Capstone recommendation project

Music recommendation system
Milestone 1

Massachusetts
Institute of
Technology

Luis Alberto Vázquez Alfonsín

1 The context

- Music recommendation is performed on platforms like **Spotify and Last.fm** and every day becomes more and more important, as the number of songs grows and the songs exploring time is critical for an user. This id counting of the songs which an user listen to can be very large and sparse on the user-item matrix. Given that, a model is built to **recommend songs which their users neighbors or neighbors or neighbors li**sten as well.

- Music recommendation becomes important when **people don't have time to listen or explore** the enormous variety of music content available, and the help of automated recommender systems can improve the exploring time, effectiveness and quickness.

- **Searching for nice music becomes a challenging and tedious task** which takes a lot of precious time. Platforms like mentioned before recommend music based historical data and predicts suitable songs that the user is likely to enjoy. One very basic recommendation feature is predicting the next song to play (with a maximum likelihood with the user), or recommend new bands similar to the bands which the user usually prefer. Another interesting feature could be recommending music attending to the songs or albums genre information (indie, rock, pop, etc.).

2 The objectives

- The main objective is to provide a **service to users who listen to songs**, offering suitable recommendations that they may like. The music recommendations system is aim to help the **user discover song based on his neighbors music preferences**, specifically how much specific songs they listen to (number of counts per song and user becomes relevant in this context).

- It has to be collected **historical data** related to the number of songs played and the songs details: year, album, name, artist…

- It has to be **efficient and quick**. Due to the sparse matrix, it would be a good idea to save the model to avoid extra processing time, for example use the pickle variable type in Python.

- Provide the more **recent music** as possible may be important, so it could be a good idea to make the work datasets dynamic over time to obtain recent recommendations.

ed goal?

3 The key questions

- Due to the **sparse matrix**, there should be asked two questions basically: as the data also contains users who have listened to very few songs and songs listened by very few users, is it required to filter the data so that it contains users who have listened to a good count of songs and viceversa (stablishing proper cutoffs). Therefore, it is needed to stablish a minimum number of songs and users, and study both parameters to get a good balance.

- New songs are created over time, so it may be needed to work with dynamic dataset which takes new songs into account. For achieving this it could be required to work this Last.fm and Spotify API's to obtain the ultimate data, and recommend recent released song to the user based on his historical data and preferences.

- Lastly, and more importantly, it has to be selected the metrics that are important. In this case, precision with k = 5 is crucial, since it is a modified performance assessment metric for recommendation systems. The parameter k is the number of songs recommended to an user. In this case precision with k=5 can be interpreted as the fraction of the 5 recommended songs that are actually listened by an user.

4 The problem formulation

- The milestone 1 contains two datasets:

  - **song_data**, with a collection of songs id, title of the song, name of the released album, name of the artist and release year.

  - **count_data**, with a collection of songs id, users id and the play count (number of times that the user has listen to a specific song).

- The **datasets** are very large, but not every song is listened by every user and not every user listen to all the songs, so the datasets **are sparse**, one could say. This could be seen on the next statistics summary.

- Some statictics:

  - **count_df dataset** (not null or NA values)

    → Number of rows: 1048575
    → Number of unique users: 40336
    → Number of unique songs: 10.000

  - **song_df dataset** (not null or NA values)

    → Number of rows: 1000000
    → Number of unique songs: 149288