Capstone recommendation project

Music recommendation system
Milestone 2

Massachusetts Institute of Technology

Luis Alberto Vázquez Alfonsín

1 Refined insights

- With **little information** (song id, user id, play frequency, song title, release, artist name and year) it can be built an **interesting recommendation system**.

- The negative part of the problem is that the **datasets needed to build the recommendation systems are very big** (a million rows!), and the main part of the rows are not used, so **datasets usage are not optimized**.

-
- **The data is very sparse**, not every user listening to every song and not every song it's listened by every user. Also, there is the need of a **cutoff for performance issues to filter only songs which are listened a certain number of times** (which is 90).

- It would be a wise idea to **set a cutoff based on the behaviour of the song listeners**, for example take the average of listened songs in a day, and from that, set the cutoff to an specific number, both for time performance and for better recommendations issues.

  - Ideally, the **recommendation system should be dynamic**, for **recommending the lastest released songs and catch the local or global trends**, so I think to work with last.fm or Spotify API's should be a good idea. This dataset is no dynamic and recommends only songs listened in the past (years ago).

- **The dataset does not contain any music genre information**, and this is very **limiting**; so if we would know the music genre the quality of the recommendation system would improve a lot. Every person has specific music tastes and prefer a specific genre over others.

2 Comparison of various techniques and their relative performance

- In this music recommendation work several recommendations techniques have been used to recommend content to users, like:

  → User-User similarity-based collaborative filtering

  RMSE: 1.0657
  Precision: 0.374
  Recall: 0.683
  F_1 score: 0.483

  → Item-Item similarity-based collaborative filtering

  RMSE: 1.0689
  Precision: 0.4
  Recall: 0.545
  F_1 score: 0.461

  → Model-based collaborative filtering

  RMSE: 1.0232
  Precision: 0.393
  Recall: 0.54
  F_1 score: 0.455

  → Cluster based recommendation

  RMSE: 1.1272
  Precision: 0.388

Recall: 0.496
F_1 score: 0.435

→ **NLP content based recommendation.**

- The **best performace in terms of F1 score is for the user-user recommendation system.**

- **The cluster based recommendation** could be used **for recommending certain type of mu**sic to clustered based users (for example, clustered users by music genres).

- **For a new user, without historic data, the function top_n_songs would be interesting to use.**

- The **item-item similarity-based collaborative filtering** I think **does not work as well as user-user because the cutoff that was set** (songs listened at least 90 times), and without this handicap I would use it as well, but I would work with a larger database of listened songs. This option could be very interesting to compare similarities over the entire song dataset.

3 Proposal for the final solution design

- The principal model I would use is the **user-user similarity-based collaborative filtering**, not only because it has given the best outcomes, otherwise I think is **important to follow recommendations from people with the same musical tastes than you**; it seems natural that if you like certain type of music you will find more interesting recommendations listening to some bands that your similar-music-taste peers listen to.

- I would use as well item-item similarity-based collaborative filtering and **cluster based recommendations to filter genres of music**, clustering people which listen to similar type of music together (genres). As well, I would use natural language processing content based recommendations for lyrics, recommending similar lyric songs for users that find **this issue important.**

  - **The data is very sparse** and to avoid storage issues I would use, for example, Spotify play count for reducing the storage complexity problems. I would try not to use much a complete matrix of user id, song id, song listened because is very inefficient. I would gather this data directly from online music service providers like Spotify or Last.fm.

- A crucial aspect of the music recommendation system would be the **time performance** so I would **store the models in a pickle variable to avoid any time performance issues**.

- I would offer the **possibility to new users to choose their favourite genres and bands**, and from that, I would recommend songs which are liked a lot by the users majority.

- Also, I would offer the **possibility to new users to gather historical music data from their personal profiles on popular websites**, like Last.fm or Spotify.