Reinforcement Learning & Content Based Music Recommendation Engine

Group - PRO

Shirish Jain 201501128
Mihir Gajera 201501127
Harsh Vyas 201501101
Maulik Patel 201501175
Adit Shah 201501179

Overview..

The purpose of our Personalized Music Recommendation Engine is to use reinforcement learning approach to build a music recommender system. We formulate the problem of interactive recommendation as a contextual multi-armed bandit, learning user preferences recommending new songs and receiving their ratings. We show that using reinforcement learning solves the problem of exploitation-exploration trade-off and the cold-start problem. We integrate the novelty of songs to the model. We have embedded a content based approach to provide best recommendations.

The problem of music recommendation is modeled as a contextual multi-armed bandit which gives us the following advantages.

- Learning incorporates user feedback (i.e. ratings of songs) into recommendations thus fitting diversity in users preferences, as opposed to context-free multi-armed bandit.
- The model can take into account both a content based approaches and Reinforcement Learning.
- The model can use any type of features.
- Using a good policy balances exploration and exploitation thus mitigating the problems of cold-start and novelty.

What is contextual multi-arm bandit?

The multi-armed bandit is a intensively studied reinforcement learning dilemma. It consists of a bandit (usually a casino slot machine) with K arms, each giving a payoff r

sampled from an unknown probability distribution p(i). The player has a defined n number of pulls and her goal is to choose wisely which arm to play in order to maximize the total payoff.
The contextual multi-armed bandit is a generalization of the multi-armed bandit where the reward obtained from each arm depends also on a context. The context contains information about the arm and the player.

The context for each arm (song) is a features vector containing the user preferences and the song features.
The reward (rating) obtained from pulling the arm (listening the song) depends on this context. When the rating is returned, the user preferences are updated depending on the song features and the rating.

Algorithm Overview

Initially we are asking a new user to give ratings to the songs we are providing.Then we are providing him 10 songs one by one.
After each song the listener will rate the song and accordingly the feature vector (theta) of the user will get updated.
Now our next recommendation will be the song whose utility is maximum.

Algorithm in depth

Content based The content based model tries to find a song that is close to the learned user preferences θ. It takes into account the song genre, song year and novelty. Novelty can be defined as if the song has been recently listened or not.
The song genre and song year can be considered as a one-hot encoded vector x. Without considering other factor, the utility of a user for a song can be represented as the following.

$$U_c = (\theta^T x)$$

Distinctive users have different preferences θ. Also, we do the hypothesis of a stationary true value for θ, meaning that while, we learn the user preference, its preferences stay the same.
The novelty is defined as with t being the current time and s an hyper-parameter defining how fast we forget and want a repetition. Obviously if we want to recommend book/movie instead of music, an higher value of s is needed. The utility for a user in term of novelty can be defined as follows.

$$U_n = \left(1 - e^{-t/s}\right)$$

The combined utility (i.e. the expected rating for a song and a user) is defined as follows.

The task of the recommender is to propose a song to the user depending on the expected ratings computed for each song and to update the user preferences θ according to the rating received from the user. We use an iterative mean.

$$\theta(t+1) = \theta(t) + r(t) \times x(t)$$

where θ(t) is the user preferences at epoch t, r(t) is the rating received for the song recommended at epoch t and x(t) is the song features for the song recommended at epoch t.

For our recommendation engine,

Utility = (userFeature).(musicFeature) x $(1- e^{-T/s})$
T = t - last_t
last _t : Last time when the song was recommended
s: Hyper Parameter (Here, s = 50)

NOTE: The value of s is small because we want songs (which are rated high) , to get repeated after some sufficient amount of time.

Exploration - Exploitation

The greedy algorithm is a widely used algorithm, for each round of the simulation,
the algorithm selects the arm (song) with the highest rating with a probability of 1 - E and selects a random arm with a probability E.
Here, E variate over time in order to maximize the exploration at first and reduce it in order to increase exploitation over time.

$$\epsilon = \frac{1}{\sqrt{t'+1}}$$

Here, we have chosen t' = t - 10 for (t<10) (t will get start from 10).
&
E = 0.3 , if (t > 10)

The **reason behind** doing this is that in the initial phase when t is low, value of E will be high that means exploration will occur in initial phases whereas as t increases, E decreases and after value of t' = 10 (i.e. t=20), value of E will be will be very low and thus the engine will **overlearn** and will not show any novelty. So, after value of t' = 10, we are fixing value of E = 0.3. So, exploration always occurs!

Now, why using t' = t-10 ?

The reason behind choosing t' = t-10 and not simply starting t from zero is that it may happen that in the initial phase, the engine recommends user a song of the genre he likes but the user still doesn't give good rating because he does not like that particular song!
So, to **reduce heavy penalty** at early stages, we are starting from t= 10.

Otherwise the engine will be very afraid to show the song to the user (even though the user like that genre).

Initial t = 10
To reduce heavy penalty at early phase,

New User Feature =
(t - 1) / t * (Old User Feature) + 1 / t * (Song Feature * Song rating)

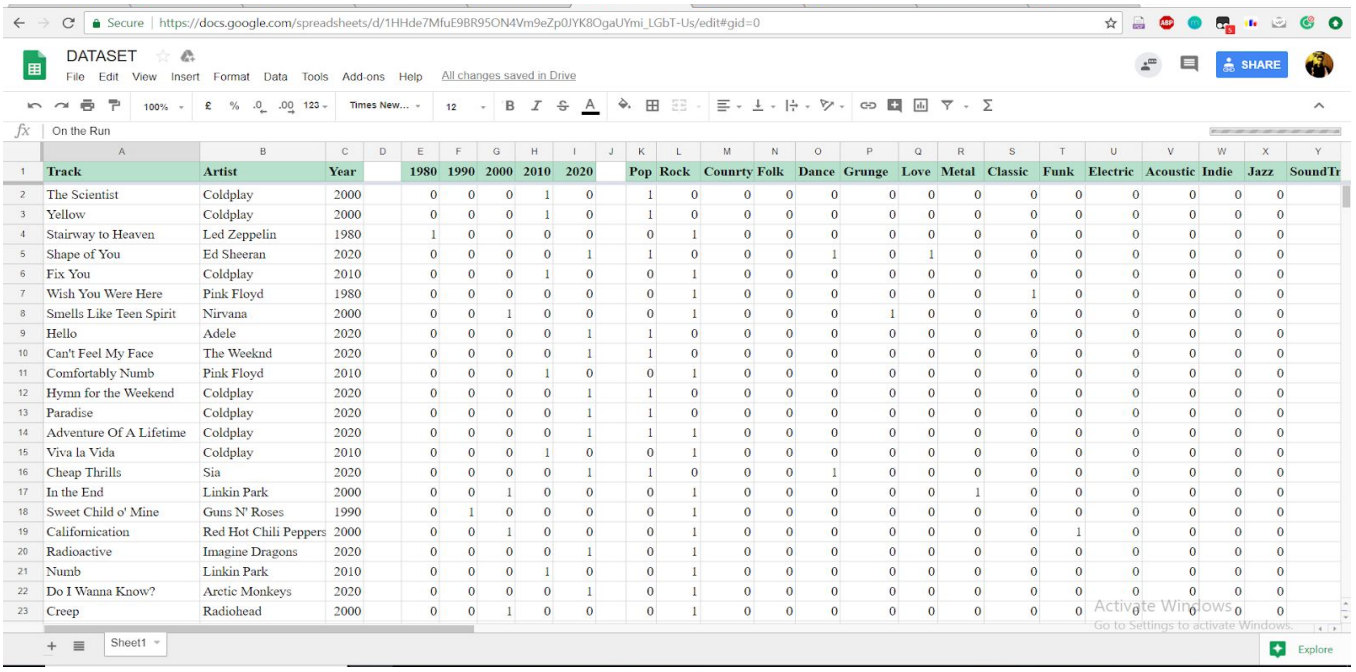
Dataset

For dataset,

- We have extracted data from last.fm using its API.
- Dataset contained 1000 track and year.
- We divided all the 1000 songs in their respective genres **(manually)**

We are adding data of userid, song name and its rating in other file. It will get updated as users will give rating to new songs.

Screenshot Of Dataset :



DATASET

Track	Artist	Year	1980	1990	2000	2010	2020	Pop	Rock	Counrty	Folk	Dance	Grunge	Love	Metal	Classic	Funk	Electric	Acoustic	Indie	Jazz	SoundTr
The Scientist	Coldplay	2000	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	
Yellow	Coldplay	2000	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	
Stairway to Heaven	Led Zeppelin	1980	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	
Shape of You	Ed Sheeran	2020	0	0	0	0	1	1	0	0	0	1	0	1	0	0	0	0	0	0	0	
Fix You	Coldplay	2010	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	
Wish You Were Here	Pink Floyd	1980	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	
Smells Like Teen Spirit	Nirvana	2000	0	0	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	
Hello	Adele	2020	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	
Can't Feel My Face	The Weeknd	2020	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	
Comfortably Numb	Pink Floyd	2010	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	
Hymn for the Weekend	Coldplay	2020	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	
Paradise	Coldplay	2020	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	
Adventure Of A Lifetime	Coldplay	2020	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	
Viva la Vida	Coldplay	2010	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	
Cheap Thrills	Sia	2020	0	0	0	0	1	1	0	0	0	1	0	0	0	0	0	0	0	0	0	
In the End	Linkin Park	2000	0	0	1	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	
Sweet Child o' Mine	Guns N' Roses	1990	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	
Californication	Red Hot Chili Peppers	2000	0	0	1	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	
Radioactive	Imagine Dragons	2020	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	
Numb	Linkin Park	2010	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	
Do I Wanna Know?	Arctic Monkeys	2020	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	
Creep	Radiohead	2000	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	

How to Run:

- Download all the folder along with dataset.
- Run python file from src folder.
- Python recommed.py

What else we could have done, given more time ?

- We could have improved our reward-penalty function, taking into account implicit ratings,half-listened songs,like-dislike features, how to rate that song if user even shares a song, etc.

- Also, we could have made it signing by FACEBOOK and therefore would have learnt more about user taste and preferences. Recommendations can also take into account the user’s age, country in which he lives, college, general interests,etc.