

        users[uid].bookmarks[bid] = 1
```

6. Evaluation Strategy:

The model was based on the principle of user's Friends and Bookmarks [Pages Liked by the user].The model followed the following methodology

It created a group of bookmarks similar to each other i.e. clustering of bookmarks into categories using the biclustering algorithm into sections which could have certain level of intersection.

Then to calculate the precision and the satisfied users the following procedure was followed

1. It was assumed the user had no friends in the list but a bunch of bookmarks For the reference . Thus using the bookmarks there was a list of friends that were Recommended to the user using which the precision of the code was estimated.
2. Then the section of the code had to determine the satisfaction levels which was the intersection of the accurate bookmarks and the friends suggested was calculated as follows $[f \cap y / f \cup y]$.
   Where f: is the suggested number of {Bookmarks,Friends} to the users;
   y: is the actual number of {Bookmarks,Friends} to the users;
3. Novelty is the third parameter that is calculated using the similar formulae Stated above. Novelty is the expected new bookmarks suggested to a user With given bookmarks and friends.

The model which is based on Libraries $numpy$, $panda$, $sklearn$, $tkinter$, $math$.

Then the Similarity function returned weights for bookmarks and friends using relevant tags and their given weights. The final structure of the returned data was according to the descending weights of the friends and bookmarks to be suggested which is returned on the basis of $\alpha \; and \; \beta$ which is the minimum level of similarity b/w user's preference and the offered bookmarks and recommended friends.
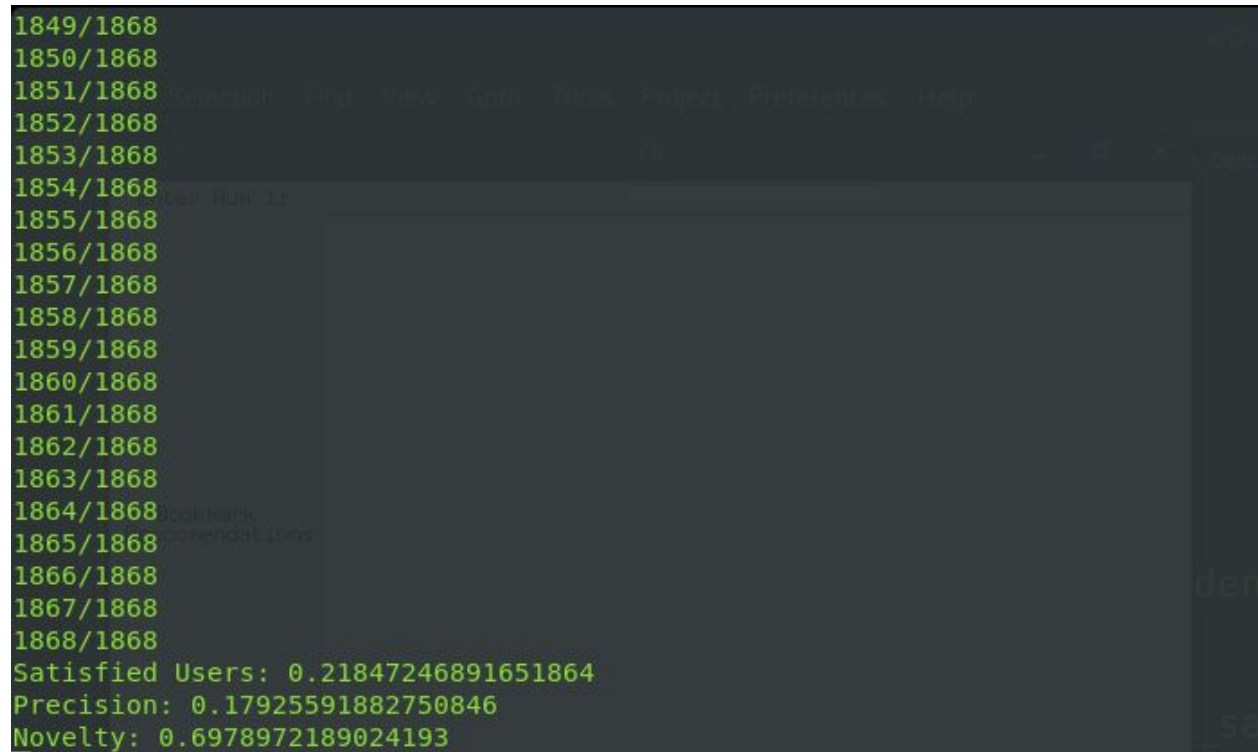
Using the Tkinter package we made the GUI of the program that takes the user id in form of *Integer* number and returns expected *bookmarks* and *friends* in two different sections

7. Experimental Results and evaluation: Present your results and evaluation of your system

For different values of alpha and beta the precision and satisfied users percentages change.
The GUI is fairly simple and has the option of recommending an existing user new bookmarks and friends

Alpha = 0.1, Beta =0.1


```
1849/1868
1850/1868
1851/1868
1852/1868
1853/1868
1854/1868
1855/1868
1856/1868
1857/1868
1858/1868
1859/1868
1860/1868
1861/1868
1862/1868
1863/1868
1864/1868
1865/1868
1866/1868
1867/1868
1868/1868
Satisfied Users: 0.21847246891651864
Precision: 0.17925591882750846
Novelty: 0.6978972189024193
```

Alpha = 0.20 ,Beta=0.20

```
1849/1868
1850/1868
1851/1868
1852/1868
1853/1868
1854/1868
1855/1868
1856/1868
1857/1868
1858/1868
1859/1868
1860/1868
1861/1868
1862/1868
1863/1868
1864/1868
1865/1868
1866/1868
1867/1868
1868/1868
Satisfied Users: 0.15822130299896586
Precision: 0.17184801381692574
Novelty: 0.5916238239799995
```

Alpha = 0.05 ,Beta = 0.05

```
1849/1868
1850/1868
1851/1868
1852/1868
1853/1868
1854/1868
1855/1868
1856/1868
1857/1868
1858/1868
1859/1868
1860/1868
1861/1868
1862/1868
1863/1868
1864/1868
1865/1868
1866/1868
1867/1868
1868/1868
Satisfied Users: 0.2714285714285714
Precision: 0.14993686868686867
Novelty: 0.817236344868305
```

User Recommendations

Enter User ID   8

Reccomendations

13102
80282
80298
40387

Quit   Bookmarks   Friends

Bookmark Recommendations

Enter User ID   8

Bookmark
Reccomendations

www.hdsb.ca
www.media-awareness.ca
www.publicsafety.gc.ca
www.virtualmuseum.ca
allencentre.wikispaces.com
www.dyslexia-surrey.co.uk

Quit   Bookmarks   Friends

8. Conclusion and future work

We employ both friendships among users and rating records (tags) to predict the missing values (tags) in the user-item matrix. The two aspects of social network information are employed in designing social regularization terms. We firstly cluster the dataset to obtain the groups of friends with similar favors to calculate the similarities. The experiments on real large dataset show that our approach outperforms the other popular traditional methods.

As the rapid growth of social network sites continues, the social-based recommender systems become more important. In this paper, we cluster the dataset to obtain the smaller groups with the similar tastes for generating good recommendations. However, we need to investigate the following problems: the cold-start problem, the influence from distance friends who are multiple hops away, the time-series information, the place information and the dynamic variations among users, and how to incorporate the information to improve performance and accord with the practical situation. For example, the recommender systems that take the time information (seasons, weekends) or temporal context into consideration can recommend items with different time effectiveness. These are interesting work to be explored in the future.