Ashini Parikh
Maya Beydoun
SI 206 - Fall 2023

SI Final Project Report

Github link: https://github.com/mayabeydoun/SI206_Final.git

1. Original Goals:

Initial APIs that we planned to use:

- Twitter, Spotify

Data to Gather:

- Tweet engagement (likes, views, comments, followers, etc) from celebrity tweets.
- Spotify data (monthly listeners, streams, followers) from top artists.

We wanted to gather statistics from Spotify and Twitter and compare monthly listeners and streaming data to tweet engagement and social media presence.

2. Achieved Goals:

APIs Used:

- Ticketmaster, Spotify

Data Gathered:

- Popularity scores, followers and top genres of Spoify's top artists in 2023
- Features of top artists' top songs (Danceability, Energy, Acousticness, Liveness)
- Ticketmaster popularity scores for top musicians.

We were able to use the data we collected to plot and analyze artists' popularity compared to the features of their music.

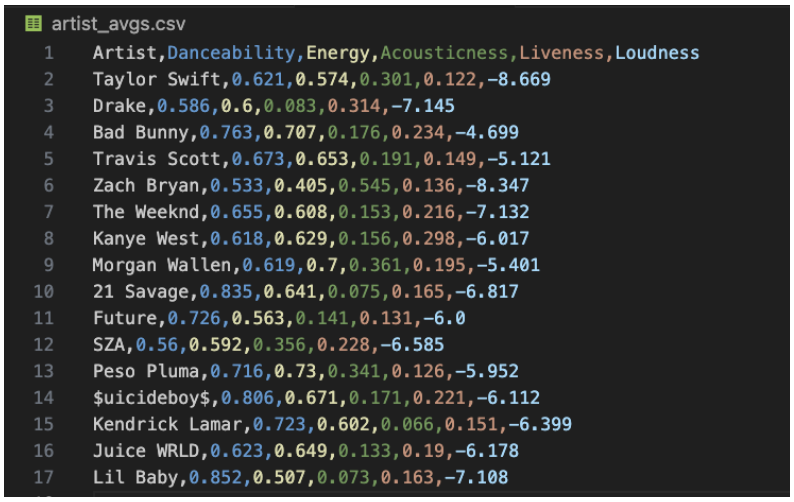
3. The problems that we faced:

- Learning and working with the various APIs
- Reaching the request limit while running the code

4. The calculations from the data in the database

Attached in project folder:

- artist_avgs.json
- artist_avgs.csv



artist_avgs.csv

```
Artist,Danceability,Energy,Acousticness,Liveness,Loudness
Taylor Swift,0.621,0.574,0.301,0.122,-8.669
Drake,0.586,0.6,0.083,0.314,-7.145
Bad Bunny,0.763,0.707,0.176,0.234,-4.699
Travis Scott,0.673,0.653,0.191,0.149,-5.121
Zach Bryan,0.533,0.405,0.545,0.136,-8.347
The Weeknd,0.655,0.608,0.153,0.216,-7.132
Kanye West,0.618,0.629,0.156,0.298,-6.017
Morgan Wallen,0.619,0.7,0.361,0.195,-5.401
21 Savage,0.835,0.641,0.075,0.165,-6.817
Future,0.726,0.563,0.141,0.131,-6.0
SZA,0.56,0.592,0.356,0.228,-6.585
Peso Pluma,0.716,0.73,0.341,0.126,-5.952
$uicideboy$,0.806,0.671,0.171,0.221,-6.112
Kendrick Lamar,0.723,0.602,0.066,0.151,-6.399
Juice WRLD,0.623,0.649,0.133,0.19,-6.178
Lil Baby,0.852,0.507,0.073,0.163,-7.108
```

5. The visualization that you created

Attached in project folder:

- average_loudness.png
- danceability_energy_relationship.png
- grouped_averages.png
- scatterPopularityComparison.png

6. Instructions for running your code

1. Make sure all necessary libraries are installed by running:
   pip install sqlite3
   pip install ticketpy
   pip install spotipy

2. Run the python file "getTicketmasterData.py" 4 times
   - This will add 100 artists' Ticketmaster data to the database

3. Run the python file "getSpotifyData.py" 8 times
   - This will add 16 artists data to the database in two tables:
   - The first table containing information on each of the artists
   - The second table contains information and data on each of the artists top 10 most streamed songs

4. Run the python file "analyzeAndVisualize.py" once
   - This will calculate averages for the song data and provide 4 visualizations analyzing the data stored in the database

7. Function Documentation

1. getTicketmasteData.py

get_data_ticketmaster():
- Fetches artist data from Ticketmaster API
- Inputs: None
- Functionality:
    - Initializes Ticketmaster API client
    - Defines list of artists
    - Randomly shuffles artist list
    - Loops through each artist
        - Makes API request to get artist data
        - Inserts artist data into SQLite database table
    - Tracks number of unique artists added
    - Commits data and closes database connection after 25 artists
- Outputs:
    - Populates "artists_tm" SQLite table with artist data

2. getSpotifyData.py

get_data_spotify():
- Fetches artist and track data from Spotify API
- Inputs: None
- Functionality:
    - Initializes Spotify API client
    - Gets top artists for the year
    - Loops through up to 16 artists:
        - Extracts artist info
        - Checks if artist already processed
        - Gets audio features for top 10 tracks
        - Inserts artist and track data into SQLite tables for 2 of the artists
- Outputs
    - Populates "artists_spotify" and "tracks" SQLite tables

strip_alphabet_chars():
- Helper function to strip alphabetic chars from artist ID
- Inputs:
    - input_string: artist ID string containing both letters and numbers
- Functionality:
    - Strips alphabetic characters
    - Returns integer form of artist ID to use as table key

- Outputs:
    - Unique integer version of input artist ID string

3. analyzeAndVisualize.py

average_audio_features():
- Analyzes audio features data for tracks
- Inputs: None
- Functionality:
    - Queries track audio features data
    - Computes average danceability, energy, acousticness, liveness, and loudness for each artist's top 10 tracks
    - Exports analysis to CSV and JSON
- Outputs:
    - artist_avgs: Dictionary with average audio features by artist
    - artist_avgs.csv: CSV export of averages
    - artist_avgs.json: JSON export of averages

spotify_tm_popularity_scatter():
- Creates scatter plot comparing Spotify and Ticketmaster popularity metrics
- Inputs: None
- Functionality:
    - Joins Ticketmaster and Spotify artist data
    - Plots Spotify popularity vs. Ticketmaster popularity
    - Assigns unique color and legend entry for each artist
- Outputs:
    - scatterPopularityComparison.png: scatter plot visualization file

plot_grouped_bar():
- Visualizes audio features analysis through grouped bar chart
- Inputs:
    - artist_avgs: Dictionary output from average_audio_features()
- Functionality
    - Plots a grouped bar chart with average danceability, energy, acousticness, liveness, and loudness per artist
- Outputs:
    - grouped_averages.png: grouped bar chart visualization file

plot_loudness_ranges():
- Visualizes average loudness per artist through bar chart

- Inputs:
    - artist_avgs: Dictionary output from average_audio_features()
- Functionality:
    - Extracts average loudness per artist
    - Plots average loudness values for each artist as bars
- Outputs:
    - average_loudness.png: bar chart visualization

danceability_energy_relationship():
- Scatter plot showing danceability vs. energy by artist
- Inputs:
    - artist_avgs: Dictionary output from average_audio_features()
- Functionality
    - Plots each artist's danceability against energy
    - Artist-specific color coding
- Outputs:
    - danceability_energy_relationship.png: scatter plot visualization

8. Resource Documentation

Date	Issue Description	Location of Resource	Result (did it solve the issue?
Dec 7	Learn Spotify API	https://developer.spotify.com/documentation/web-api	Yes
Dec 7	Learn TicketMaster API	https://developer.ticketmaster.com/products-and-docs/apis/getting-started/	Yes
Dec 7	Debugging throughout project	ChatGPT	Yes
Dec 7	Solving issues with APIs	Piazza	Yes
Dec 8	Troubleshooting with creating visualizations	Piazza	Yes
Dec 8	SQL Help	ChatGPT	Yes
Dec 12	Help on Report	Piazza	Yes