Music Recommendation with Collaborative Filtering on Last.fm Data

Karel Beckeringh, s2600358
Wessel Reijngoud, s2580748
Nik van ’t Slot, s2536420

Contents

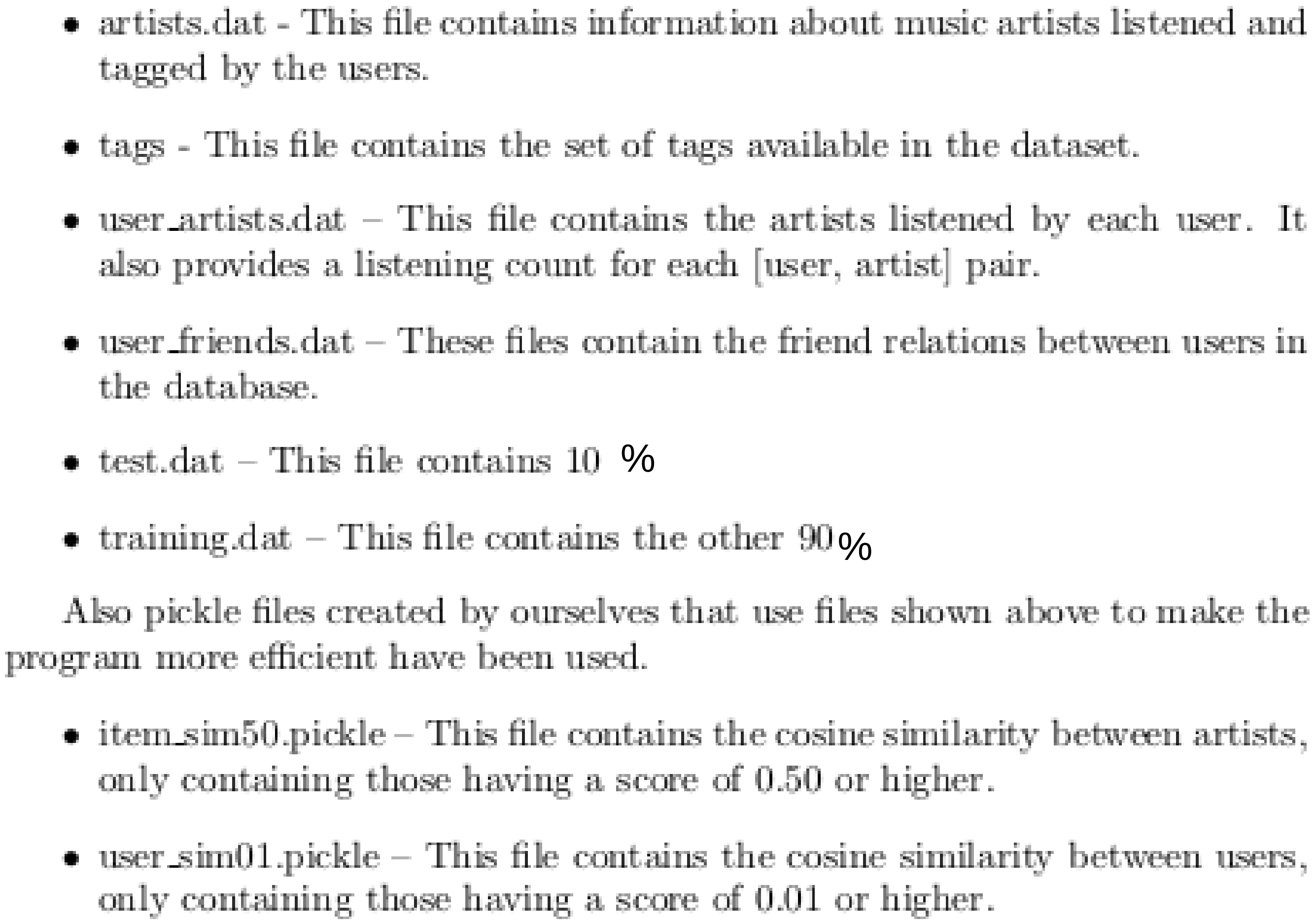
- Introduction
- Materials
- Methodology
- Evaluation
- Results
- Analysis
- Description of individual contributions

Introduction

- Last.fm, website counts listens
- Dataset is rather unique and appeals to us on a personal level
- Friend relations
- User-based, item-based or a combination of techniques
- Earlier research
- What (combination of) recommender technique(s) generates the best results in recommending artists to users?

Materials

- The dataset used is mostly the same as found in HetRec (1)



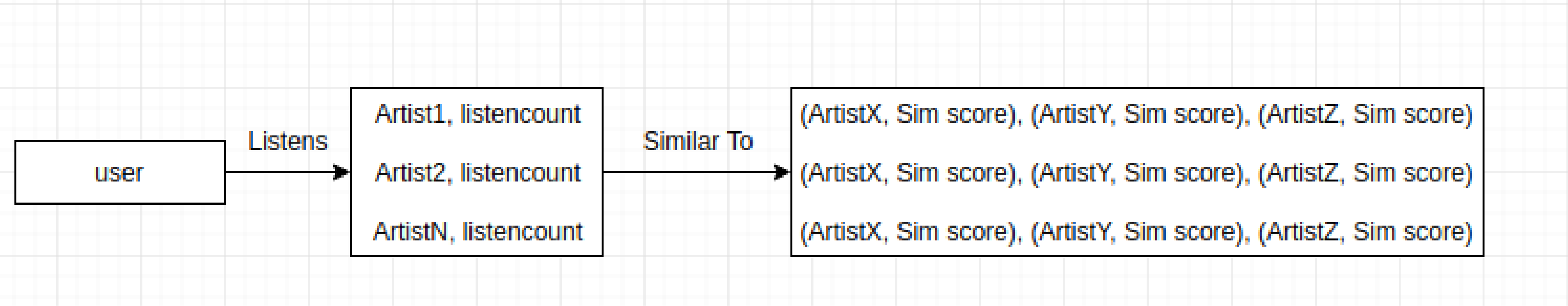
- artists.dat - This file contains information about music artists listened and tagged by the users.
- tags - This file contains the set of tags available in the dataset.
- user_artists.dat – This file contains the artists listened by each user. It also provides a listening count for each [user, artist] pair.
- user_friends.dat – These files contain the friend relations between users in the database.
- test.dat – This file contains 10 %
- training.dat – This file contains the other 90%

Also pickle files created by ourselves that use files shown above to make the program more efficient have been used.

- item_sim50.pickle – This file contains the cosine similarity between artists, only containing those having a score of 0.50 or higher.
- user_sim01.pickle – This file contains the cosine similarity between users, only containing those having a score of 0.01 or higher.

Methodology

Item based



user

Listens

Artist1, listencount

Artist2, listencount

ArtistN, listencount

Similar To

(ArtistX, Sim score), (ArtistY, Sim score), (ArtistZ, Sim score)

(ArtistX, Sim score), (ArtistY, Sim score), (ArtistZ, Sim score)

(ArtistX, Sim score), (ArtistY, Sim score), (ArtistZ, Sim score)

Sorted by listencount

Sorted by similarity score

Weight = Listencount * Similarity score

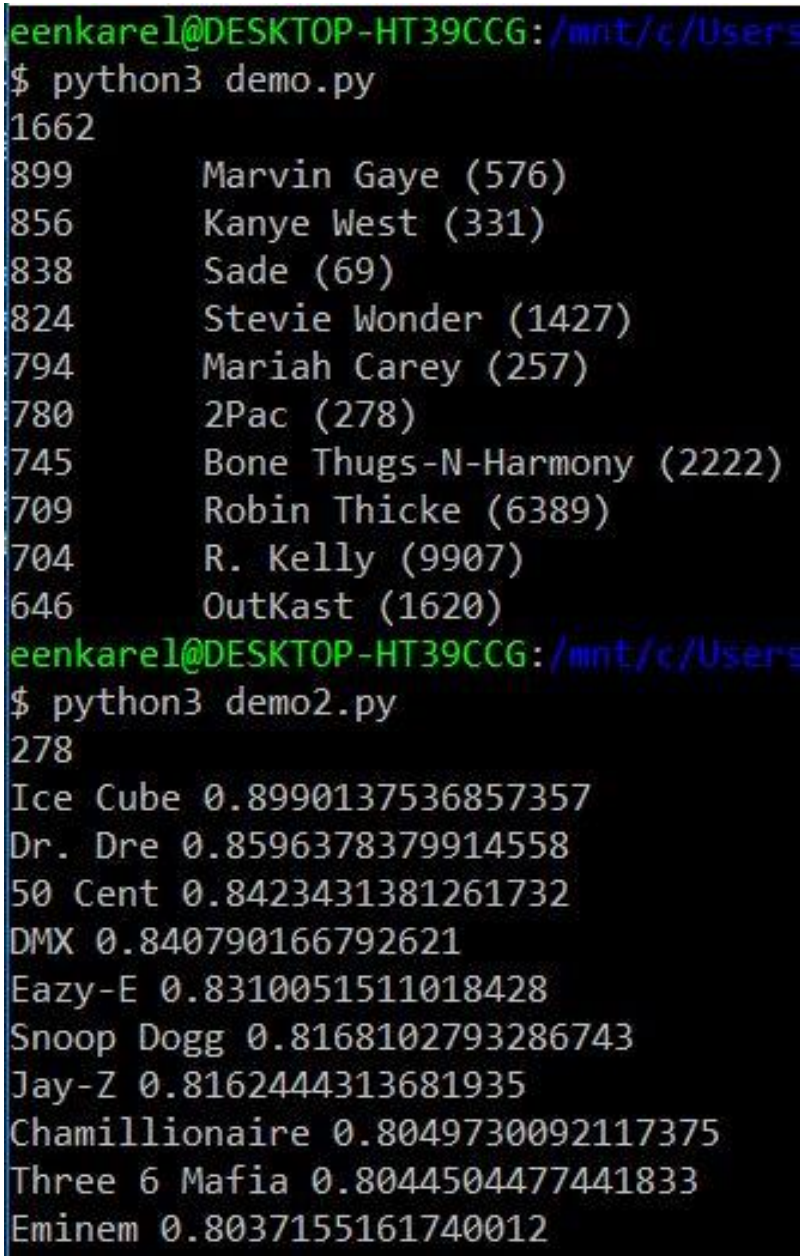
Heighest weights recommended

N=30 most efficient

Methodology

Item based

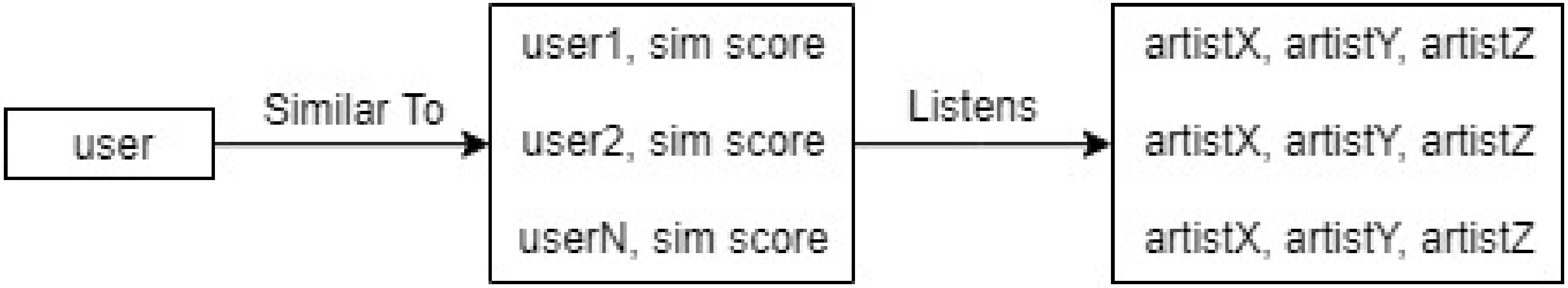
Demo:



```
eenkarel@DESKTOP-HT39CCG:/mnt/c/Users
$ python3 demo.py
1662
899      Marvin Gaye (576)
856      Kanye West (331)
838      Sade (69)
824      Stevie Wonder (1427)
794      Mariah Carey (257)
780      2Pac (278)
745      Bone Thugs-N-Harmony (2222)
709      Robin Thicke (6389)
704      R. Kelly (9907)
646      OutKast (1620)
eenkarel@DESKTOP-HT39CCG:/mnt/c/Users
$ python3 demo2.py
278
Ice Cube 0.8990137536857357
Dr. Dre 0.8596378379914558
50 Cent 0.8423431381261732
DMX 0.840790166792621
Eazy-E 0.8310051511018428
Snoop Dogg 0.8168102793286743
Jay-Z 0.8162444313681935
Chamillionaire 0.8049730092117375
Three 6 Mafia 0.8044504477441833
Eminem 0.8037155161740012
```

Methodology

User-based


user
Similar To
user1, sim score
user2, sim score
userN, sim score
Listens
artistX, artistY, artistZ
artistX, artistY, artistZ
artistX, artistY, artistZ

Sorted by similarity score

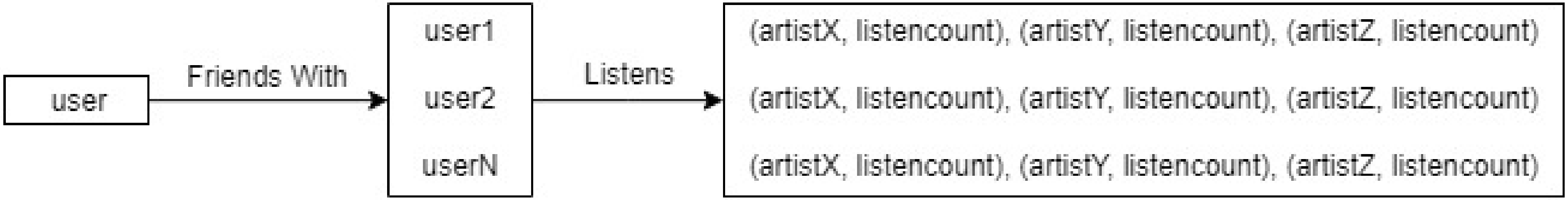
Weight = similarity score

Heighest weights recommended

N=34 most accurate

Methodology

Friend-based



user

Friends With

user1

user2

userN

Listens

(artistX, listencount), (artistY, listencount), (artistZ, listencount)

(artistX, listencount), (artistY, listencount), (artistZ, listencount)

(artistX, listencount), (artistY, listencount), (artistZ, listencount)

Considering all friends, no sorting

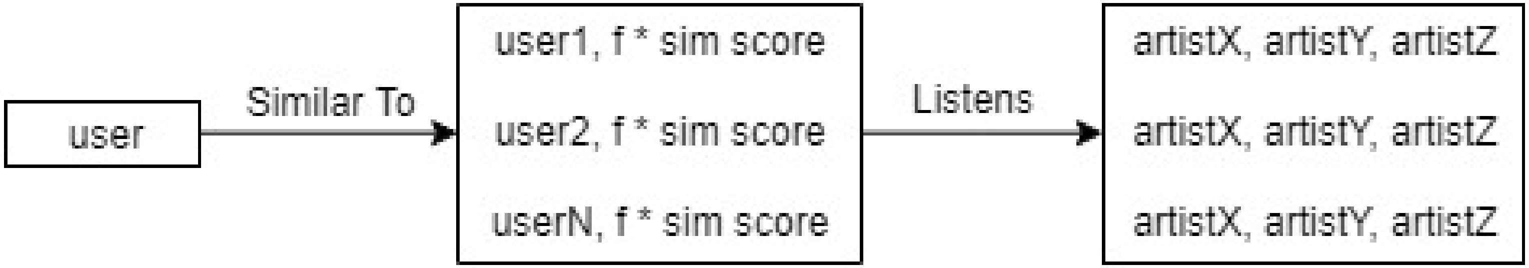
Weight = listencount

Heighest weights recommended

Methodology

Combinations

User-based with friend priority



user

Similar To

user1, f * sim score

user2, f * sim score

userN, f * sim score

Listens

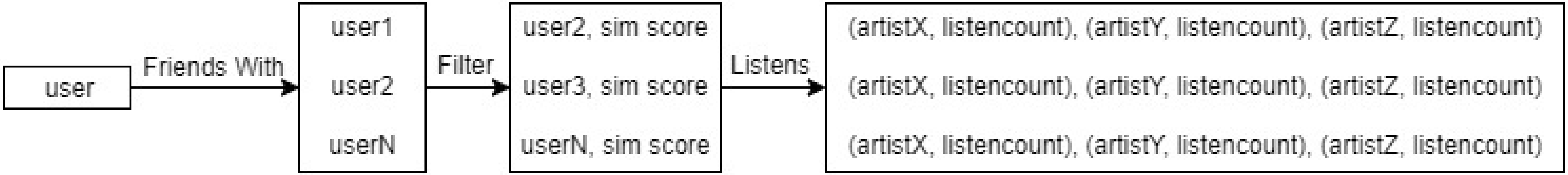
artistX, artistY, artistZ

artistX, artistY, artistZ

artistX, artistY, artistZ

(f = 5 for friends, 1 for non-friends)

Friend-based with similarity filter



user

Friends With

user1

user2

userN

Filter

user2, sim score

user3, sim score

userN, sim score

Listens

(artistX, listencount), (artistY, listencount), (artistZ, listencount)

(artistX, listencount), (artistY, listencount), (artistZ, listencount)

(artistX, listencount), (artistY, listencount), (artistZ, listencount)

(friends with a similarity < 0.01 are discarded)

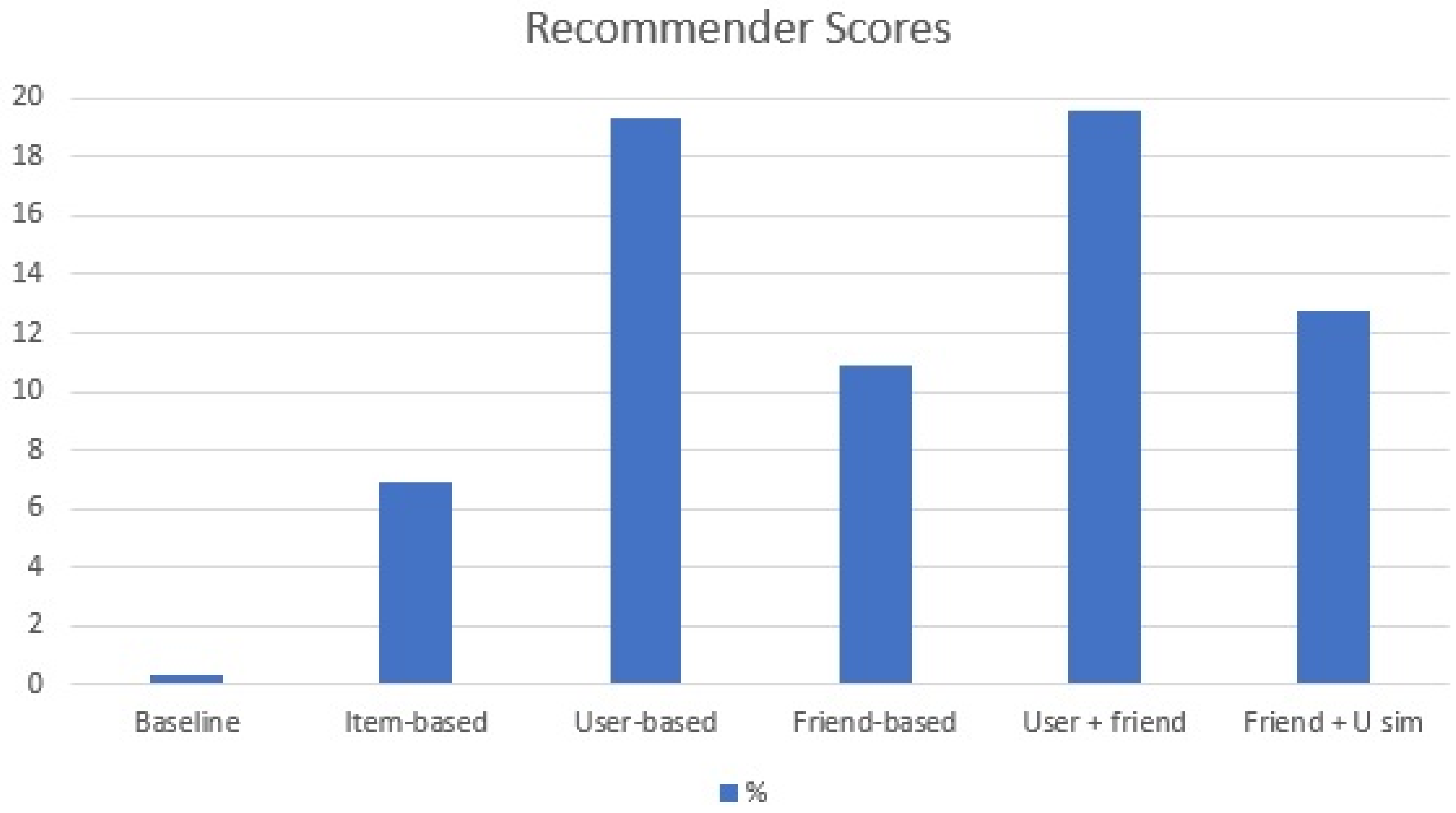
Evaluation

- user_artist.dat split → 10% test 90% training
- Baseline Random recommendation
- Like assignment 3: predict missing artists
- Accuracy = # hits / (# hits + misses)

Results

	Baseline	Item-Based	User-Based	Friend-Based	User + friend	Friend + User
Correct out of 9283	28	643	1794	1010	1823	1186
Accuracy	0.30%	6.93%	19.33%	10.88%	19.64%	12.78%
Time in s	13.127	54.887	20.639	6.929	22.813	6.953

Results


Recommender Scores
20
18
16
14
12
10
8
6
4
2
0
Baseline
Item-based
User-based
Friend-based
User + friend
Friend + U sim
■ %

Analysis

- All better than baseline
- Item-based low accuracy
- User-based (with friend priority) best results
- Way of evaluating could be better
- False misses

Contributions

- Research proposal used – Karel

- User / Friend based – Karel
- Item based – Karel, Wessel, Nik
- Report – Karel, Wessel, Nik
- Presentation – Karel, Wessel, Nik