gRecs: A collaborative filtering framework for group recommendations

Eirini Ntoutsi[1], Kostas Stefanidis[2], Kjetil Norvag[2], Hans-Peter Kriegel[1]

1 Institute for Informatics, Ludwig-Maximilians-Universität (LMU) München, Germany

2 Department of Computer and Information Science, Norwegian Uni. of Science and Technology (NTNU) ,Trondheim, Norway

The 36th Annual Conference of the German Classification Society on Data Analysis, Machine Learning and Knowledge Discovery, 1-3 August 2012, Hildesheim

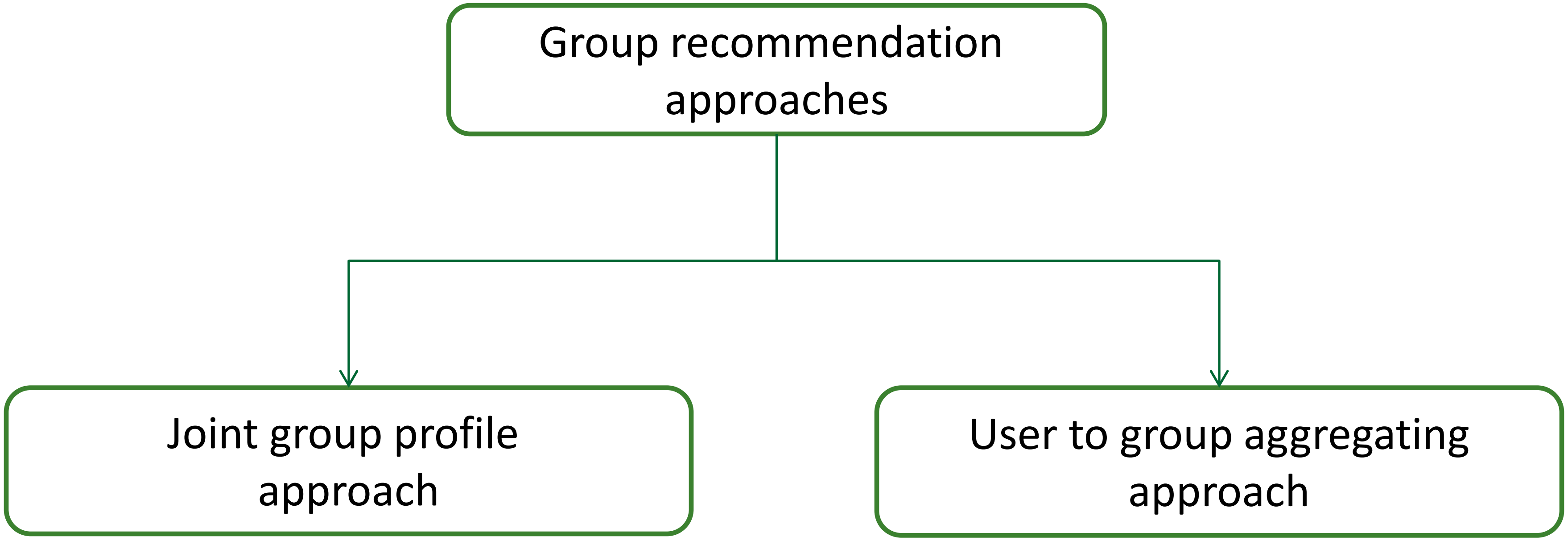
Outline

Recommendation systems

- Provide users with suggestions about products, movies, restaurants, …
- Very popular nowadays, e.g., Amazon, NetFlix, MovieLens

- The majority of recommendation systems are designed for personal recommendations.
- However, there are cases where the items to be selected are not intended for personal usage but for a group of people (group recommendations)
  - e.g., friends planning to watch a movie
  - e.g., a family selecting a holiday destination
  - e.g., colleagues planning to dine together

Related work on group recommendations


Group recommendation approaches
Joint group profile approach
User to group aggregating approach

Create a joint profile for the group and suggest items w.r.t. this group profile.

- e.g., Yu et al. 2006

Aggregate the single user recommendations into group recommendations.

- e.g., Con et al, 2001

gRecs: Group Recommendations

- We propose a framework for group recommendations following the collaborative filtering approach.
  - The most prominent items for each user of the group are identified based on items that similar users liked in the past.
  - Single user recommendations are aggregated into group recommendations based on different aggregation designs
- We leverage the power of a top-k algorithm for efficient aggregation.
- The main bottleneck in collaborative filtering is locating the most similar users for a given user.
  - Naïve approach: locate the most similar users to a given one by searching the whole database of users.
  - Clustering approach: model the user-item interactions in terms of clustering and use the extracted clusters for predictions.

Recommendation model

- $I = \{i_1, i_2, \ldots, i_d\}$, set of items e.g. movies, books, restaurants
- $U = \{u_1, u_2, \ldots, u_n\}$, set of users
- *preference(u, i)* ∈ [0,1]: the preference/rating of user *u* ∈ *U* for item *i* ∈ *I*.

- But, typically users rate only a few items (and |I| is to high!)
- For the unrated items *i'*, we estimate their relevance for a user
  - *relevance(u,i')*: estimated relevance score of *u* for *i'*, if *preference(u, i')* = Ø
  - Different ways to estimate relevance:
    - Content-based
    - Collaborative filtering
    - Hybrid

Personal recommendations

- How can we estimate *relevance(u,i)*?
- Collaborative filtering idea: use preferences of other users that exhibit *the most similar behavior* to the given user in order to produce relevance scores for unrated items of the given user.

- Similarity is estimated in terms of some similarity/distance function
- F_u: the set of similar users to u, also called friends

Definition 1 (Friends). *Let $\mathcal{U}$ be a set of users. The friends $\mathcal{F}_u$, $\mathcal{F}_u \subseteq \mathcal{U}$, of a user $u \in \mathcal{U}$ is a set of users, such that, $\forall u' \in \mathcal{F}_u$, $simU(u, u') \geq \delta$ and $\forall u'' \in \mathcal{U} \backslash \mathcal{F}_u$, $simU(u, u'') < \delta$, where δ is a threshold similarity value.*

Personal recommendations scores

- Relevance computation based on the set of friends

$$relevance(u,i) = \frac{\sum_{u' \in (\mathcal{F}_u \cap \mathcal{P}_i)} simU(u,u') preference(u',i)}{\sum_{u' \in (\mathcal{F}_u \cap \mathcal{P}_i)} simU(u,u')}$$

P_i: users u' ∈ U with preference(u',i)

- But, how confident are the relevance scores associated with the recommended items, considering the sparsity of the matrix *U* X *I*?

$$support(u,i) = |\mathcal{F}_u \cap \mathcal{P}_i| / |\mathcal{F}_u|$$

- To estimate the worthiness of an item recommendation we combine relevance and support scores:

Definition 2 (Personal Value). *Let $\mathcal{U}$ be a set of users and $\mathcal{I}$ be a set of items. Let $w_1, w_2 \geq 0 : w_1 + w_2 = 1$. The personal value of an item $i \in \mathcal{I}$ for a user $u \in \mathcal{U}$ with friends $\mathcal{F}_u$, such that, $\nexists preference(u,i)$, is:*

$$value_{\mathcal{F}_u}(u,i) = w_1 \times relevance(u,i) + w_2 \times support(u,i)$$

Group recommendations

- What is the relevance score of a group of users $G=\{u_1, u_2, \ldots, u_k\} \subseteq U$ for an item $i \in I$?
- Idea: aggregate the personal recommendation scores of the group members into overall group recommendations

Definition 3 (Group Value). *Let $\mathcal{U}$ be a set of users and $\mathcal{I}$ be a set of items. Given a group of users $\mathcal{G}$, $\mathcal{G} \subseteq \mathcal{U}$, the group value of an item $i \in \mathcal{I}$ for $\mathcal{G}$, such that, $\forall u \in \mathcal{G}$, $\nexists preference(u, i)$, is:*

$$value(\mathcal{G}, i) = Aggr_{u \in \mathcal{G}}(value_{\mathcal{F}_u}(u, i))$$

Users to group aggregation

- 3 different aggregation designs
  - Least misery design: Strong member preferences act as veto
    - e.g., do not recommend steakhouses if there is a vegetarian in the group

$$value(\mathcal{G}, i) = \min_{u \in \mathcal{G}}(value_{\mathcal{F}_u}(u, i))$$

  - Most optimistic design: the most satisfied member is the influential
    - e.g., recommend a movie to the group if a member is highly interested in it and the others are reasonable satisfaction

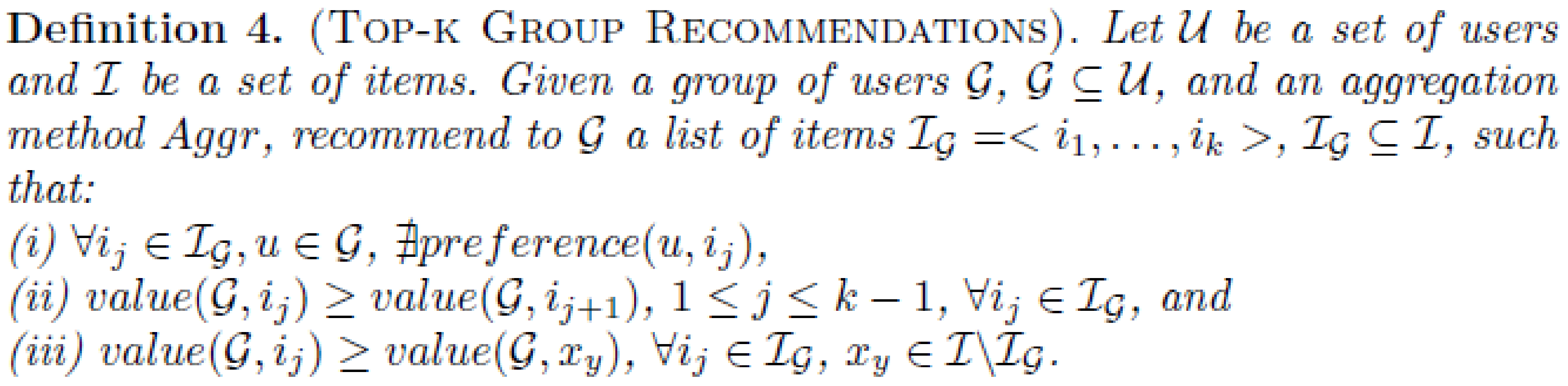
$$value(\mathcal{G}, i) = \max_{u \in \mathcal{G}}(value_{\mathcal{F}_u}(u, i))$$

  - Fair design: democracy wins
    - e.g., recommend a holiday destination if on average the group is satisfied

$$value(\mathcal{G}, i) = \left(\sum_{u \in \mathcal{G}} value_{\mathcal{F}_u}(u, i)\right) / |\mathcal{G}|$$

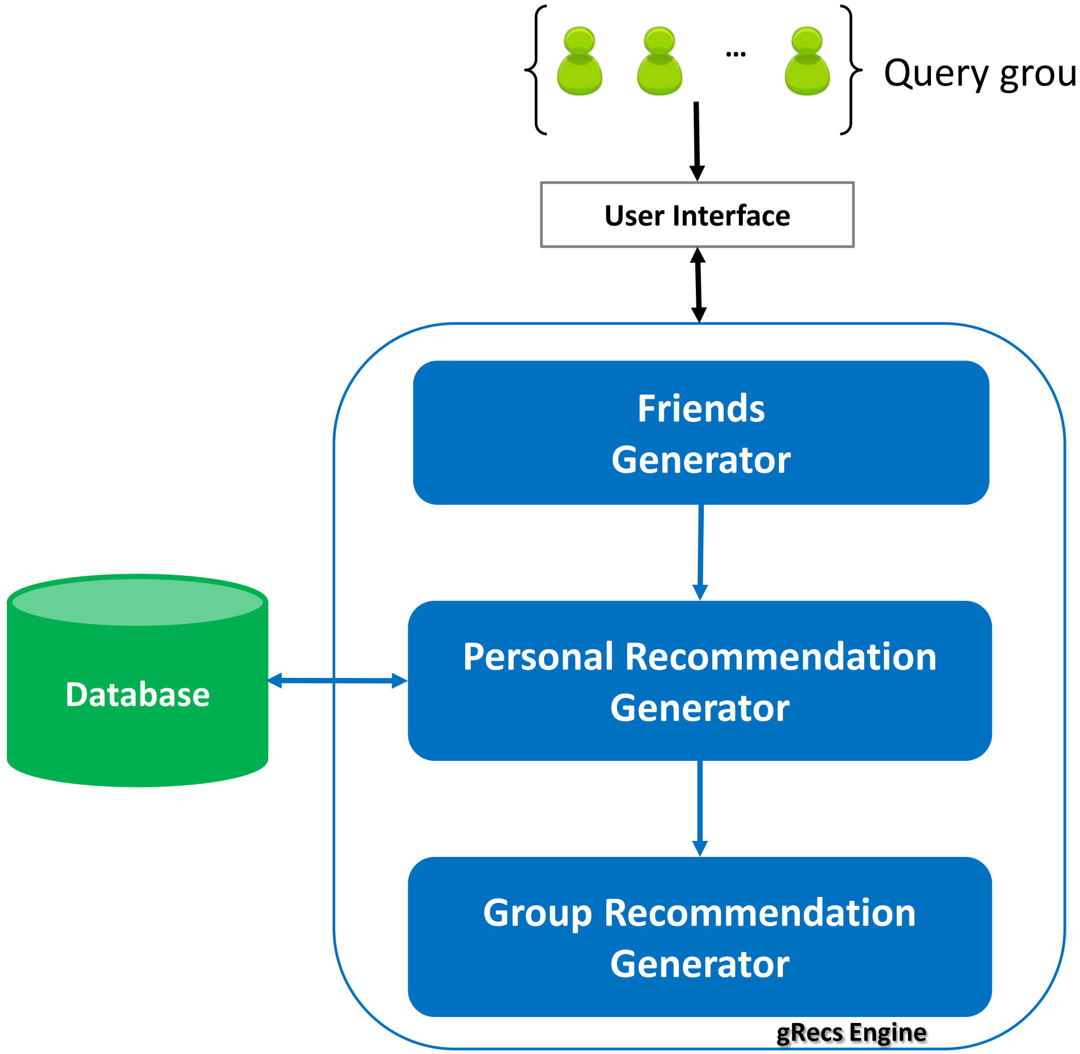
Top-k group recommendations

- Given a group of users and a restriction *k* on the number of the recommended items, we would like to provide *k* suggestions for items that are highly relevant to the preferences of all the group members and, also, exhibit high support.



Definition 4. (Top-k Group Recommendations). *Let $\mathcal{U}$ be a set of users and $\mathcal{I}$ be a set of items. Given a group of users $\mathcal{G}$, $\mathcal{G} \subseteq \mathcal{U}$, and an aggregation method Aggr, recommend to $\mathcal{G}$ a list of items $\mathcal{I}_{\mathcal{G}} =< i_1, \dots, i_k >$, $\mathcal{I}_{\mathcal{G}} \subseteq \mathcal{I}$, such that:*
(i) $\forall i_j \in \mathcal{I}_{\mathcal{G}}, u \in \mathcal{G}$, $\nexists preference(u, i_j)$,
(ii) $value(\mathcal{G}, i_j) \geq value(\mathcal{G}, i_{j+1})$, $1 \leq j \leq k-1$, $\forall i_j \in \mathcal{I}_{\mathcal{G}}$, and
(iii) $value(\mathcal{G}, i_j) \geq value(\mathcal{G}, x_y)$, $\forall i_j \in \mathcal{I}_{\mathcal{G}}$, $x_y \in \mathcal{I} \backslash \mathcal{I}_{\mathcal{G}}$.

Group recommendations computation



...
User Interface
Friends Generator
Database
Personal Recommendation Generator
Group Recommendation Generator
gRecs Engine

Query group *G*, aggregation design *Aggr()*, *k*

(1) Locate the set of users F_u, for all users *u∈G.*

(2) Compute the personal value scores for all users *u∈G.*

(3) Combine the independent scores w.r.t. *Aggr()* and derive the top-*k* group value scores for *G.*

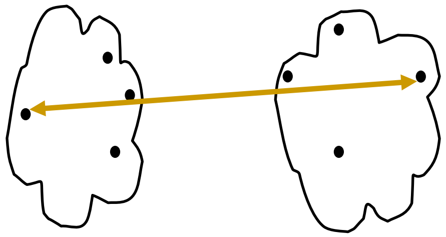
(1) Friends generator

- Baseline approach:
  - For each user $u \in G$, the friend set F_u consists of all its similar users in U, i.e., $F_u=\{u' \in U: simU(u,u') \geq \delta\}$
    - No pre-computations required
    - Inefficient in large systems
- User clustering approach:
  - Organize users into clusters of similar users.
  - For a user $u \in G$, the friend set F_u consists of the members of its corresponding cluster C, i.e., $F_u=\{u' \in C\}$.
    - Pre-computed groups
    - Faster computations

User clustering approach

- We employ an agglomerative hierarchical clustering algorithm
- Initially, each user is placed in its own cluster
- At each step the two most similar clusters are merged
  - Complete link distance:

    similarity between two clusters is the min similarity between any two users in the clusters



- Stop, if the similarity of the closest pair of clusters violates the user similarity threshold *δ*.

- Property: For each pair of users *u, u' Є C, simU(u,u')≥δ*
  - No false positives, true negatives possible

(2) Personal recommendation generator

- For each user *u Є G*, and for his/her unrated items i‘, use collaborative filtering to compute *value(u,i’).*
- 2 possible implementations depending on the friends generator step (1):
  - $value_{F_u}(u,i')$ if baseline approach if adopted
  - $value_{C_u}(u,i')$ if user clustering approach if adopted
- (item, value) pairs are generated for each user *u Є G* → V_u

(3) Top-*k* group recommendation generator

- Naïve approach:
  - aggregate V_u for all u∈ G and compute the group value scores for all i ∈ *I*
  - rank the scores and report the top-*k* valued items
- A faster approach: TA algorithm [FagLotNao01]
  - Use the ranked sets V_u ; 2 types of item access: sorted access, random access
  - Do sorted access to each V_{uj}. For each item seen, do random accesses to the other ranked sets to retrieve the missing item personal value scores.
  - Compute the group value score of each item that has been seen. Rank the items based on their group value scores and select the top-k ones.
  - Stop to do sorted accesses when the group value scores of the k items are at least equal to a threshold value that is defined as the aggregation score of the scores of the last items seen in each ranked set.

Explanations

- The success of recommendations relies on explaining the cause behind them [TinMas11].
  - Except for the top-*k* suggested items, we provide an explanation of why the specific item appears in the top-*k* list.
- Explanation template:

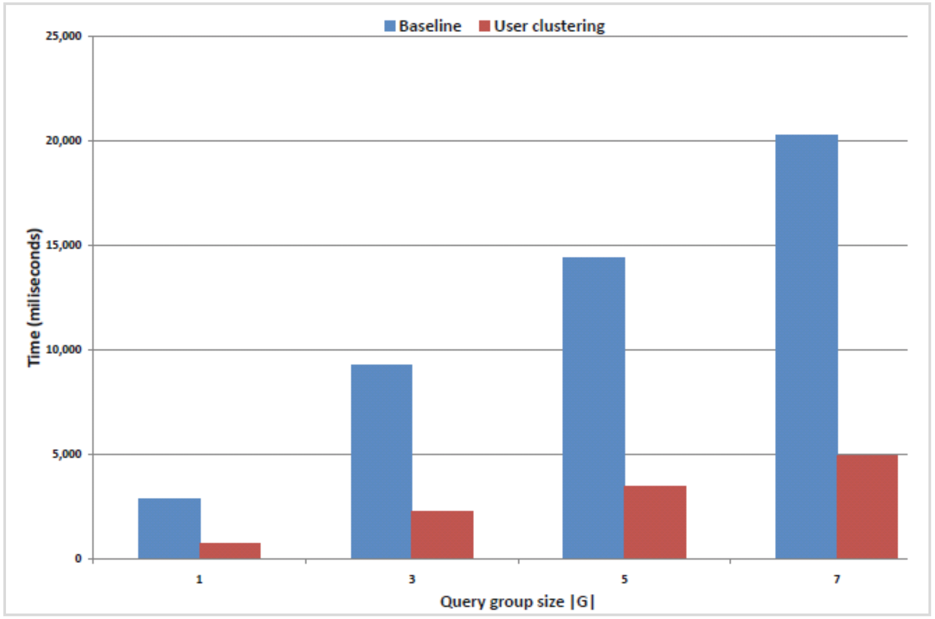
```
ITEM i HAS GROUP VALUE SCORE value(G,i) BECAUSE OF
USER(S) {u1, …, uy}.
```

  - e.g., “Movie Dracula has group value score 0.9 because of user Jeffrey”.
- Explanations depend on the aggregation design:
  - Least misery: for each suggested item, the person with the min personal value score is reported.
  - Most optimistic design: for each suggested item, the person with the max personal value score is reported.
  - Fair design: for each suggested item, the members of the group close to the average value are reported.

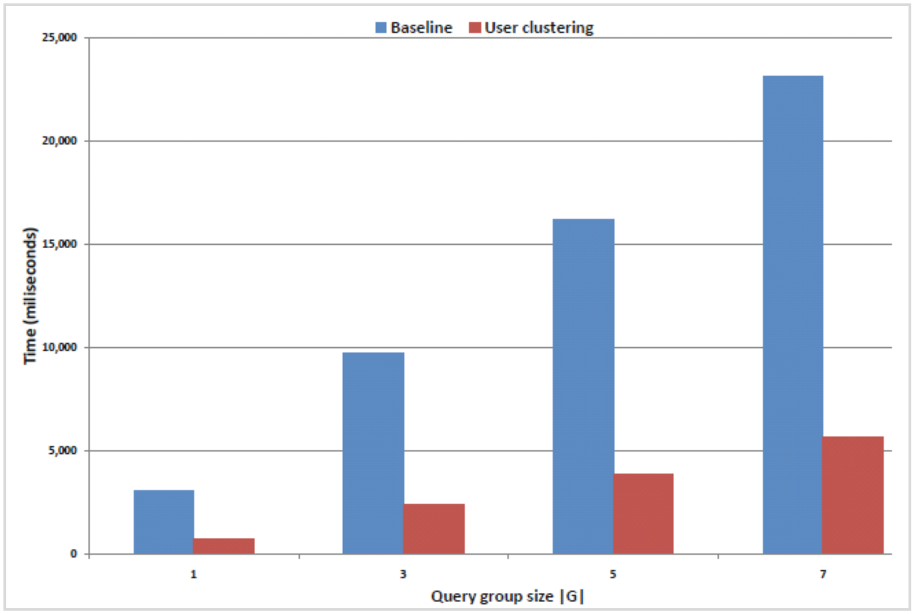
Experiments

- Goal: evaluate user clustering approach vs baseline approach
- MovieLens dataset (1,000 users; 1,700 items; 100,000 ratings)
- Evaluation criteria
  - Quality of recommendations
    - *commonRecs* : # common suggested items by both approaches.
    - *rankRecsDist* : distance between two partial rankings based on # of pairwise disagreements between them [FagKumSiv03].
  - Efficiency of recommendations
    - Execution time for computing personal recs of the query group members.
      - We omit the aggregation time for computing top-k, since this is the same for both cases.
- To set up a query group, we randomly select the members of the group from the user base *U*.
- We run each experiment 100 times and report avg values.

Execution time



(a) δ=0.15



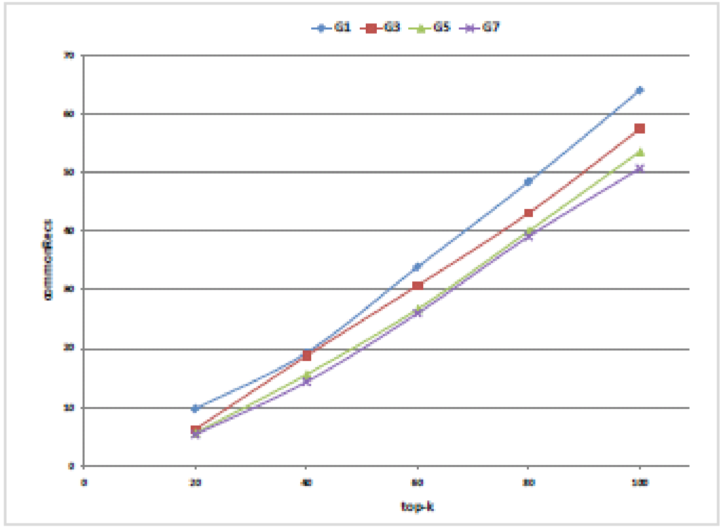
(b) δ=0.30

Time complexity for fair design with w_1=0.5, w_2=0.5

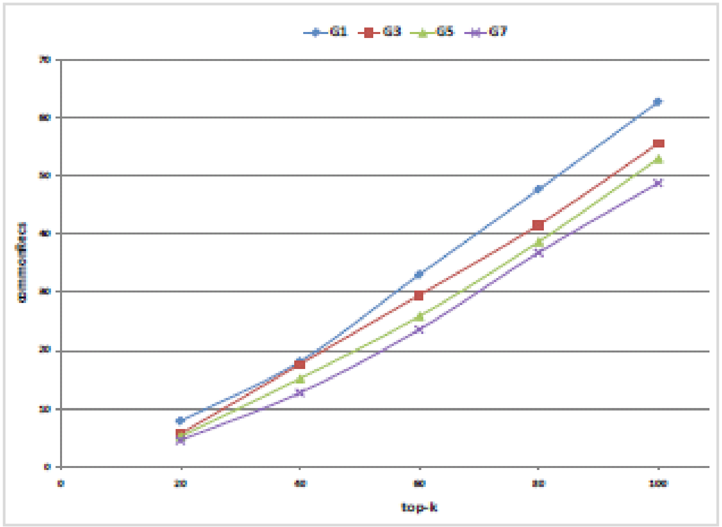
- User clustering requires almost 25% of the time required by baseline approach
- For larger |G|, reduction becomes more evident

Quality of recommendations: Fair design ($w_1=w_2=0.5$)

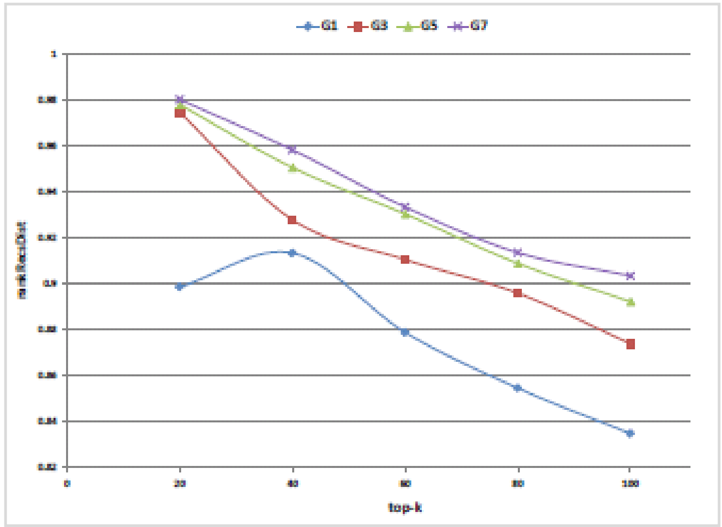
δ=0.15



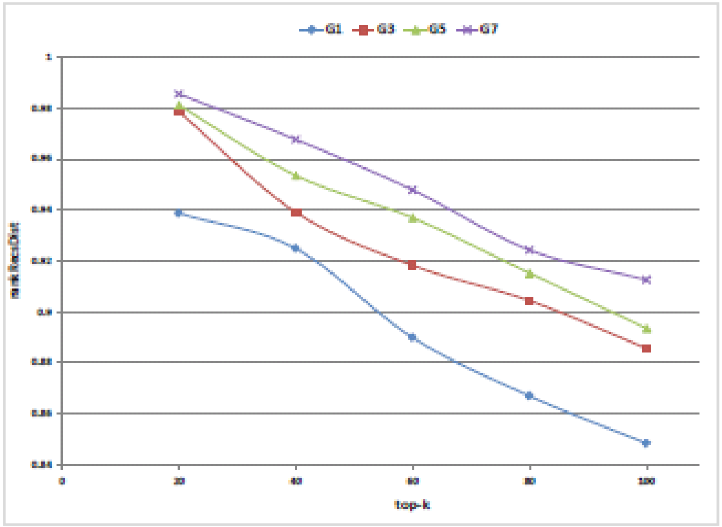
commonRecs



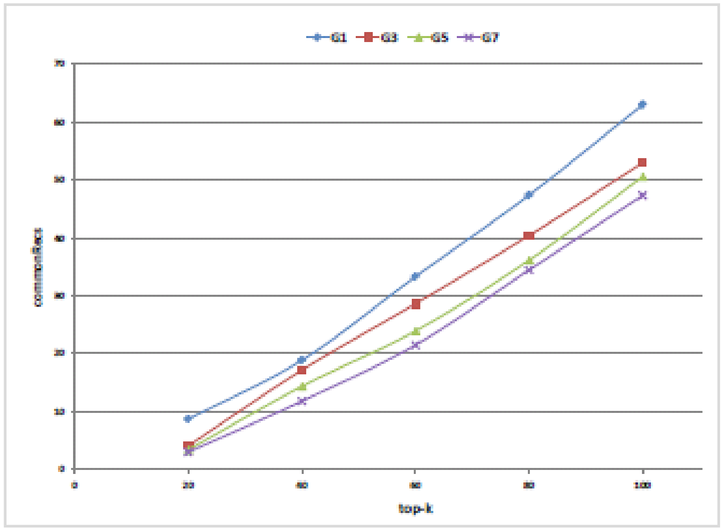
δ=0.30

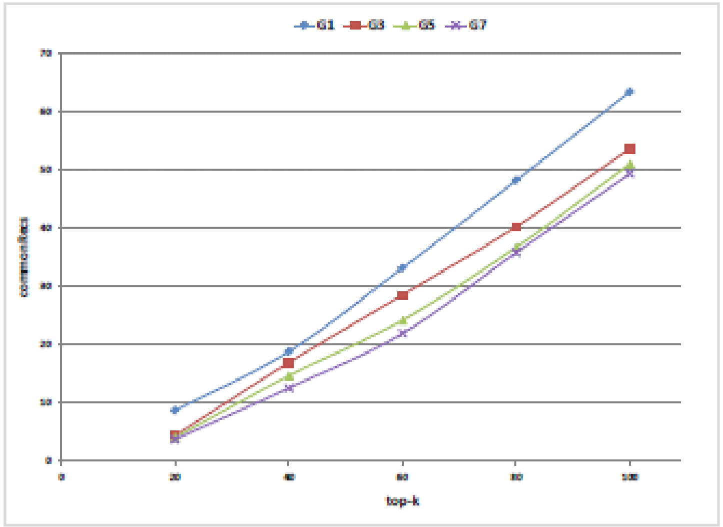


rankRecsDist



Quality of recommendations: Most optimistic design ($w_1=w_2=0.5$)

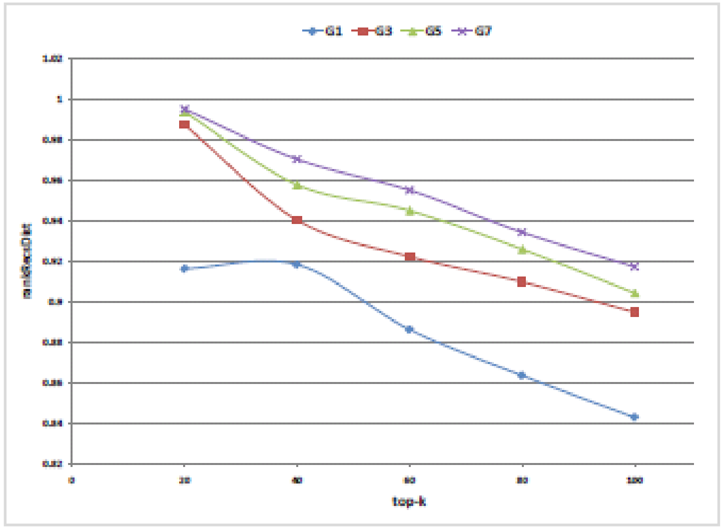


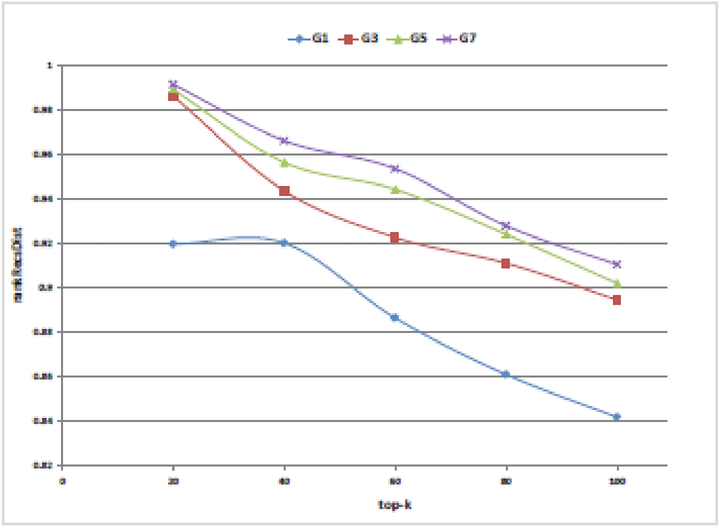


commonRecs

$\delta=0.15$

$\delta=0.30$

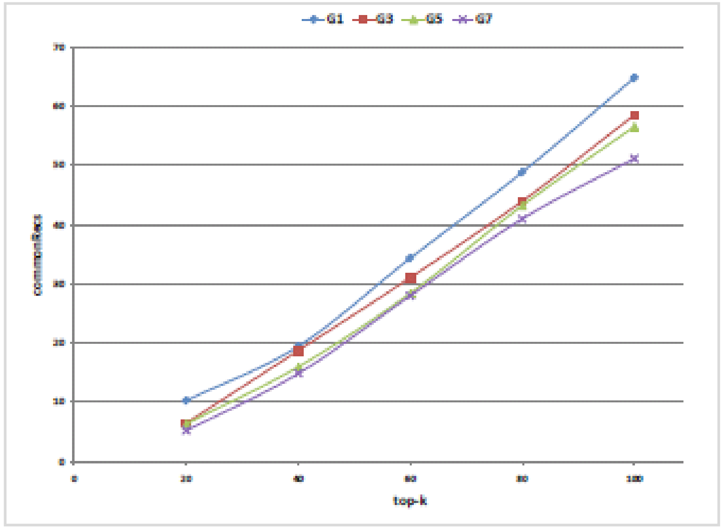




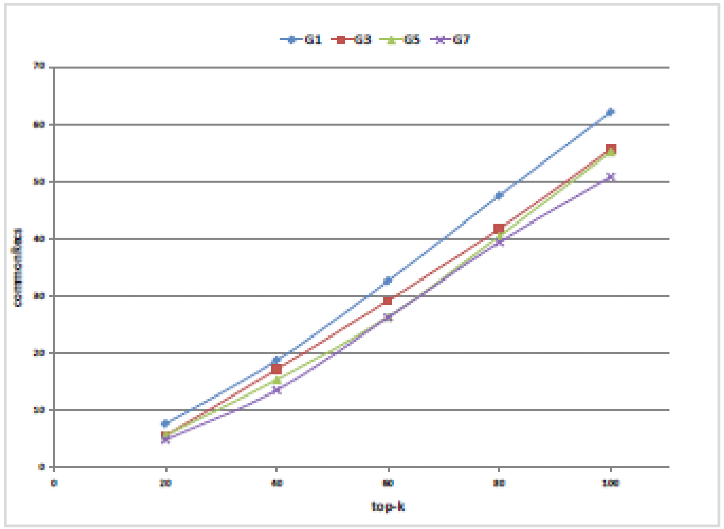
rankRecsDist

Quality of recommendations: Least misery design (w_1=w_2=0.5)

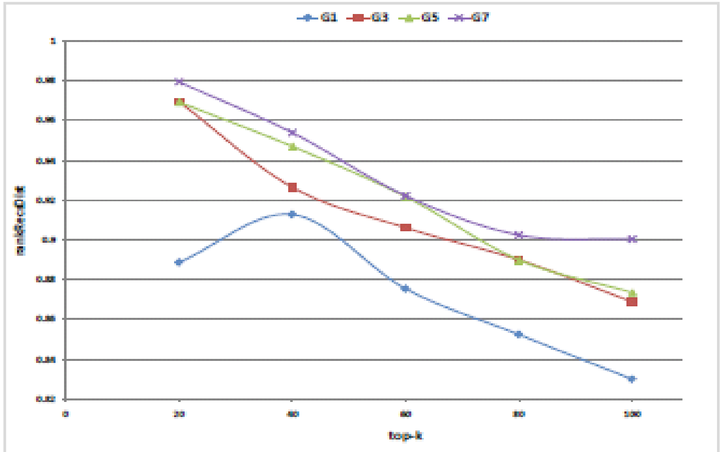
δ=0.15



commonRecs

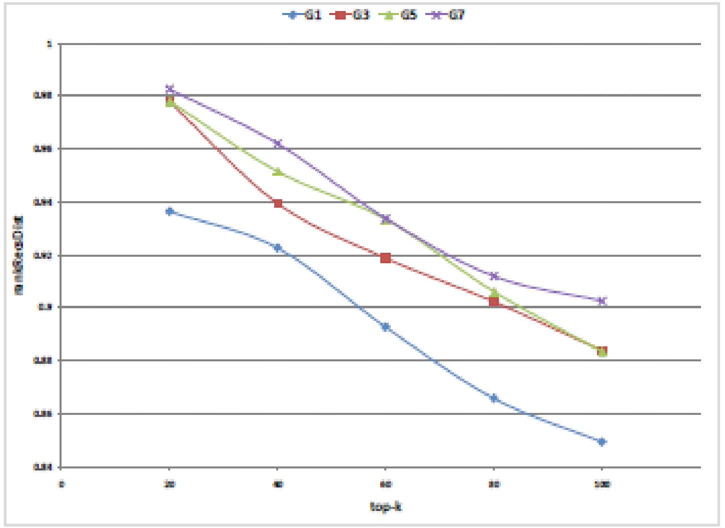


δ=0.30



(e) Least misery design

rankRecsDist

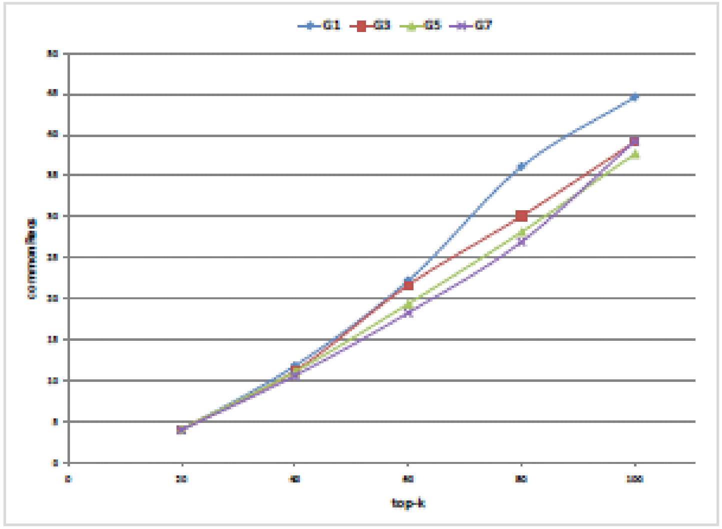


Quality of recommendations overview

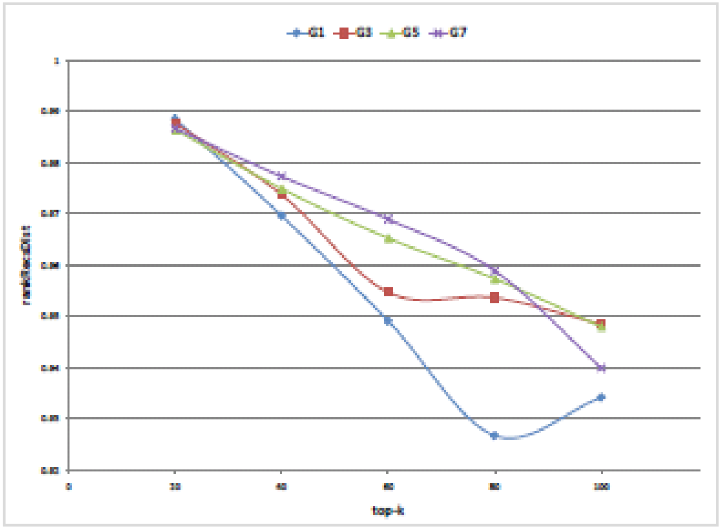
- As |*G*| increases, *commonRecs* score decreases, since the group recommendations rely in a more diverse set of users and personal values.
- Accordingly, *rankRecsDist* increases as |*G*| increases

- The fair and the least misery designs achieve better results when compared to the most optimistic design.
  - Since the members of the query group are selected randomly, this is expected, since it is more difficult to find agreements for max personal values.
- When *δ increases, the commonRecs decreases for the fair and least misery* designs and slightly increases for the most optimistic design.
  - Corresponding findings also hold for *rankRecsDist.*

Effect of weighting factors w_1, w_2



(a) *commonRecs*



(b) *rankRecsDist*

Fair design for δ=0.30 with w_1=1, w_2=0

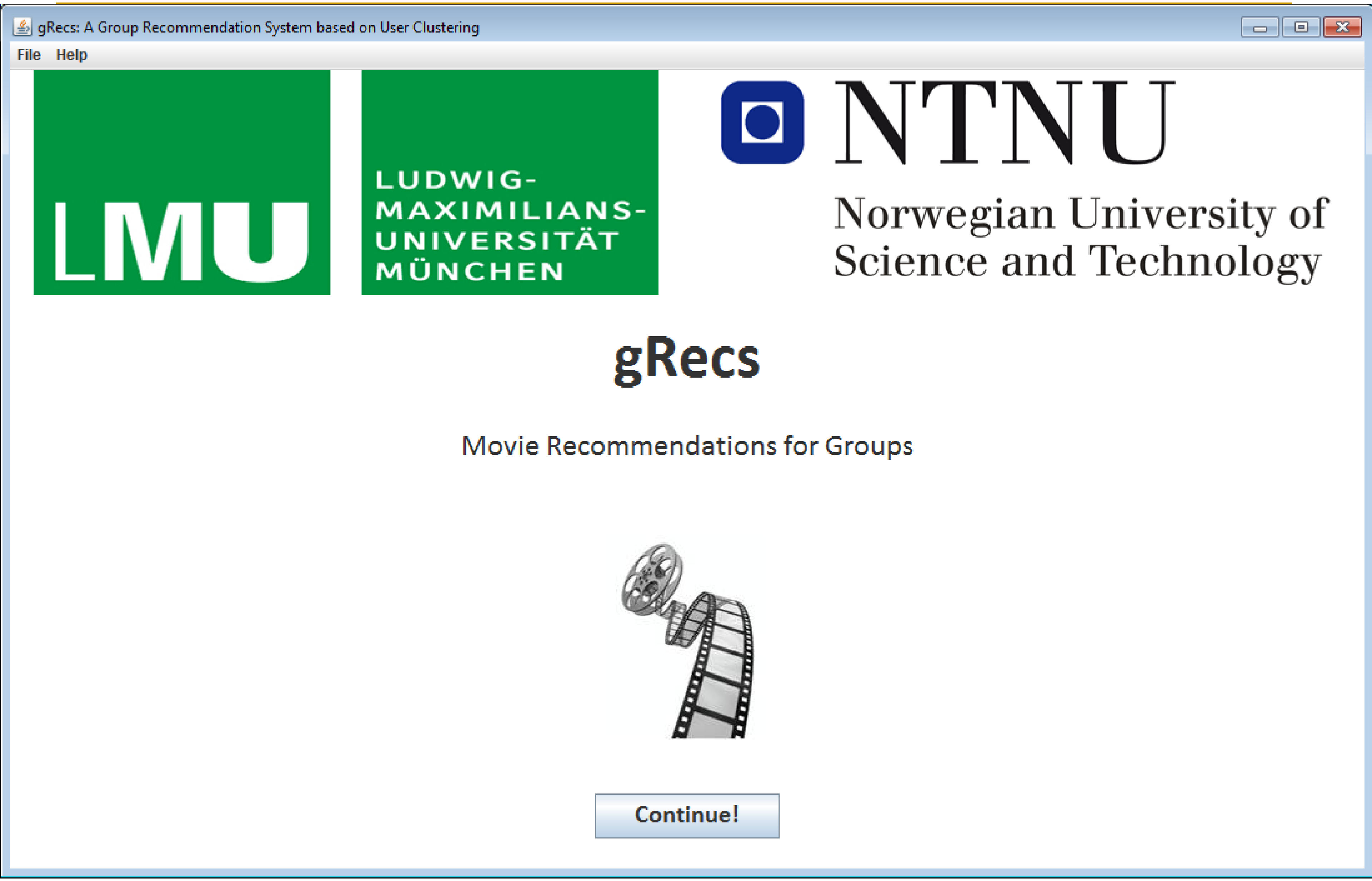
- *commonRecs and rankRecsDist behave worst comparing* to the equal importance case (w_1=w_2=0.5)
- It seems that support improves the quality of recommendations.

Conclusions

- We presented gRecs, a collaborative filtering framework for group recommendations
  - We do not exhaustively search for similar users in the whole user base, but we pre-partition users into clusters of similar ones and use the cluster members for recommendations.
  - We efficiently aggregate the single user recommendations into group recommendations by leveraging the power of a top-k algorithm
- Our results show that employing user clustering considerably improves the execution time, while preserves a satisfactory quality of recommendations.

Outlook

- Further experimentation with the parameters, other aggregation designs and other datasets.
  - e.g., assign higher weights to individuals or subgroups
  - e.g., datasets where the notion of friends is already specified like social networks
- Different clustering algorithms
  - e.g., partitioning based methods
- To deal with the high dimensionality and sparsity of ratings, we envision subspace clustering to find clusters of similar users and subsets of items where these users have similar ratings for the items.
  - Most subspace clustering algorithms ignore missing values
- Dealing with evolving user preferences



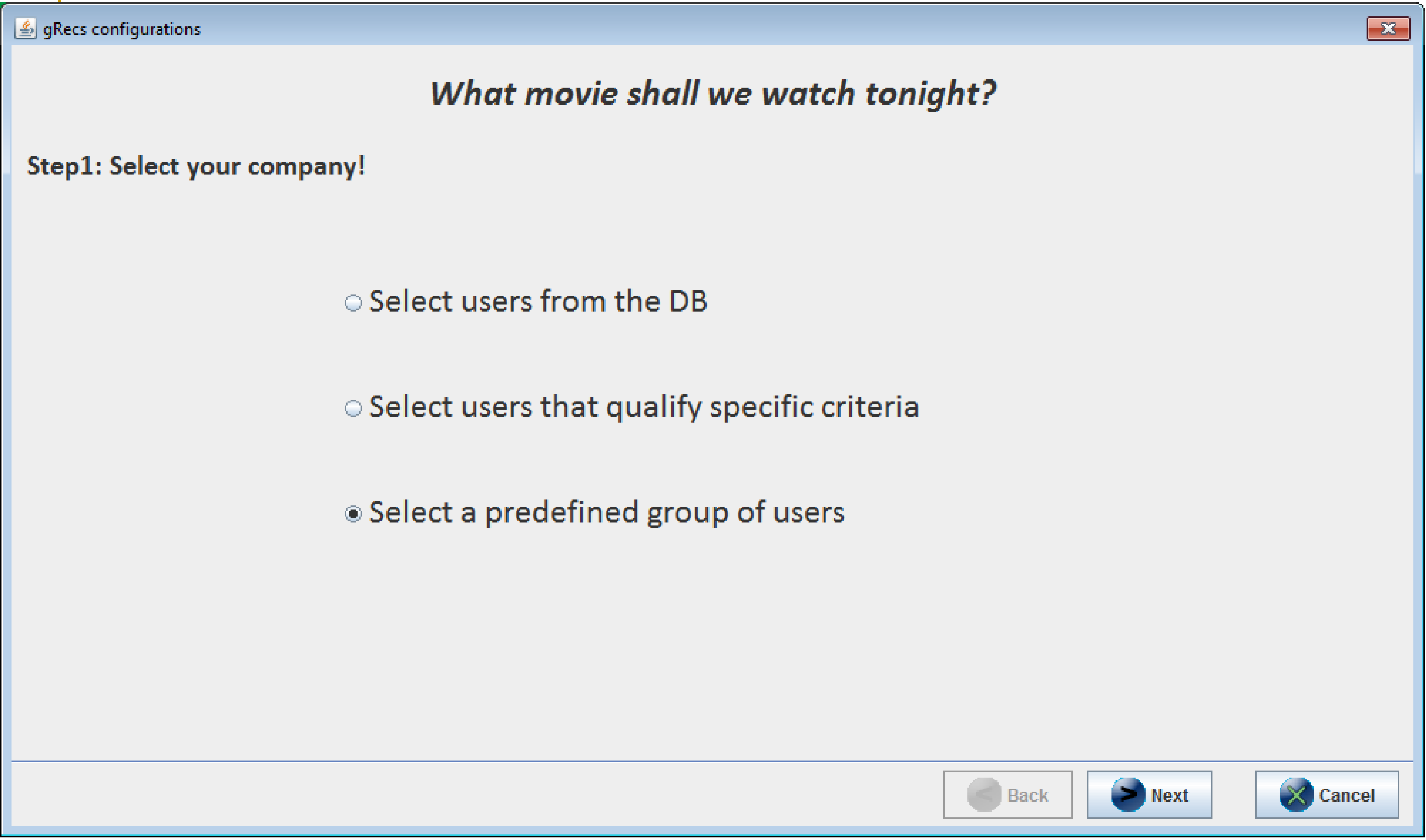
gRecs: A Group Recommendation System based on User Clustering

File Help

gRecs

Movie Recommendations for Groups

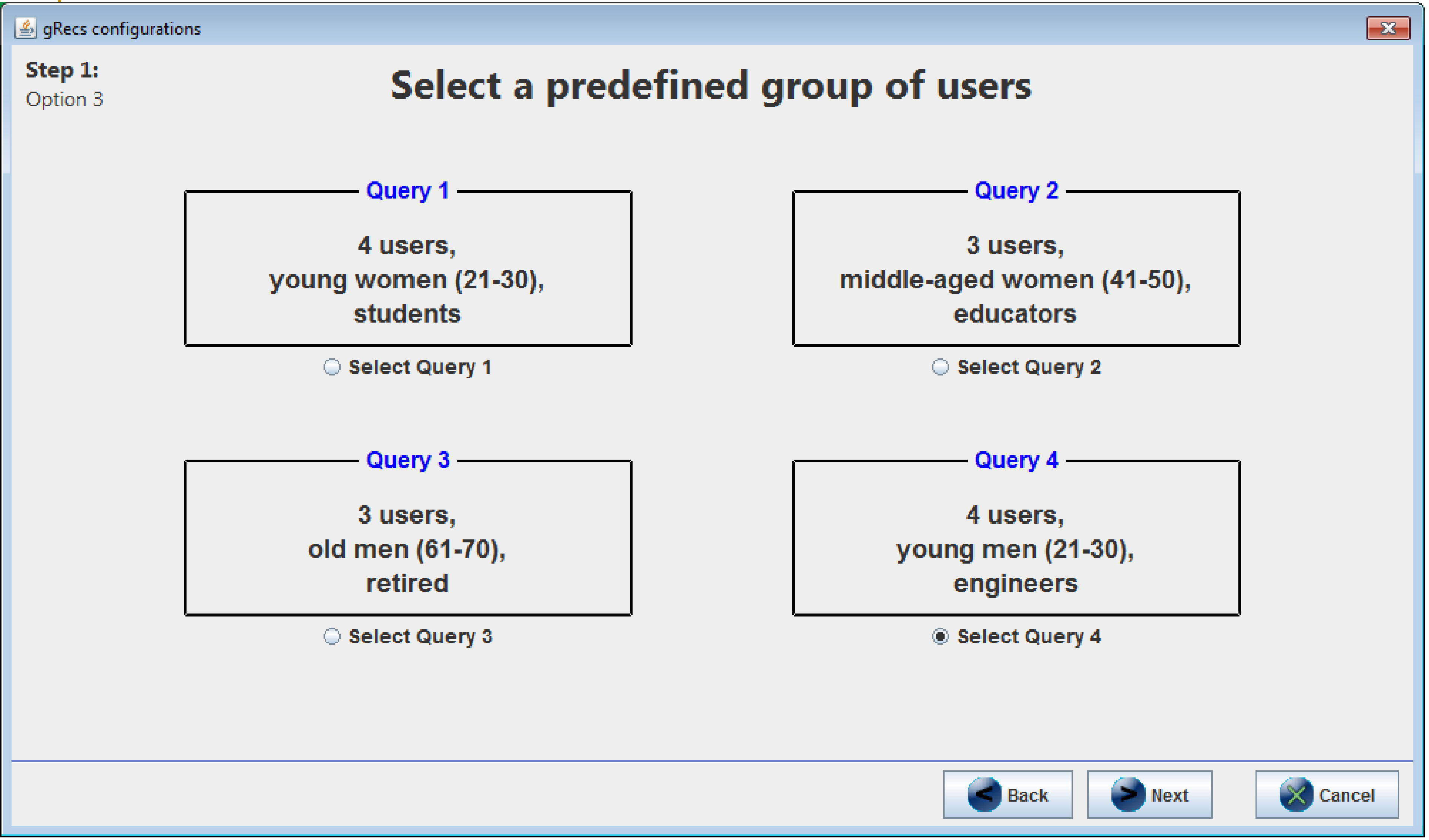
LMU
LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN
NTNU
Norwegian University of Science and Technology
Continue!



What movie shall we watch tonight?

Step1: Select your company!

- ○ Select users from the DB
- ○ Select users that qualify specific criteria
- [✓ ● Select a predefined group of users]



gRecs configurations

Step 1:
Option 3

Select a predefined group of users

Query 1
4 users,
young women (21-30),
students

○ Select Query 1

Query 2
3 users,
middle-aged women (41-50),
educators

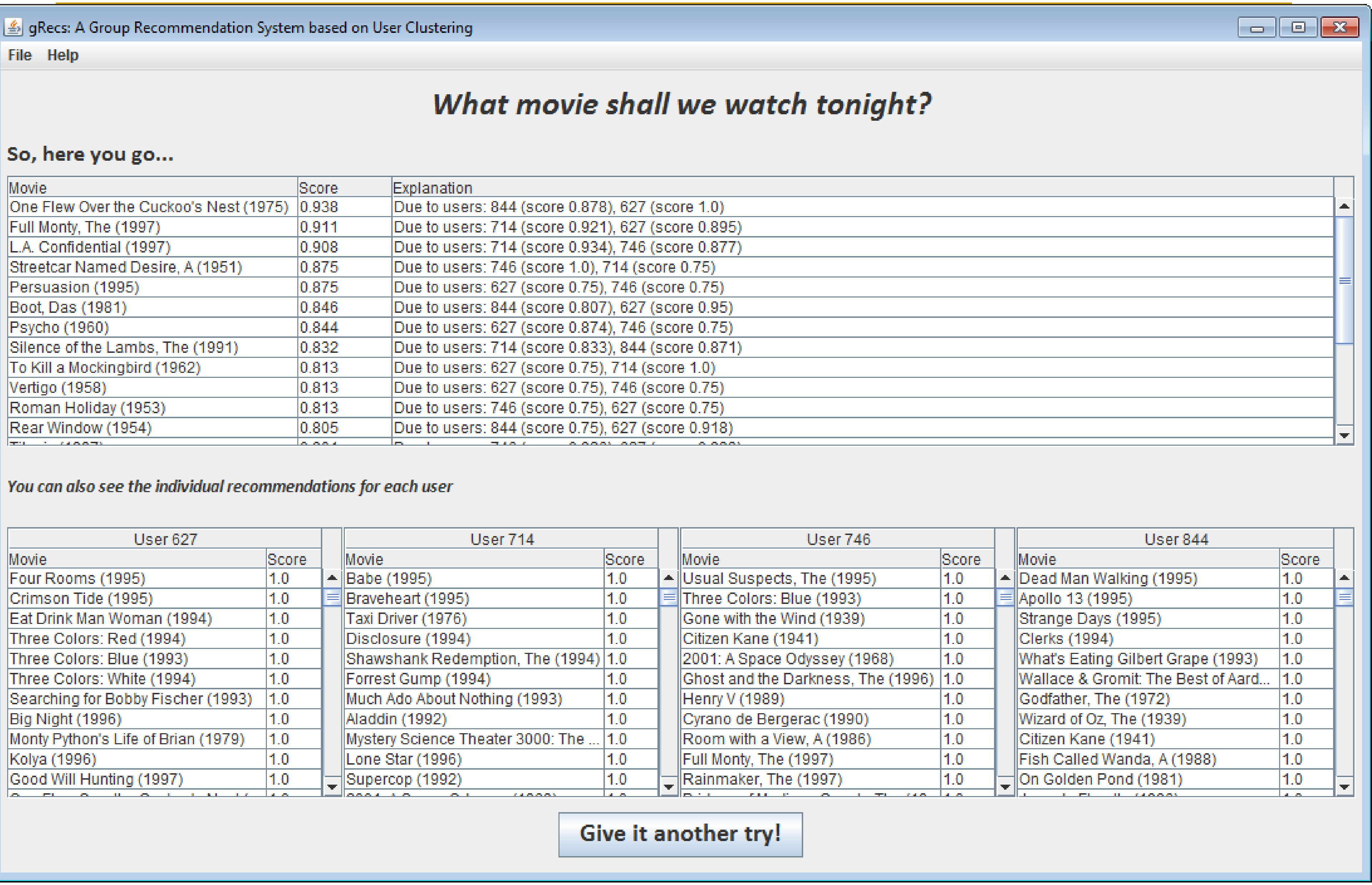
○ Select Query 2

Query 3
3 users,
old men (61-70),
retired

○ Select Query 3

Query 4
4 users,
young men (21-30),
engineers

[✓ Select Query 4]


gRecs: A Group Recommendation System based on User Clustering
File Help

What movie shall we watch tonight?

So, here you go...

Movie	Score	Explanation
One Flew Over the Cuckoo's Nest (1975)	0.938	Due to users: 844 (score 0.878), 627 (score 1.0)
Full Monty, The (1997)	0.911	Due to users: 714 (score 0.921), 627 (score 0.895)
L.A. Confidential (1997)	0.908	Due to users: 714 (score 0.934), 746 (score 0.877)
Streetcar Named Desire, A (1951)	0.875	Due to users: 746 (score 1.0), 714 (score 0.75)
Persuasion (1995)	0.875	Due to users: 627 (score 0.75), 746 (score 0.75)
Boot, Das (1981)	0.846	Due to users: 844 (score 0.807), 627 (score 0.95)
Psycho (1960)	0.844	Due to users: 627 (score 0.874), 746 (score 0.75)
Silence of the Lambs, The (1991)	0.832	Due to users: 714 (score 0.833), 844 (score 0.871)
To Kill a Mockingbird (1962)	0.813	Due to users: 627 (score 0.75), 714 (score 1.0)
Vertigo (1958)	0.813	Due to users: 627 (score 0.75), 746 (score 0.75)
Roman Holiday (1953)	0.813	Due to users: 746 (score 0.75), 627 (score 0.75)
Rear Window (1954)	0.805	Due to users: 844 (score 0.75), 627 (score 0.918)

You can also see the individual recommendations for each user

User 627	
Movie	Score
Four Rooms (1995)	1.0
Crimson Tide (1995)	1.0
Eat Drink Man Woman (1994)	1.0
Three Colors: Red (1994)	1.0
Three Colors: Blue (1993)	1.0
Three Colors: White (1994)	1.0
Searching for Bobby Fischer (1993)	1.0
Big Night (1996)	1.0
Monty Python's Life of Brian (1979)	1.0
Kolya (1996)	1.0
Good Will Hunting (1997)	1.0

User 714	
Movie	Score
Babe (1995)	1.0
Braveheart (1995)	1.0
Taxi Driver (1976)	1.0
Disclosure (1994)	1.0
Shawshank Redemption, The (1994)	1.0
Forrest Gump (1994)	1.0
Much Ado About Nothing (1993)	1.0
Aladdin (1992)	1.0
Mystery Science Theater 3000: The ...	1.0
Lone Star (1996)	1.0
Supercop (1992)	1.0

User 746	
Movie	Score
Usual Suspects, The (1995)	1.0
Three Colors: Blue (1993)	1.0
Gone with the Wind (1939)	1.0
Citizen Kane (1941)	1.0
2001: A Space Odyssey (1968)	1.0
Ghost and the Darkness, The (1996)	1.0
Henry V (1989)	1.0
Cyrano de Bergerac (1990)	1.0
Room with a View, A (1986)	1.0
Full Monty, The (1997)	1.0
Rainmaker, The (1997)	1.0

User 844	
Movie	Score
Dead Man Walking (1995)	1.0
Apollo 13 (1995)	1.0
Strange Days (1995)	1.0
Clerks (1994)	1.0
What's Eating Gilbert Grape (1993)	1.0
Wallace & Gromit: The Best of Aard...	1.0
Godfather, The (1972)	1.0
Wizard of Oz, The (1939)	1.0
Citizen Kane (1941)	1.0
Fish Called Wanda, A (1988)	1.0
On Golden Pond (1981)	1.0

Thank you for your attention!

Questions?