**Report on the Project: Creating a structured dataset from 'musicsuggestions' Subreddit Comments: An Analysis of User Intents and Preferences**

"For the complete dataset and code used in this project, please visit [[Git-Repository]](https://github.com/tobiaswollendorfer/musicsuggestion_subreddit_dataset.git)."

**Introduction:**

The aim of this project was to create a dataset based on posts from the "musicsuggestions" subreddit. Reddit is an online network of communities, known as subreddits, each focusing on a specific topic or interest. Within these subreddits, members can submit posts, comment on them, and express their preferences through 'upvotes' for content they appreciate and 'downvotes' for content they do not. The visibility of posts and comments is determined by Reddit's rating system, which calculates visibility based on the net score (upvotes minus downvotes) of each post or comment. This dataset will be analyzed in the future to understand the intentions behind why people listen to music. As the name suggests, the "musicsuggestions" subreddit serves as a platform where users ask for music recommendations. A typical post might be: "I need music while being sad," where other users suggest songs in the comments that they listen to when feeling sad. These comments may include song names or links to Spotify songs/playlists and YouTube videos.

By using Reddit's public API, the comments along with their ratings were extracted. The goal is to gain a good understanding of how music recommendations are exchanged within this community and what musical preferences or moods drive the discussions.

**Data Collection Process:**

The data collection process for this project was multi-staged and utilized various techniques and tools to gather and process data from the "musicsuggestions" subreddit. The core tools and methods described below include the use of the PRAW library in Python, temporary storage of data in JSON files, and the application of time delays to comply with Reddit API limitations.

**Use of PRAW**

The Python Reddit API Wrapper (PRAW) library was employed to facilitate access to and extraction of data from Reddit. PRAW allows for efficient interaction with Reddit's API to systematically collect posts, comments, and their ratings. For more information on PRAW, refer to official documentation at [PRAW Documentation](https://praw.readthedocs.io/en/stable/index.html).  
Details on Reddits´s API and its capabilities can be found at [Reddit API Documentation](https://www.reddit.com/dev/api/).

**Temporary Storage in JSON Files**

To enable efficient processing and analysis of the collected data, all extracted information was temporarily stored in JSON files. This step not only ensured structured data storage but also facilitated further work with the data, especially in terms of analysis and comparison with other datasets.

**Compliance with Reddit API Guidelines**

To adhere to Reddit API guidelines and prevent overload, time.sleep() was integrated into the data collection script. This function inserted a pause between requests to limit the number of requests per minute, thus respecting the API restrictions.

**Data Generation Process**

The data generation process involved several steps:

**Storing the Posts**: Initially, all posts, including ratings (upvotes, downvotes, and rating, where the rating was calculated as the difference between upvotes and downvotes), and post-ID were stored.

**Extraction of Top Comments**: The post-ID was used to identify the top 10 comments for each post. These comments, along with their ratings, were stored in a DataFrame.

**Processing Attempts**:

**Attempt 1**: Initially, an effort was made to filter comments that might contain songs by searching for patterns such as "-", "~", or "by". Although this method identified some relevant comments, further processing and accurate matching with actual songs proved problematic. We explored using Spacy Matcher, a tool from the Spacy Natural Language Processing (NLP) library designed to identify and match sequences of tokens based on patterns in text. This approach aimed to create patterns for all song/artist names to filter relevant comments more precisely. However, this method was discontinued due to the significant time investment required to generate comprehensive patterns for each song and artist name.

**Attempt 2**: This approach concentrated on extracting all Spotify and YouTube links, which were the most commonly shared types of links within the ‘musicsuggestions’ subreddit discussions. Comments with no links where dismissed. For Spotify links, this method enabled direct access to detailed information about the song, artist, or playlist through the Spotify API. When encountering a playlist link, the first 50 songs (due to API restrictions) and, if available, the playlist's description were included in the dataset entry. As a final step, the data were cross-referenced with the LFM2b dataset and the Music4All dataset to match the Spotify IDs.

YouTube links, on the other hand, were analyzed separately and compared with the Music4All dataset, with the findings stored in a distinct DataFrame.

**Results**:

The final data collection comprised two main datasets, stored as dataset\_spotify.json and dataset\_youtube.json. These files contain comprehensive information about music recommendations and discussions from the "musicsuggestions" subreddit, ready for detailed analysis and comparison with other music-related datasets.

**Technical Implementation:**

**Libraries and Frameworks:** Several specialized Python libraries and frameworks were employed to execute the project:

* **PRAW (Python Reddit API Wrapper)**: Enabled interaction with the Reddit API for data collection [PRAW Documentation](https://praw.readthedocs.io/en/stable/index.html)
  + Version used: 7.1.4
* **pandas**: Used for data management and analysis, particularly for processing and storing data in DataFrames. [Pandas Documentation](https://pandas.pydata.org/docs/)
  + Version used: 2.1.3

* **numPy**: Utilized for creating and handling arrays during the initial data storage phase. [Numpy Documentation](https://numpy.org/doc/)
  + Version used: 1.24.3
* **json**: Allowed for data storage in JSON format for later use and analysis. [Json Documentation](https://docs.python.org/3/library/json.html)
* **time**: Supported the implementation of waiting times between API requests to respect rate limits. [Time Documentation](https://docs.python.org/3/library/time.html)
* **requests**: Used to interact with the Spotify API and retrieve data from it. [Request Documentation](https://docs.python-requests.org/en/v2.0.0/)
  + Version used: 2.31.0
* **re (Regular Expressions)**: Enabled the processing and filtering of text data. [Regular Expressions Documentation](https://docs.python.org/3/library/re.html)
* **Spotify API**: Provided access to detailed information about songs and playlists. [Spotify API Documentation](https://spotipy.readthedocs.io/en/2.22.1/)

**Data Structure**

Data collection initially occurred in NumPy Arrays, which were then converted into Pandas DataFrames to enable efficient data manipulation and analysis. The final data storage was in JSON format.

**Data Extraction Process**

The data extraction followed a structured schema that allowed for a comprehensive collection of relevant information, including post titles, post upvotes, post downvotes, post score, Spotify IDs, song names, artists, comment upvotes, comment downvotes, comment score, and IDs, as well as playlist descriptions and IDs from the Music4all and LFM2b datasets.

**Handling API Limitations**

**Reddit API**: The limitations of the Reddit API were circumvented by implementing time.sleep() between requests, based on the limit of 100 queries per minute per OAuth client ID.

**Spotify API**: The rate limits of the Spotify API were addressed by evaluating the Retry-After response in the header. This allowed for a dynamic adjustment of the waiting time between requests to comply with API limits.

**Data Analysis**

The Spotify dataset consists of a total of 52,658 entries, reflecting a rich diversity of music preferences within the "musicsuggestions" subreddit.

**Dataset Details:**

* **Artists**: There are a total of 13,372 different artists in the dataset.
* **Songs**: 25,240 different songs were identified.
* **Categories (Unique post-name)**: The dataset consists of 390 unique post title that serves as categories to organize and facilitate the search for comments containing links to songs or playlists. These categories helped identify specific music recommendations within the discussion.
* **Comments**: Although the dataset contains 626 unique comments, this number may seem small relative to the total number of songs and artists identified. This discrepancy arises because the extraction process focused on comments containing direct links to Spotify playlists or songs, each Spotify playlist included up to 50 songs. From these 626 comments, a significant number of individual song entries were derived, illustrating the effectiveness of using direct links as a data source for music recommendations.

To provide a clearer overview of the dataset, the following table summarizes additional key statistics regarding the data collected.

|  |  |
| --- | --- |
| statistics | **Value** |
| Total number of categories (unique post titles) | 390 |
| Total number of comments | 626 |
| Mean number of comments per category | 1.6 |
| Mean number of music tracks extracted per category | 135.02 |
| Mean number of music tracks extracted per comment | 84.11 |
| Mean number of post score (upvotes-downvotes) | 11.77 |
| Mean number of comment score | 1.86 |
| Mean number of matches with Music4ALL | 0.18 |
| Mean number of matches with LFM2b | 0.51 |

To gain a deeper understanding of the motivations behind each post, I computed the cosine similarity between the post titles and 129 distinct music functions. After calculating all cosine similarities, I refined the selection by determining the average similarity score and disregarded any correlations below this average.

To visually comprehend the distribution, I analyzed the average similarity score across all music functions, where each row represents a single function. The resulting plot revealed varying degrees of correlation between music functions and post titles.

A green and white graph

Description automatically generated

The top five music functions, based on their average similarity score are:

1. "Because I would like to identify with a particular music scene."
   * average similarity score per occurrence: 0.24
   * occurrences: 366
   * classic example: “Music that feels like this picture? especially without words, just music” (similarity score: 0.39)
2. "Because I want to be informed about hits and trends."

* average similarity score per occurrence: 0.23
* occurrences: 342
* classic example: “Based on this, what should I listen to?” (similarity score: 0.37)

1. "Because I can recognize myself in the lyrics."

* average similarity score per occurrence: 0.23
* occurrences: 336
* classic example: “Songs that explore themes of self-identity, resilience, and the importance of staying true to oneself” (similarity score: 0.40)

1. "Because I want to know what’s going on in the music scene."

* average similarity score per occurrence: 0.22
* occurrences: 341
* classic example: “Looking to expand the types of music I listen to” (similarity score: 0.40)

1. "Because music just fits into my life."

* average similarity score per occurrence: 0.22
* occurrences: 333
* classic example: “Name a song that you attribute to a significant moment in your life” (similarity score: 0.43)

Upon manually examining the top 25 music functions, certain findings were particularly noteworthy, showcasing interesting reasons why people listen to music:

* "Because I can sing or hum along." (Rank: 10)
* "Because my friends like the same music as I do." (Rank: 12)
* "Because it puts me in the right mood for going out." (Rank: 17)
* "Because it is a good way to entertain myself." (Rank: 22)

The next step involved creating a visualization that offers an insightful perspective on the frequency with which various music functions achieve the highest similarity scores in relation to individual post titles. It’s important to acknowledge that not every function consistently reaches the top position. For instance, 'Music Function 1' has a count of 76, meaning that this music function has obtained the highest similarity score in 76 separate post titles—more than six times the count of 'Music Function 7,' which is at rank 7.

A screen shot of a graph

Description automatically generated

The visualization shows a marked decline in the frequency of music functions achieving the highest similarity scores after the top five, with a significant drop-off of nearly 50% between the fifth and sixth ranks. This observation prompted the decision to concentrate our attention on the top five music functions, which are as follows.

* 'Because I want to be informed about hits and trends.' - Count: 76
* 'Because I would like to identify with a particular music scene.' - Count: 73
* 'Because I can recognize myself in the lyrics.' - Count: 39
* 'Because it gives comfort to me when I'm sad.' - Count: 21
* 'Because when listening, I can imagine how the music would sound in a concert.' - Count: 21

Upon comparing these findings with the previously calculated average similarity scores, it's evident that three of the top five functions remain consistent across both analyses. However, two functions that emerged in the second analysis were not as prominent in the first. For instance, 'Because it gives comfort to me when I'm sad.' could be more specific than others, meaning it may have very high similarity scores with certain titles but also very low scores with others. While this specificity makes it highly relevant to certain discussions, it might not be broadly applicable across a wider range of titles, leading to a lower overall average score.

These insights suggest that certain music functions have stronger correlations with post titles than others, highlighting the diverse motivations behind music-related discussions. These range from seeking a connection with music scenes to finding songs that reflect personal experiences or enhance moods.

**Analysis of Matches with the LFM2b and Music4all Datasets:**

* **LFM2b Matches**: 26.962 entries (13.632 unique tracks) could be matched with LFM2b IDs, accounting for 51% of the total dataset.
* **Music4all Matches**: 9.802 entries (4.753 unique tracks) had non-null Music4All IDs, indicating a less extensive coverage compared to LFM2b.

Overall, 53.31% of the entries could be matched with either LFM2b or Music4all.

**Graphical Analysis: Artist-to-Song Ratios in comments**

In the graph below, we present the top 20 artists most frequently mentioned in comments. Among these, Nirvana, Radiohead, Neutral Milk Hotel, The Olivia Tremor Control, Nine Inch Nails, and Lansing-Dreiden are particularly notable, with each artist receiving up to double the number of songs mentions compared to others on the list. This pronounced skew may stem from the frequent sharing of artist-specific playlists. For instance, comments like A great band for rock newcomers is “Nirvana”; here's a to a playlist with songs from them: [Link]'

A graph of a number of songs

Description automatically generated

**Analysis of the YouTube Dataset:**

Matching with Music4ALL: Among the 554 entries in the YouTube dataset, each representing a "youtube" link, only 6 could be successfully matched with a Music4ALL YouTube ID from the Music4ALL dataset. Consequently, I divided the YouTube dataset into two distinct subsets: one containing the matches and the other without matches to the Music4ALL YouTube ID. The subset without matches comprises 548 entries.

The usefulness of these data will be seen in future processing of the datasets.

**Conclusion:**

With a total of over 50.000 entries and a wide variety of categories and songs, this dataset provides a solid foundation for various future research purposes. Especially for the planned analysis to understand the intentions behind why people listen to music, this dataset represents a valuable resource.