Milestone 4

Boesinger Léopaul, Egli Marc, Khalfi Yassine
CS-421 EPFL

May 18, 2021

1 Our approach

Our goal is to create a music playlist that fits the taste of a group, given previous listening habits of the group. In order to achieve this, we want to apply both machine learning techniques (such as SVD recommender systems), and heuristics (using the genre, and musical features such as danceability, energy and tempo). Our approach is separated in several parts and here is a diagram describing the different parts of our group system recommender.

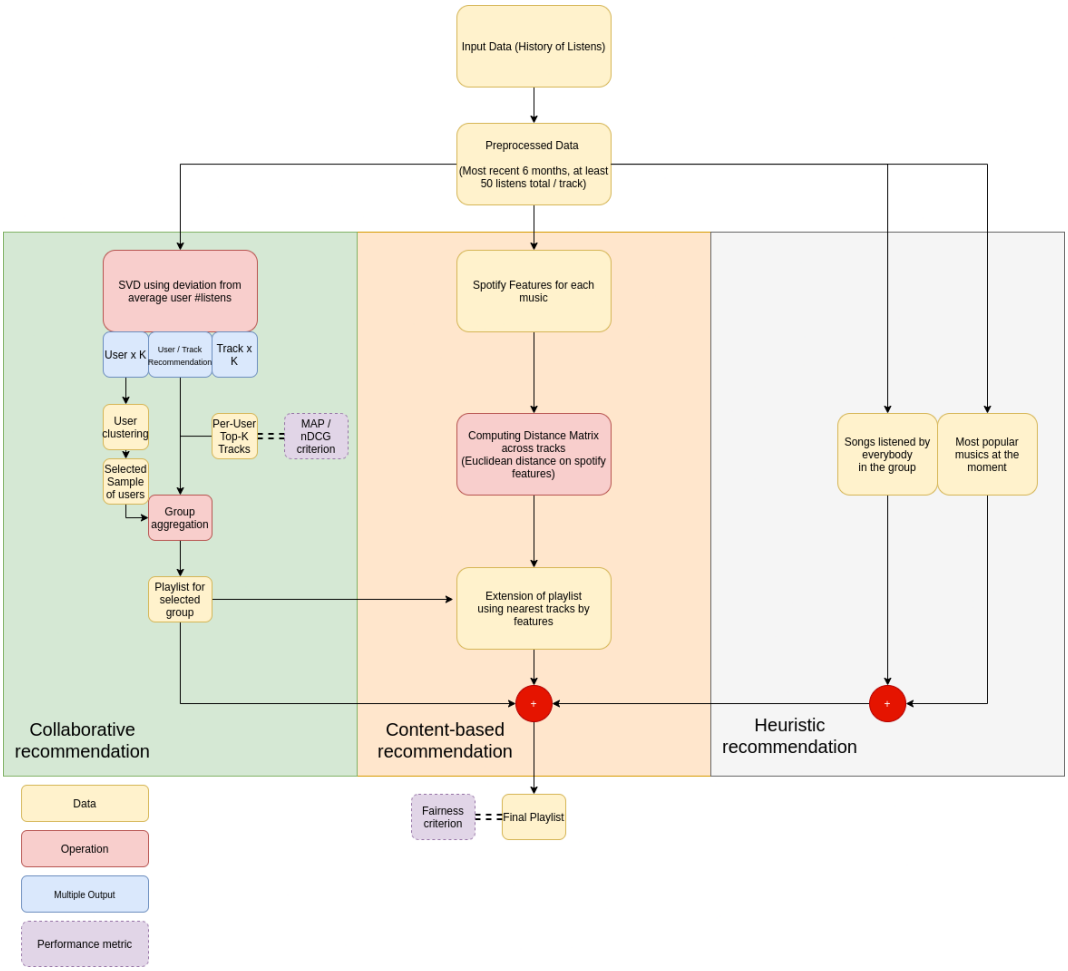

Figure 1: Recommender Flow

As you can see in our diagram, our recommendation relies on several mechanisms: Collaborative recommendation where we use SVD and PMF to find recommendations thanks to the information of the community of users, Content Based recommendation where we use the features of the music to create new recommendations by searching for music similar to the most listened songs, Heuristic recommendation which allows us to recommend to the group the music/genre/artists commonly listened.

In this report we will describe the results obtained for each of these parts.

2 Heuristic recommendation

2.1 Recommendation based on common songs/artists

When we want to create a common playlist for a group of people, a basic idea would be to see what artists, music or genre we listen to in common. In this heuristic part, this is what we do first, we try to find the artists listened to in common by all the users. Then we can take music from these artists to create the playlist of the group.

In this example to provide results, we will somewhere describe a typical situation. Let's imagine that you are on a road trip with 3 friends and that you want to create a common playlist for the trip. Starting from your listening, we will first try to find the artists that you listen to all 4, by executing this code for 4 users taken at random in our dataset we end up with these data:

```
Camera Obscura, Amy Winehouse, Mgmt, Beck, Bloc Party, Sufjan Stevens, Michael Jackson, Iron & Wine, Arctic Monkeys, Yeah Yeah Yeahs,
The Shins, Daft Punk, Damien Rice, Muse, Abba, Radiohead, Hot Chip, Belle And Sebastian,
```

Figure 2: Common artists

We find a list of artists that we have all listened to, not necessarily the same music but it is a base that will allow us to put in the playlist artists that we have all listened.

If we do this operation in a more restrictive way putting only the music that we have all listened to in common and therefore it will allow us to put in our playlist music that will be relevant for all users. By executing the code relative to this part with the same 4 users we find ourselves with the following results:

```
The Boy With The Arab Strap, Dancing Queen, Kids, Jigsaw Falling Into Place, Rehab, Time To Pretend, Step Into My Office, Baby,
```

Figure 3: Common songs

This first part allows us to add some music to our playlist which are relevant for the users.

2.2 Recommendation based on popular songs

In this heuristic part, another way to feed our playlist is to add popular music. Here we start from the principle that popular music is popular because it appeals to a large number

of people and we start from the principle that proposing to our users that they are popular to a large number of people and that they have a great chance to be enjoyed by our users. Moreover we bring a nuance, we add popular music in the genres that our users listen to, so if our users only listen to classical music it is useless to propose them pop music. A first step is to find the genres listened by all the users, if we look for the genres listened by all the users for the same group as in the previous part, we find the following result:

```
rock, art pop, electro, psychedelic rock, scottish rock, australian dance, vocal jazz, new rave, dance pop, brill building pop, twee p
op, swedish indie pop, electroclash, power pop, hip pop, electropop, country rock, melancholia, electronica, freak folk, stomp and hol
ler, experimental pop, soul, british soul, britpop, glam rock, dance rock, chillwave, beatlesque, shoegaze, synthpop, new wave pop, fu
nk, swedish pop, modern rock, la indie, permanent wave, anti-folk, singer-songwriter, piano rock, pop, nu metal, sheffield indie, cana
dian indie, pop rap, madchester, indie pop, post-punk, noise pop, new weird america, downtempo, modern blues rock, metropopolis, ontar
io indie, canadian singer-songwriter, brighton indie, oxford indie, mellow gold, albuquerque indie, r&b, neo soul, new romantic, alter
native metal, irish singer-songwriter, yacht rock, classic rock, rap rock, nu gaze, soft rock, indie folk, chamber psych, indie rock,
garage rock, big beat, album rock, neo mellow, alternative dance, europop, neo-synthpop, dance-punk, scottish indie, nu disco, new wav
e, classic uk pop, rap metal, acoustic pop, urban contemporary, funk metal, blues rock, dream pop, roots rock, trip hop, punk blues, n
oise rock, washington indie, folk rock, chamber pop, adult standards, hard rock, baroque pop, shimmer pop, punk, acid rock, uk post-pu
nk, pop rock, art rock, modern alternative rock, alternative rock, irish rock, indietronica, filter house, portland indie,
```

Figure 4: Common genres

So we find a large number of genres, because in our data each song can have up to ten genres. Once we have found the genres listened to by all the users, we can look at the most popular music of these genres, and we find the following results:

	musicbrainz-track-id	artist-name	track-name
1392	2b9241b3-a7c4-4866-a15d-17344cf44541	Britney Spears	Circus
1196	258bc802-8111-4d16-b081-fae0ea6aecba	Britney Spears	If U Seek Amy
173	051caac1-1f67-4733-80b5-62cb32660daa	Kings Of Leon	Sex On Fire
6188	c697b759-2ef6-43bb-a97a-2c56409abade	Franz Ferdinand	Ulysses
2853	5a536521-64ae-4a57-8afa-2a4235bc8841	Britney Spears	Womanizer

Figure 5: Popular songs among common genres

These are very well known musics that most probably will please our users.

In this first part, as we could see through the common listening and the common tastes of the users, we can build the first part of our playlist.

3 Collaborative recommendation

For our collaborative recommend system part, our operation pipeline is described in our previous milestone. To summarize we start from our dataset and first of all since we don't have a rating for each music we compute a rating for each music based on the number of listenings of the user. The rating is calculated from the deviation of the number of listens of a song from the average number of listens of the user. This rating is based on the fact that we listen more often to the music we like. Once the rating is computed for each music, we can then model our data with different algorithms and various matrix representations:

SVD, Probabilistic Matrix Factorization (PMF), Non-negative Matrix Factorization (NMF), KNN. The results obtained are below:

	SVD	PMF	NMF	User-based KNN with Baseline	Item-based KNN with Baseline
rmse	2.037814	2.075639	2.251284	2.042821	1.862981
mae	1.676423	1.671144	1.788077	1.670835	1.541124
mean_precision@k	0.522092	0.589175	0.063132	0.093707	0.185050
mean_map@k	0.590438	0.648942	0.058723	0.085155	0.183500
std_precision@k	0.358369	0.371258	0.099202	0.133381	0.217984
std_map@k	0.359221	0.371466	0.114210	0.142743	0.228116

Figure 6: Metric results of our different models

We observe that depending on the technique used the values of our performance indicators are very different. If for the rmse and Mae the values are similar for the different models, for the rank aware metrics which are the metrics to be privileged for this kind of model, the performances of the SVD and the PME are much better than the other models. We use 2 features : precision@k and map@k, where k is equal to 10. The precision@k is the percentage of the top-k items that are "relevant", while map@k is a bit more complex than that and also takes the position of each relevant item among the most relevant items. So with the PMF which is our model we have satisfactory results, for example almost 60% of the top 10 predictions are relevant. In his case, a song is relevant if the user has already listened to the artist at least once.

We can now use our PMF to recommend songs to users and create the playlist for a group of users, although we still have to fine-tune its parameters to guarantee better results.

4 Content Based Recommendations

4.1 Discovering new tracks

In order to include domain-knowledge about tracks and to not recommend only musics that were already listened, we use Spotify's API to obtain musical features about tracks. We have continuous attributes like loudness or even danceability for each track but also discrete attributes like the genre of the music. We first compute the euclidean distance between tracks using the continuous attributes. Then we compute build a second distance matrix which represents the hamming distance between tracks according to their genre. Finally, we Min-Max Scale both distance matrix between 0 and 1 and merge them together with both having weight 0,5. To retrieve the closest n tracks to a track T we simply take the tracks with the smallest distance to T in the final distance matrix. Thus, we are now able to extend a playlist for a user with new tracks. However, we have to keep in mind that we will not have any scoring for these (user, tracks) pairs. g.

	artist-name	track-name	genres
0	The Dandy Warhols	Not If You Were The Last Junkie On Earth	{'alternative rock', 'dance-punk', 'indie rock...
1719	The Dandy Warhols	Get Off	{'alternative rock', 'dance-punk', 'indie rock...
5766	The Dandy Warhols	We Used To Be Friends	{'alternative rock', 'dance-punk', 'indie rock...
432	The Dandy Warhols	Minnesoter	{'alternative rock', 'dance-punk', 'indie rock...
5859	The Dandy Warhols	Bohemian Like You	{'alternative rock', 'dance-punk', 'indie rock...
5926	Kaiser Chiefs	Always Happens Like That	{'modern rock', 'alternative rock', 'dance-pun...
4910	Kaiser Chiefs	Love'S Not A Competition (But I'M Winning)	{'modern rock', 'alternative rock', 'dance-pun...
510	Kaiser Chiefs	Never Miss A Beat	{'modern rock', 'alternative rock', 'dance-pun...
2826	Kaiser Chiefs	Ruby	{'modern rock', 'alternative rock', 'dance-pun...
6824	Kaiser Chiefs	Na Na Na Na Naa	{'modern rock', 'alternative rock', 'dance-pun...
7242	Kaiser Chiefs	Like It Too Much	{'modern rock', 'alternative rock', 'dance-pun...

Figure 7: Nearest tracks for *"Not if You Were The Last Junkie On Earth"*

We will use this to extend our final group playlist. This will ensure that the newly added tracks will satisfy the most users in the group.

5 New Experiment

5.1 Experiment on Aggregation function

After reading papers on Group recommenders, we have made the choice to use the same aggregation function as defined in [1], as a function of the Average relevance, along with the Average Pair-wise Disagreement by :

$$rel(\mathcal{G}, i) = \frac{1}{|\mathcal{G}|} \sum relevance(u, i)$$

$$dis(\mathcal{G}, i) = \frac{2}{|\mathcal{G}|(|\mathcal{G}| - 1)} \sum_{u \neq v} |relevance(u, i) - relevance(v, i)|$$

$$rating(|\mathcal{G}|, i) = \gamma * rel(|\mathcal{G}|, i) + (1 - \gamma) * (1 - dis(|\mathcal{G}|, i)), \gamma \in [0, 1]$$

Where the γ parameter is chosen by the user at the time of the prediction.
Using a γ close to zero will give more weight to minimize the disagreement, while a γ close to 1 will try to maximize the average relevance.
We now show an example for different γ values :

	track-name	artist-name
90961	The Birds	Telefon Tel Aviv
147593	Helen Of Troy	Telefon Tel Aviv
70704	Your Mouth	Telefon Tel Aviv

	track-name	artist-name
69319	Kickstart The Fight	Combichrist
941	In The Flowers	Animal Collective
170925	Nightlife	Iamx

	track-name	artist-name
17085	Closer	Nine Inch Nails
53449	The Bitter End	Placebo
199583	Pale Blue Eyes	The Velvet Underground

Figure 8: Selected sample of users' top tracks

	track-name	artist-name
0	The View	Modest Mouse
1	I Still Remember	Bloc Party
2	Milkshake	Holy Fuck
3	Mellowship Slinky In B Major	Red Hot Chili Peppers
4	Piranha	The Prodigy

	track-name	artist-name
0	I Still Remember	Bloc Party
1	Milkshake	Holy Fuck
2	Mellowship Slinky In B Major	Red Hot Chili Peppers
3	Piranha	The Prodigy
4	Hate To Say I Told You So	The Hives

	track-name	artist-name
0	Closer	Nine Inch Nails
1	The Bitter End	Placebo
2	You Are The Worst Thing In The World	Telefon Tel Aviv
3	Kickstart The Fight	Combichrist
4	505	Arctic Monkeys

Figure 9: Resulting Top 5 Tracks using respectively, a coefficient of 0, 0.5, and 1

We can see in 8 that the first user is a fan of *Telefon Tel Aviv* which is electro group contrary to user 2 and 3 that really appreciate rock. When we generate a playlist for the group of the 3 users with $\gamma = 0$, the playlist contains no artist in the top 3 of each user. This is because minimizing disagreement is done regardless of the rating given by each user to the tracks. For example, commonly not liking a track is seen as agreement in that case. On the other hand, setting γ to 1 will maximize the relevance and thus the generate playlist will contain the top average rated tracks by all users. Luckily γ can have any value between 0 and 1 and therefore can generate an adapted playlist to the group need.

5.2 Experiment on Group Size and User sampling

We have then tried experimenting on how the size of the group, and its heterogeneity impairs our results.
Using the User matrix from SVD (where SVD gives us U x S x V), we used PCA to lower its dimensionality, and then performed k-means clustering on it (with k=5 in our case, selected using the elbow method).
From the clustering, we created three Sampling methods for creating User groups : The first, divides equally the number of users in all clusters (in blue, our worst case scenario, they are all different). The second, samples all users from the biggest cluster (in red, our best case scenario, they are all close). The third, samples users uniformly (in green, average case).
We then used the samplers for different sizes of groups, and computed the average relevance of the Top-10 results the model could obtain, shown in figure 10 :

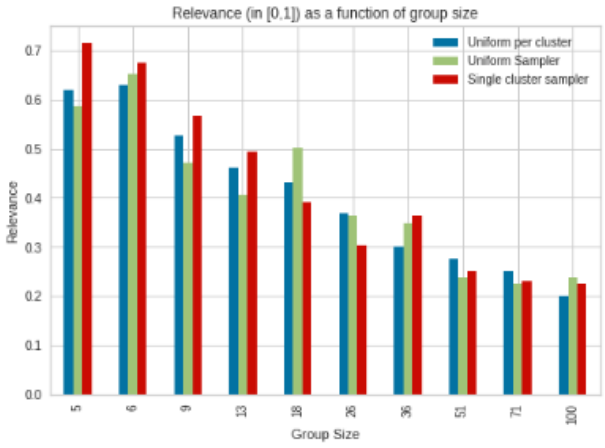

Figure 10: Relevance as a function of the group size

We see that while the sampler that takes user from a single cluster seems to perform the best with lower group sizes, the difference starts to lower with bigger groups. It however comforts us because it makes sense that for our test case (small-group parties with friends), we are able to provide results that should be relevant, as long as we assume that the SVD performs its job correctly (which seems to be the case, according the to the precision@k and map@k from before).

References

[1] Sihem Amer-Yahia et al. "Group Recommendation: Semantics and Efficiency". In: *Proc. VLDB Endow.* 2.1 (Aug. 2009), pp. 754–765. ISSN: 2150-8097. DOI: 10.14778/1687627.1687713. URL: https://doi.org/10.14778/1687627.1687713.