Affiliation Recommendation using Auxiliary Networks

Course: Introduction to data mining

Faculty of Science – Department of Mathematics

22.12.2020.

Tin Kranželić

Contents

Goal

- Find and construct meaningful data network
- Carry out and describe 2 models for solving the problem
- Discuss about the implementation of the two models
- Discuss about the metods which concern evaluation of efficiency of various algorithms

Problem description

- N_u - number of users in network
- N_g - number of groups in network
- Friendship network / matrix S ($\in \mathbb{R}^{N_u \times N_u}$)
- Affiliation network / matrix A ($\in \mathbb{R}^{N_u \times N_g}$)
- Generate $N_u \times N_g$ score matrix for ranking of the groups per user

Data

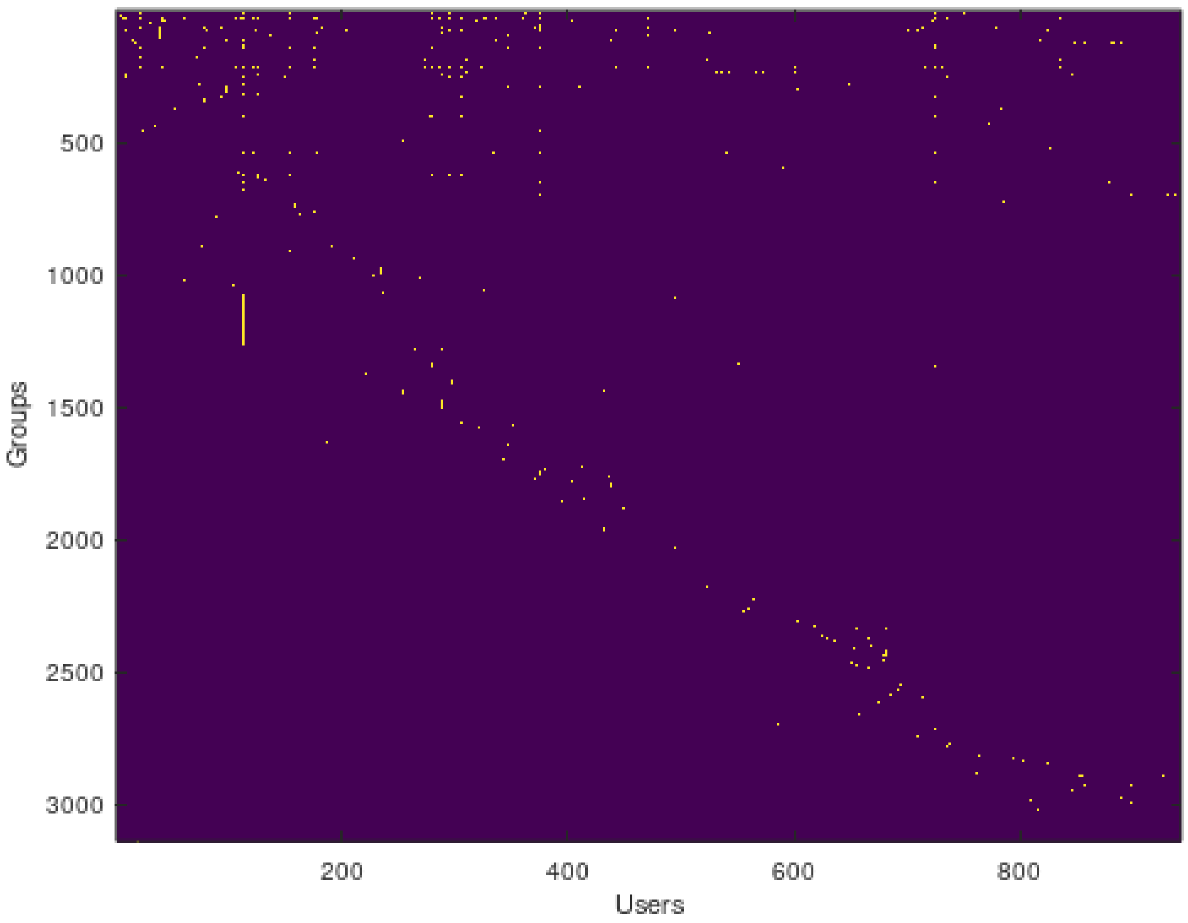
- YouTube dataset used was downloaded from http://socialnetworks.mpi-sws.org/data-imc2007.html
- Number of users: > 1 000 000
- Number of groups: > 30 000

- Using FormData() and IsConnected() functions we extracted a smaller connected component

Descriptive Statistics

Feature	Youtube
N_u	942
N_g	3137
Average number of groups per user	6.8
Minimum number of groups per user	1
Mode of number of groups per user	1
Average number of users per group	2.03
Mode of number of users per group	1
Minimum number of users per group	1
Average number of friends per user	16.2
Mode of number of friends per user	1

Visual representation of the matrix A



- S is symmetric ($S^T = S$), and represents the undirected graph
- We observe matrix $\begin{bmatrix} 0 & A \\ A^T & 0 \end{bmatrix}$ as undirected bipartite graph
- We merge the upper two graphs into one:

$$C = \begin{bmatrix} \lambda S & A \\ A^T & 0 \end{bmatrix},$$

where $\lambda \geq 0$ is a parameter which controls the weight of the matrix S

Models

Graph Proximity model:

- Score matrix will be a matrix whose entries will represent the distance between corresponding vertices in the graph

Latent Factors model:

- Score matrix will be a matrix whose entries will represent the product of corresponding latent factors

Graph Proximity model

- Proximity/distance of the 2 veritces is calculated as a weighted sum of number of paths of different lengths between the 2 veritces
- Katz measure:

(matrix S) $\quad$ $\mathrm{Katz}(S;\ \beta) = \beta S + \beta^2 S^2 + \beta^3 S^3 + \ldots = \sum_{i=1}^{\infty} \beta^i S^i$

$(S^k)_{i,j}$ ... Number of paths of length k between vertices i and j

- $\text{extKatz}(A;\ \beta) = (\beta AA^T + \beta^2(AA^T)^2 + \beta^3(AA^T)^3 + \ldots)A$
  $=\text{Katz}(AA^T;\ \beta)A$

$(AA^T)_{i,j}$ … Number of the shared groups between user i and j

- Score matrix: $\text{Katz}(C;\ \beta)_{12}$
- Score matrix can be calculated in two ways:

1.

$\text{tKatz}(C,\beta,k)_{12} = \sum_{i=1}^{k} \beta^i {C_{12}}^i$

In Octave, tKatz()

2.

$\text{Katz}(C;\ \beta)_{12} = ((I - \beta C)^{-1} - I)_{12}$

In Octave, AltKatz()

Latent Factors model

- For ∀ user $i \in N_u$ and ∀ group $j \in N_g$ we assume that their represenations exist, i.e. low-dimensional vectors u_i and g_j
- Affinity of user i towards group j will correspond to the inder product of their vectors, i.e. $\boxed{{u_i}^T g_j}$

- Subsequently, we have $A \approx U^T G$,
- $U \in \mathbb{R}^{d \times N_u}$ is a matrix of user factors, while $G \in \mathbb{R}^{d \times N_g}$ is a matrix of group facotrs

- We observe the new combined matrix $\mathrm{C}'(\lambda, \mathrm{D}) = \begin{bmatrix} \lambda S & A \\ A^T & D \end{bmatrix}$, where the only novelty is matrix D which represent the similarity between groups

- We have the following aproximation:

$$\begin{bmatrix} \lambda S & A \\ A^T & D \end{bmatrix} \approx \begin{bmatrix} U^T \\ G^T \end{bmatrix} \begin{bmatrix} U & G \end{bmatrix}$$

- Concretely, we want to minimize the following expression:

$$\|U^T U - \lambda S\|^2 + 2\|U^T G - A\|^2 + \|G^T G - D\|^2$$

- Solution to the minimization problem: $SVD(C', d)$

- Remark: D can be of the form $\lambda A^T A$, where $(A^T A)_{i,j}$ simply represents the number of the shared users between groups i and j

- In Octave function LFM()

Performance of the algorithms

- We split the data in 3 groups: $training,\ validation,\ test$ (30 %):

$$training\ \cap validation = \emptyset$$
$$training\ \cap set = \emptyset$$
$$validation\ \cap test = \emptyset$$

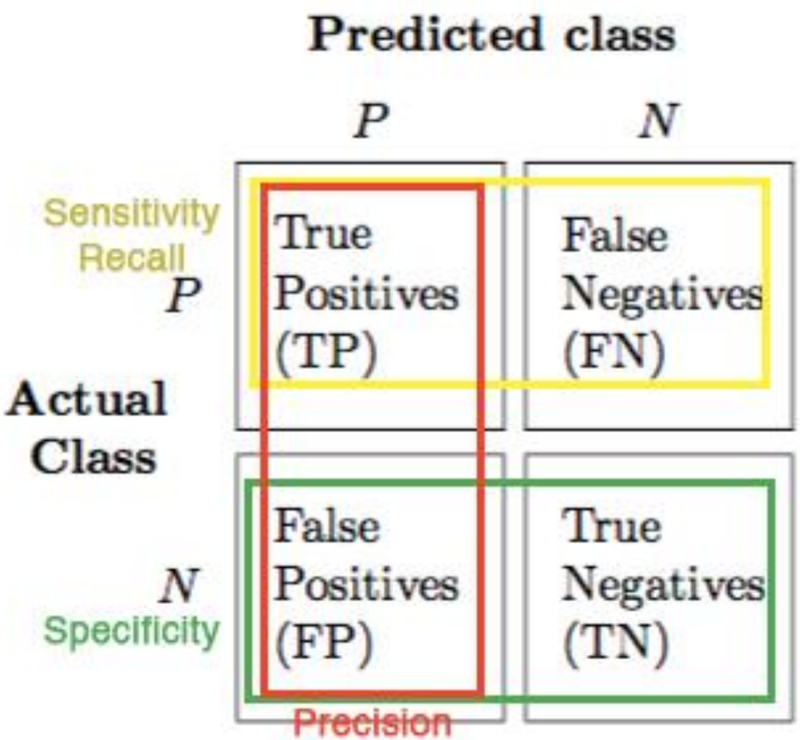
- In Octave function SplitData()

Best parameters

Algorithm	Youtube
Katz(A)	$\beta = 10^{-12}$
Katz(C)	$\beta = 0.01, \lambda = 0.02$
LFM(A)	$d = 20$
LFM(C)	$d = 40, \lambda = 0.7$
LFM(C'(A^TA))	$d = 20, \lambda = 0.8$

- Best parameters found during the validation process for different algorithms
- TOP 15 recommendations per user

Evaluation approach



- $Precision = \frac{TP}{TP+FP}$
- $Sensitivity = \frac{TP}{TP+FN}$
- $Specificity = \frac{TN}{FP+TN}$
- ROC curve - $Sensitivity$ vs $(1 - Specificity)$ plot
- AUC = area under ROC curve
- Greater AUC $\Rightarrow$ more effective algorithm

”per user” vs. ”global”

- ”per user” $Sensitivity$:

$$N_u{}^{-1} \sum_u \frac{k(n_u)}{|test_u|} \,,$$

where $k(n_u)$ denotes the number of good recommendations made in n_u recommendations

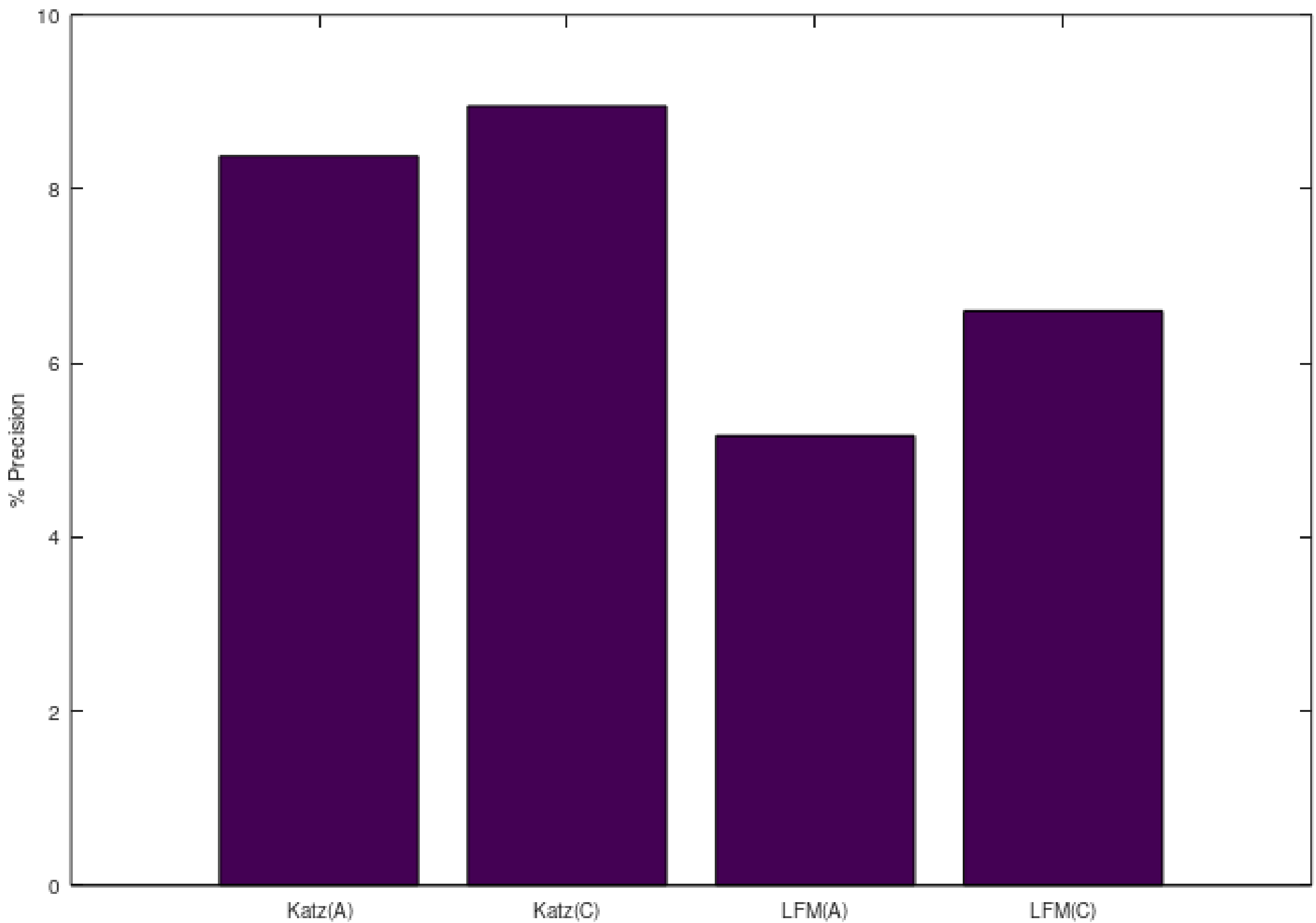
- ”global” $Sensitivity$:

$$\frac{k'(n)}{\sum_u |test_u|}$$

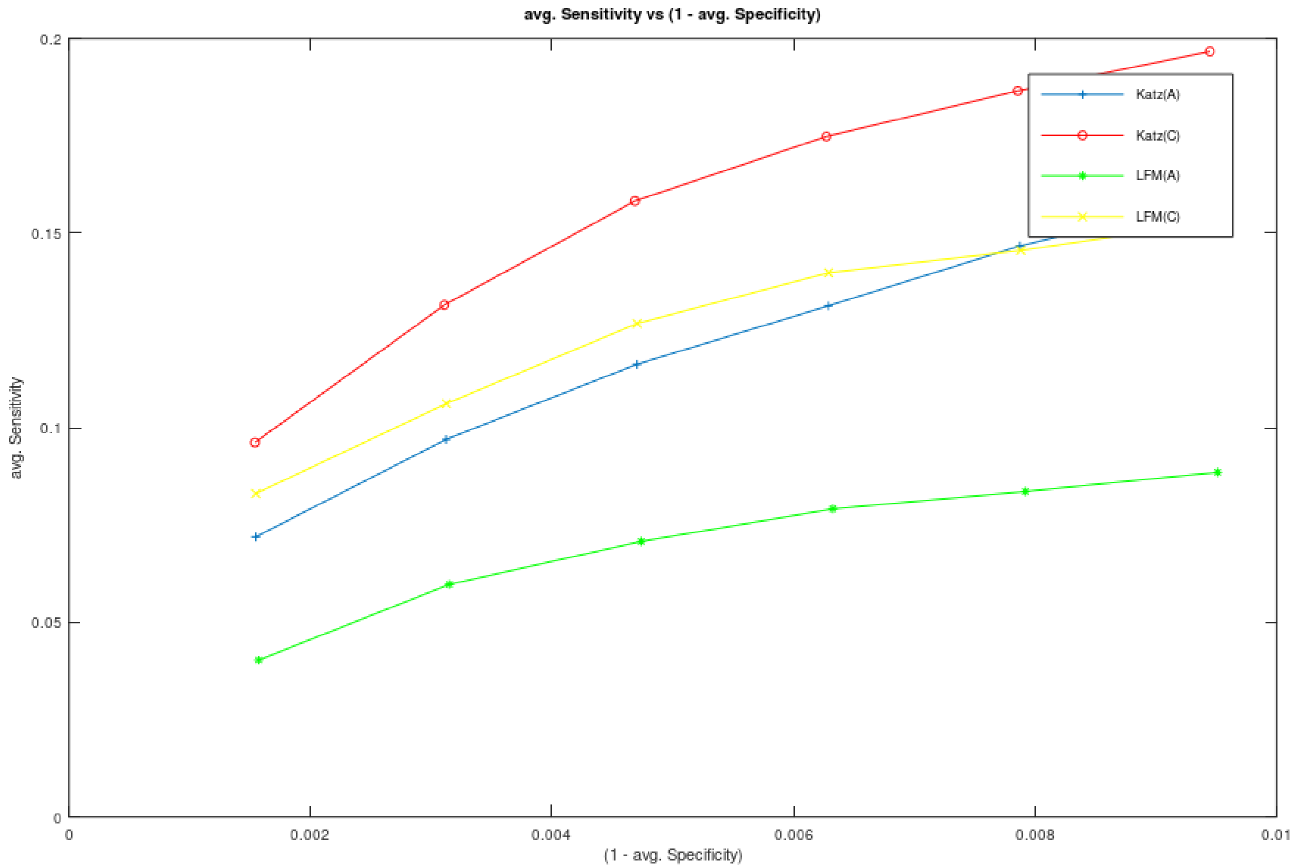
where $k'(n)$ denotes the number of good recommendations made in overall n recommendations

- In Octave functions Prec() i AvgSenSpec()

”global” Sensitivity

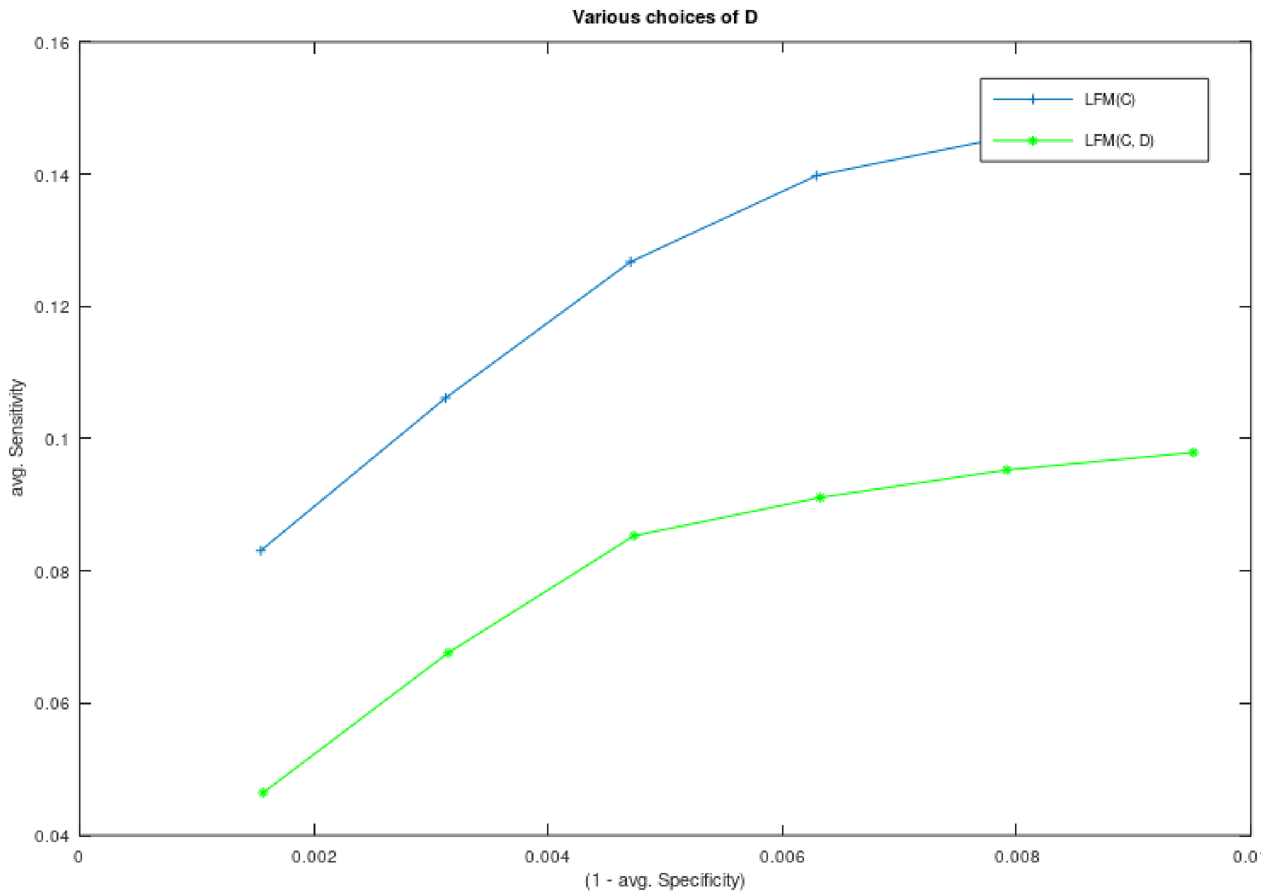


ROC curves – ”per user” Sensitivity



$$n_u = \{5, 10, 15, 20\ , 25, 30\}$$

Different choice of D in $C'(D, \lambda)$



$$D = A^T A$$

Conclusion

- We saw how algorithms fruitfully exploited the information acquired from the matrix S

- GPM > LFM

List of functions

- sem1.m – main program
- FormData() – function which returns binary matrices S and A and indices(ID) of users and groups which are present in those matrices
- SplitData() – function which splits the matrix A in 3 parts and returns disjoint matrices/datasets: $training, validation, test$
- AltKatz() – returns score matrix using the Graph Proximity model
- LFM() – returns score matrix using the Latent Factors model
- AvgSenSpec() – calculates and returns ”per user” $Sensitivity$ and $(1 - Specificity)$
- Prec() – returns ”global” $Precision$
- Val() – returns the number of good recommendations
- GroupRecommend() – returns the matrix of sorted indices (descend, by magnitude of affinity) of groups per user
- IsConnected() – returns 1 if matrix(graph) is connected, otherwise 0

References

- [1] Vishvas Vasuki, Nagarajan Natarajan, Zhengdong Lu, Berkant Savas, Inderjit Dhillon: Affiliation Recommendation using Auxiliary Networks, Department of Computer Science and ICES, University of Texas at Austin
- [2] Vishvas Vasuki, Nagarajan Natarajan, Zhengdong Lu, Berkant Savas, Inderjit Dhillon: Scalable Affiliation Recommendation using Auxiliary Networks, Department of Computer Science and ICES, University of Texas at Austin
- [3] Zlatko Drmač: lectures from the Course Introduction to data mining, Zagreb 2020.