Python Programming and Practice

Put on your song!
: Music Recommendation System

Proposal

Date : 2023.11.03

Name : Seokyoung Kim

ID : 182207

1. Introduction

1) Background

According to the International Federation of the Phonographic Industry (IFPI)'s Global Music Report 2023, the global music industry generated $25.9 billion in revenue in 2022, up 18.5% from 2021. These revenues have been reaching new highs every year, with this year's growth being the largest since the 1990s. In addition, the 2021 Music User Survey report released by the Korea Creative Content Agency found that 71.5% of respondents said they have spent more time listening to music since the coronavirus (COVID-19) pandemic. 88.3% of all respondents said they listen to music "at least once a week," with 51.7% saying they listen to music "almost every day. The number of people streaming music has also been on the rise, with 60.5% in 2019, 63.2% in 2020, and 65% in 2021. To compete in this rapidly growing music market, differentiated service technologies are being developed by streaming companies such as Melon, Genie, and YouTube Music.

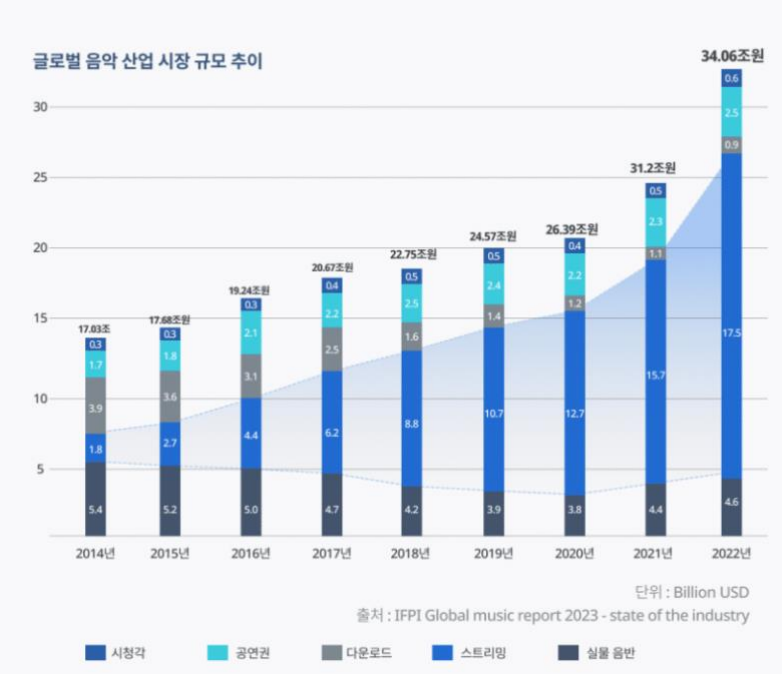

Figure 1. Global music industry market size trends

2) Project goal

In this project, we aim to build a system that recommends music to users by analyzing existing music data. It also aims to automatically create new playlists based on the user's playlists and generate titles for them.

3) Differences from existing programs

Traditional recommendation systems use other users with similar tastes to the user to recommend songs that the user has listened to. However, these systems simply recommend songs based on what you've listened to recently, rather than what you've listened to in the past, even on the same day, depending on your mood and the time of day. In addition, this requires a large enough number of users to collect data, and this limitation makes it difficult to recommend music from new users or independent artists. Especially with newer music, it can be difficult to recommend it to users before we have enough information to recommend it. In addition, the music recommendation systems currently used by many streaming apps are mainly limited to paid subscribers or song purchasers, and free users have limited access to them.

To solve these problems, this project proposes a system that recommends songs without existing user data based on a systematic analysis of music information. This is expected to reduce the gap in the music industry caused by the imbalance of user data.

2. Functional Requirement

*** The following functions can be changed during implementation.**

1) Load Dataset Function

- Download the dataset for analysis

  (1) Load dataset – music genre and file

2) Data Preprocessing Function

- Preprocessing data to extract clean features

  (1) Analyzing singer-sepecific data

  (2) Analyzing data by gender

  (3) Analyzing data by genre

  (4) Analyzing of the frequency of lyrics

  (5) Top-ranked frequency analysis

  (6) Analysis of the frequency of songwriting

3) Remove specific words Function

- Remove strange words/english combinations

  (1) Analyzing singer-sepecific data

  (2) Delete outlier or incorrect word

4) Tokenize Function

- Tokenize the words in sentences

  (1) Using tokenize library in tensorflow

  (2) Tokenize words and divide to ‘singer’, ‘genre’, etc

5) Clustering Data Function

- Clustering the data with traditional algorithm

  (1) K-means Clustering

  (2) Agglomerative Clustering

  (3) Analysis the morpheme

6) Main Function

- Set the epoch, save path, load path, samples, data file paths, batch size

  (1) Download the model (ex. GPT2LMHeadModel, kogpt2)

  (2) Download vocabulary

  (3) Call the tokenizer

  (4) Train the model using dataset

  (5) Evaluate the trained model

  (5) Save the results

7) Get a song/singer/playlist input and recommend system Function

- Recommended by receiving the song title and checking if the song is in the database you own.

(1) Recall 'doc2vec' model and run similarity

(2) Find the cluster to which each song belongs

(3) Call the tokenizer

(4) Choose randomly selected songs from songs in the cluster

(5) If the input is a singer, check the cluster that contains the most songs by the singer and select the randomly selected song

(6) Run similarly if the input is a playlist.

3. Schedule

Task	11/3	11/10	11/17	11/24
Proposal	---------->			
Function 1		---------->		
Function 2,3			---------->	
Function 4,5				---------->
Task	**12/1**	**12/8**	**12/15**	**-**
Function 6	---------->			
Function 7	---------->			
Additional Functions	---------->			
Retrospect			---------->	