Song Database

Group – 16

19CSE353 – Mining of Massive Datasets

Problem Statement

Analyze a music streaming platform's user behavior and preferences. Using the provided dataset with columns of track_name, id, popularity, artist name, artist id, artist popularity, and duration, etc., you must identify patterns and insights that can be used to improve the user experience and drive user engagement.

Dataset Description

The dataset contains songs from several playlists where each playlist is uniquely identified by the `playlist_URI`. Same songs can be a part of multiple playlists. Each song is characterised by it's track_name, id, popularity, danceability, speechiness, acousticness, instrumentalness, liveness, etc.

playlist_uri	track_name	id	popularity	main_artist_name	main_artist_id	main_artist_pop	duration_ms
37i9dQZEVXbNG2KDcFcKOF	un x100to	6pD0ufEQq0xdHSsRbg9LBK	99	Grupo Frontera	6XkjpgcEsYab502Vr1bBeW	88	194563
37i9dQZEVXbNG2KDcFcKOF	Ella Baila Sola	3qQbCzHBycnDpGskqOWY0E	85	Eslabon Armado	0XeEobZplHxzM9QzFQWLiR	86	165671
37i9dQZEVXbNG2KDcFcKOF	La Bebe - Remix	2UW7JaomAMuX9pZrjVpHAU	98	Yng Lvcas	1NNRWkhwmcXRimFYSBpB1y	81	234353
37i9dQZEVXbNG2KDcFcKOF	Flowers	4DHcnVTT87F0zZhRPYmZ3B	90	Miley Cyrus	5YGY8feqx7naU7z4HrwZM6	89	200455
37i9dQZEVXbNG2KDcFcKOF	Cupid - Twin Ver.	7FbrGaHYVDmfr7KoLIZnQ7	96	FIFTY FIFTY	4GJ6xDCF5jaUqD6avOuQT6	81	174253
...	...	...	...	...	...	...	...
37i9dQZF1DXdPec7aLTmlC	Rain On Me (with Ariana Grande)	24ySl2hOPGCDcxBxFIqWBu	74	Lady Gaga	1HY2Jd0NmPuamShAr6KMms	86	182200
37i9dQZF1DXdPec7aLTmlC	Wellerman - Sea Shanty / 220 KID x Billen Ted ...	3iw6V4LH7yPj1ESORX9RIN	73	Nathan Evans	1PKErrAhYFdfrDymGHRQRo	65	116750
37i9dQZF1DXdPec7aLTmlC	Call You Mine	2oejEp50ZzPuQTQ6v54Evp	73	The Chainsmokers	69GGBxA162lTqCwzJG5jLp	80	217640
37i9dQZF1DXdPec7aLTmlC	2step	2SUxn2O9NHL6GHGQFgwCY0	74	Ed Sheeran	6eUKZXaKkcviH0Ku9w2n3V	91	153627
37i9dQZF1DXdPec7aLTmlC	KESI - Remix	0lqCoZ168iRc9LqfrYgpZy	73	Camilo	28gNT5KBp7ljEOQoevXf9N	79	176575

Dataset Description

The dataset contains the lyrics of all **distinct** songs from several playlists where each song is uniquely identified by the `track_id`. This dataset is used to find out what the song is about by analyzing it's lyrics.

track_name	main_artist_name	lyrics
Put Your Records On	Ritt Momney	"Put Your Records On" by Ritt Momney:\n 31 ...
Can't Feel My Face	The Weeknd	"Can’t Feel My Face" by The Weeknd:\n 179 C...
Dancing In The Moonlight (feat. NEIMY)	Jubël	"Dancing in the Moonlight" by Jubël:\n 7 Co...
I'm Good (Blue)	David Guetta	"I’m Good (Blue)" by David Guetta & Bebe Rexha...
Bad Romance	Lady Gaga	"Bad Romance" by Lady Gaga:\n 121 Contribut...
God's Plan	Drake	"God’s Plan" by Drake:\n 633 ContributorsTr...
Love The Way You Lie	Eminem	"Love the Way You Lie" by Eminem:\n 340 Con...
Sunshine	OneRepublic	"Sunshine" by OneRepublic:\n 20 Contributor...
Do I Wanna Know?	Arctic Monkeys	"Do I Wanna Know?" by Arctic Monkeys:\n 212...
El Azul	Junior H	"El Azul" by Junior H & Peso Pluma:\n 8 Con...
Let Me Love You	DJ Snake	"Let Me Love You" by DJ Snake:\n 161 Contri...
There's Nothing Holdin' Me Back	Shawn Mendes	"There’s Nothing Holdin’ Me Back" by Shawn Men...
Die For You	The Weeknd	"Die For You" by The Weeknd:\n 228 Contribu...
Quevedo: Bzrp Music Sessions, Vol. 52	Bizarrap	"Quevedo: Bzrp Music Sessions, Vol. 52" by Biz...
Rain On Me (with Ariana Grande)	Lady Gaga	"2021 Nominees" by Grammys:\n 4 Contributor...
What A Man Gotta Do	Jonas Brothers	"What a Man Gotta Do" by Jonas Brothers:\n ...
This Is What You Came For	Calvin Harris	"This Is What You Came For" by Calvin Harris:\...
Green Green Grass	George Ezra	"Green Green Grass" by George Ezra:\n 14 Co...
Head & Heart (feat. MNEK)	Joel Corry	"Today’s Top Hits 9/11/20" by Spotify:\n 4 ...
Sucker	Jonas Brothers	"Sucker" by Jonas Brothers:\n 165 Contribut...

Subproblem 1

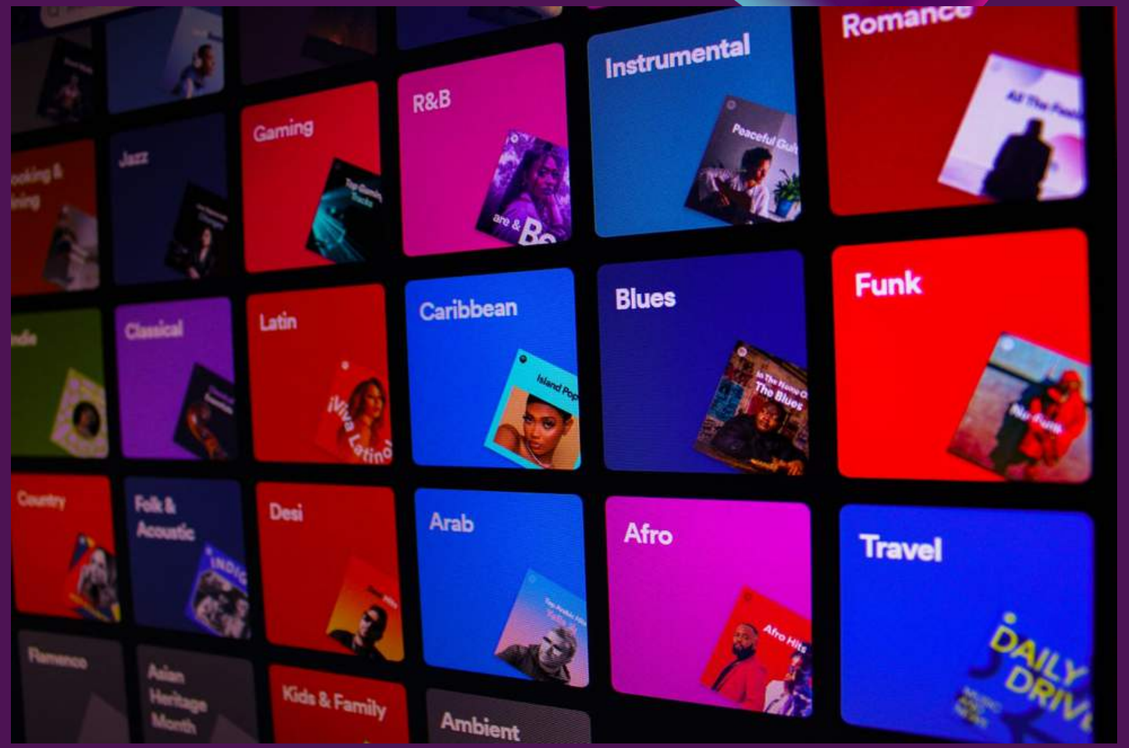
Find out what the song lyrics are about.

TF-IDF

It can be used to identify important or distinctive words or terms in the lyrics of songs in the database by assigning each term a score that reflects its importance.

Therefore, we can relate how close the name of the song is to what the lyrics say.

This is useful for writing news articles about new song releases



Example

This example finds out the most important words from song 'Hall of Fame' by The Script, using the `TfidfVectorizer` object from sklearn library.

```
lyrics = """
Yeah, you can be the greatest
You can be the best
You can be the King Kong banging on your chest
You can beat the world
You can beat the war
You can talk to God, go banging on his door
You can throw your hands up
You can beat the clock (yeah)
You can move a mountain
You can break rocks
You can be a master
Don't wait for luck
Dedicate yourself and you gon' find yourself
Standing in the hall of fame (yeah)
And the world's gonna know your name (yeah)
'Cause you burn with the brightest flame (yeah)
And the world's gonna know your name (yeah)
And you'll be on the walls of the hall of fame
"""
```

By examining the TF-IDF scores, we can identify the most important or distinctive words or phrases in the lyrics of "Hall of Fame" by The Script featuring will.i.am.

Highest TF-IDF scores, are "hall" (score of 0.666), "world" (score of 0.666), "flame" (score of 0.333), and "brightest" (score of 0.167). These terms are likely to be the most distinctive or significant in the lyrics of the song, as they have the highest TF-IDF scores.

Subproblem 2

Average popularity of a playlist

Map-Reduce

Map phase

In this phase, we map each song in the playlist to a key-value pair, where the key is the song ID and the value is a tuple containing the popularity of the song and the value 1. For example, the key-value pairs might look like this:

(playlist_id_1, (popularity_1, 1))
(playlist_id_2, (popularity_2, 1))
(playlist_id_3, (popularity_3, 1))

Shuffle phase

In this phase, all key-value pairs are sorted and grouped by key. For example, the pairs might be grouped like this:

(playlist_id_1, [(popularity_1, 1), (popularity_1, 1), ...])
(playlist_id_2, [(popularity_2, 1), (popularity_2, 1), ...])
(playlist_id_3, [(popularity_3, 1), (popularity_3, 1), ...])

Reduce phase

In this phase, we compute the average popularity of each song by summing up the popularity values and the count values for each key (song ID), and then dividing the sum of the popularity values by the sum of the count values. For example, the output might look like this:

(playlist_id_1, avg_popularity_1)
(playlist_id_2, avg_popularity_2)
(playlist_id_3, avg_popularity_3)

Subproblem 3

Finding songs that are similar to a given query song.

Locality Sensitive Hashing (LSH)

Locality-sensitive hashing (LSH) can be used in a songs dataset for similarity search. The idea is to represent each song as a vector of features (such as artist, danceability, instrumentals, etc.), and then use LSH to map similar vectors to the same bucket with high probability.

This can be useful for finding songs that are similar to a given query song

How it is used?

Feature extraction

Extract features from the songs in the database that are relevant for playlist creation, such as song tempo, genre, artist name, and duration

Hashing

Apply LSH to hash the feature vectors of the songs into a fixed number of hash buckets. Similar songs are expected to be hashed to the same or nearby buckets.

Candidate selection

find the set of songs that are in the same or nearby hash buckets as the chosen song. This set of songs are the candidates for potential matches for the query song.

Verification

We can then compute the similarity score between the query song and each potential match using a distance metric like Signature similarity or Jaccard similarity. We can then rank the matches by similarity score and return the top matches as the search results.

Subproblem 4

Recommend songs which are frequently listened together with other songs in a playlist

PCY ALGORITHM

Given multiple music playlist datasets consisting of various attributes such as track name, id, popularity, artist name, artist id, artist popularity, duration, and more. Your task is to recommend the best playlist to new users.

Applications

01

it can find all frequent itemsets of different lengths, even those with a low frequency

02

It is also easy to implement and has a straightforward logic

03

frequent itemset mining algorithm

Steps Involved

01

Data Preparation

datasets into a transactional format

02

Counting Frequencies and Hashing

- Count the frequencies of individual items
- Partition the itemsets into buckets based on their hash values
- Count the frequencies of candidate itemsets that hash to the same bucket
- Generate frequent itemsets from the counted candidate itemsets [MSC]

03

Playlist Recommendation

we can use the frequent itemsets to recommend playlists to new users based on their preferences

Example

Playlist 1: {Track A, Track B, Track C}
Playlist 2: {Track A, Track D, Track E}
Playlist 3: {Track B, Track D, Track F}

minimum support threshold to 2

Track A: 2
Track B: 2
~~Track C: 1~~
Track D: 2
~~Track E: 1~~
~~Track F: 1~~

Bucket 0: {Track A, Track B}
Bucket 1: {Track A, Track D}
Bucket 2: {Track B, Track D}

{Track A, Track B}: 2
{Track A, Track D}: 2
{Track B, Track D}: 2

Collaborative Filtering

Collaborative filtering is effective in generating personalized recommendations but may suffer from the cold start problem.

PCY Algorithm

The PCY algorithm is effective in finding the most frequent itemsets but may not be effective in situations where the dataset is sparse or where the items are highly diverse.

Subproblem 5

Searching for a song in a playlist

Bloom Filter

Searching a song can be efficiently done using Bloom filter. The set of songs in the dataset would be represented by the bit array, and multiple hash functions could be used to map each song to a set of positions in the array. They provide fast membership testing with low memory requirements.

When a new song is added to the dataset, its hash values would be used to set the corresponding positions in the bit array. When a query is made to check whether a particular song is in the dataset, the hash values for the song would be used to check the corresponding positions in the bit array.

Hash functions:

A hash function for a Bloom filter on a song dataset needs to take in a song (which could be represented as a string of characters or a set of features) and return an integer value that can be used to index into the bit array

Eg: djb2 hash function, which iterates over a string and updates the hash value based on the ASCII value.

Dealing with False Positives

1. **Choose the size of the bit array:** The size of the bit array is an important parameter of the Bloom filter that determines its memory usage and accuracy. A larger bit array reduces the false positive rate but requires more memory.
2. **Choose the number of hash functions:** Typically, a larger number of hash functions reduces the false positive rate but increases the computational cost. The number of hash functions should be chosen based on the size of the bit array and the expected number of items in the dataset.

Thank you!

Hariharan – CB.EN.U4CSE20320
Kishore Kumar – CB.EN.U4CSE20328
Nithin – CB.EN.U4CSE20342
Sanjith Raghav – CB.EN.U4CSE20355
Sujith Roshan – CB.EN.U4CSE20363