Sentiment-Driven Spotify Music Recommendation: Leveraging Social Media Posts and User Playlists for Personalized Music Experiences

Md Badiuzzaman Pranto

December 20, 2023

Contents

1 Introduction

The primary objective of this project is to build a song recommendation system based on the sentiment extracted from users' social media posts. The initial phase involved training a Logistic Regression model using Twitter posts, followed by an evaluation of its performance in sentiment detection from social media content. The model demonstrated a commendable 77% test accuracy. Subsequently, leveraging the trained model, the sentiment of individual users is detected. The recommendation system then integrates both the user's sentiment and their existing music playlist to provide personalized music suggestions. This dual consideration aims to enhance the relevance and emotional resonance of the recommended songs, contributing to a more tailored and engaging music recommendation experience.

1.1 Project goals

In the ever-evolving landscape of personalized music experiences, the project seeks to redefine the art of music curation by integrating the dynamic realm of social media sentiment analysis. The primary objective is to offer users a tailored music recommendation system that not only aligns with their individual tastes but also resonates with the emotional context conveyed through their social media posts.

Key Components:

1. Sentiment Analysis Model: Train a logistic regression model to discern sentiment from social media posts. By understanding the emotional nuances expressed in user-generated content, I aim to capture the mood and preferences that influence music choices.

2. User Playlist Integration: Leverage the sentiment scores obtained from the logistic regression model alongside the user's existing playlist data. By incorporating individual playlist preferences, our system strives to provide a holistic understanding of a user's musical inclinations.

3. Recommendation Engine: Develop a sophisticated recommendation engine that synthesizes sentiment analysis results and playlist data. The engine will dynamically adapt to users' changing emotions and preferences, ensuring that music suggestions are not only personalized but also responsive to the evolving sentiments expressed in their social media interactions.

In essence, the project work envisions a music recommendation system that transcends traditional genre-based approaches. By harnessing the power of sentiment analysis and user playlists, I aspire to create a deeply personalized and emotionally intelligent music streaming experience, revolutionizing the way users discover and connect with their favorite tunes.

1.2 Why logistic regression?

Explainability or interpretability in machine learning or deep learning models are very essential for building trust, ensuring ethical use, complying with regulations, improving model performance, and fostering collaboration between humans and AI systems. As machine learning applications continue to impact various aspects of society, the need for transparency and interpretability becomes increasingly critical. Over the last couple of years, the term *Explainable AI* has become one of the most concerning topic for researchers [1]. Many researches have been conducted to address why explaining the *black-box* models are important [1–3].

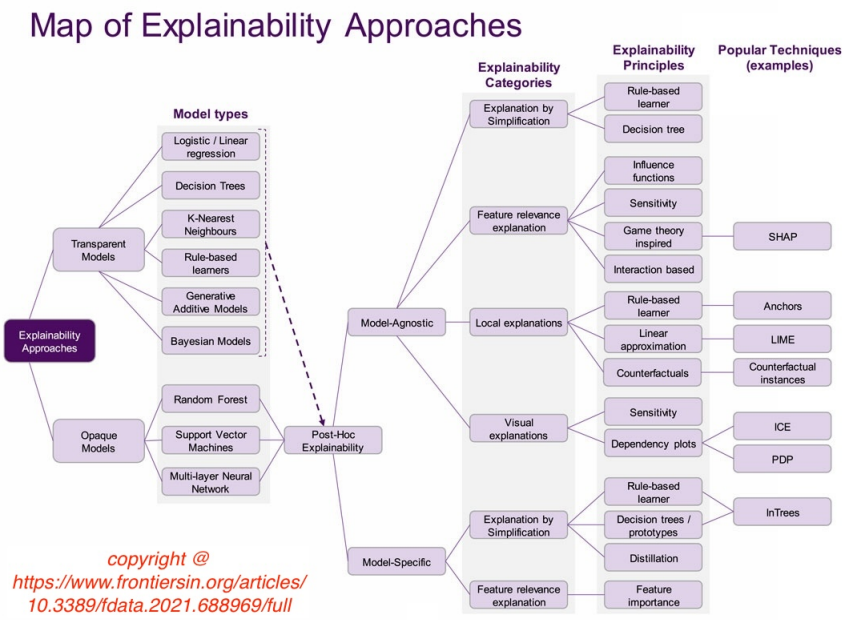

Figure 1: Map of explainability approaches.

The figure 1 above shows the explainablility approaches for different kind of models. Logistic Regression models are very simple, transparent and easy to explain. Though explaining the trained model does not fit in the scope of the project, but due to the simplicity and transparency of Logistic Regression, I started with such algorithm.

The subsequent sections of the report are structured as follows: Section 2 provides a detailed description and analysis of the datasets. Following that, Section 3 outlines the adopted methodology. Section 4 delves into the challenges encountered during the project. The results of the project are presented in Section 5, while Section 6 encompasses the discussion, remarks, and future works.

2 Data

For this project work, I required three datasets. One dataset to train a Logistic Regression model that detects sentiments from social media posts, one dataset that contains music information and one dataset that contains a user's playlist. The user's playlist dataset helps us to understand the user's taste in music. The section 2.1 below explains the acquisition and description of these datasets.

2.1 Data acquisition and description

As mentioned above, I have collected three datasets for this project work. The section 2.1.1 explains the social media posts, 2.1.2 explains the music data, and 2.1.3 describes user's playlist data respectively.

2.1.1 Social media post (Sentiment140)

The Sentiment140 dataset is a very popular open source dataset that contains *1.6 millions* twitter posts by several users. The dataset was originally collected by Alec Go and colleagues [4]. I have collected the dataset from kaggle.

Column number	Column name	Description
0	target	Polarity of the tweet (0 = negative, 4 = positive).
1	id	The id of the tweet.
2	date	The date of the tweet.
3	flag	The query. If there is no query, then this value is NO_QUERY.
4	user	The user that tweeted.
5	text	The text of the tweet.

Table 1: Summery of Sentiment140 dataset.

Column number	Column name	Description	Min	Max	Avg
0	track_name	Song name.	N/A	N/A	N/A
1	artist(s)_name	Artist(s) name.	N/A	N/A	N/A
2	artist_count	Number of artists.	1	8	1.568
3	released_year	Release year.	N/A	N/A	N/A
4	released_month	Release month.	N/A	N/A	N/A
5	released_day	Release day.	N/A	N/A	N/A
6	in_spotify_playlists	Song in number of Spotify playlist.	31	52898	4849.898
7	in_spotify_charts	Rank on Spotify charts.	0	147	11.722
8	streams	Number of streams on Spotify.	N/A	N/A	N/A
9	in_apple_playlists	Song in number of Apple playlist.	0	532	60.162
10	in_apple_charts	Song rank on Apple Music charts.	0	275	49.474
11	in_deezer_playlists	Song in number of Deezer playlist.	N/A	N/A	N/A
12	in_deezer_charts	Song rank on Deezer charts.	0	45	2.452
13	in_shazam_charts	Song rank on Shazam charts.	N/A	N/A	N/A
14	bpm	Beats per minute.	65	206	122.565
15	key	Key of the song.	N/A	N/A	N/A
16	mode	Mode of the song (major or minor).	N/A	N/A	N/A
17	danceability_%	How suitable the song is for dancing.	23	96	67.392
18	valence_%	Positivity of the song.	4	97	51.202
19	energy_%	Energy level of the song.	14	97	64.362
20	acousticness_%	Amount of acoustic sound.	0	97	26.310
21	instrumentalness_%	Amount of instrumental content.	0	91	1.677
22	liveness_%	Presence of live performance elements.	3	97	18.170
23	speechiness_%	Amount of spoken words in the song.	2	64	10.526

Table 2: Summery of Spotify songs dataset.

The table 1 shows the summery of this dataset and a short description of data source is provided below:

- Metadata URL: https://www.kaggle.com/datasets/kazanova/sentiment140
- Data URL: https://www.kaggle.com/datasets/kazanova/sentiment140
- Data Type: ZIP

2.1.2 Spotify songs

This is an open source dataset that contains a comprehensive list of the most famous songs as listed on Spotify. The dataset offers a wealth of features beyond what is typically available in similar datasets. The dataset has total *943* rows. It provides insights into each song's attributes, popularity, and presence on various music platforms. *In this project, I will refer bpm, danceability_%, energy_%, acousticness_%, instrumentalness_%, liveness_%, speechiness_% as audio features.*

The table 2 shows the summery of this dataset and a short description of data source is provided below:

- Metadata URL: https://www.kaggle.com/datasets/nelgiriyewithana/top-spotify-songs-2023
- Data URL: https://www.kaggle.com/datasets/nelgiriyewithana/top-spotify-songs-2023
- Data Type: ZIP

Column number	Column name	Description	Min	Max	Avg
0	track_name	Song name.	N/A	N/A	N/A
1	artist(s)_name	Artist(s) name.	N/A	N/A	N/A
2	artist_count	Number of artists.	1	4	1.549
3	released_year	Release year.	N/A	N/A	N/A
4	released_month	Release month.	N/A	N/A	N/A
5	released_day	Release day.	N/A	N/A	N/A
6	in_spotify_playlists	Song in number of Spotify playlist.	67	43257	5628.512
7	in_spotify_charts	Rank on Spotify charts.	0	55	9.756
8	streams	Number of streams on Spotify.	N/A	N/A	N/A
9	in_apple_playlists	Song in number of Apple playlist.	0	433	70.171
10	in_apple_charts	Song rank on Apple Music charts.	0	188	50.646
11	in_deezer_playlists	Song in number of Deezer playlist.	N/A	N/A	N/A
12	in_deezer_charts	Song rank on Deezer charts.	0	26	3.195
13	in_shazam_charts	Song rank on Shazam charts.	N/A	N/A	N/A
14	bpm	Beats per minute.	75	206	123.561
15	key	Key of the song.	N/A	N/A	N/A
16	mode	Mode of the song (major or minor).	N/A	N/A	N/A
17	danceability_%	How suitable the song is for dancing.	34	96	66.268
18	valence_%	Positivity of the song.	4	96	50.463
19	energy_%	Energy level of the song.	24	96	62.415
20	acousticness_%	Amount of acoustic sound.	0	84	27.683
21	instrumentalness_%	Amount of instrumental content.	0	14	0.317
22	liveness_%	Presence of live performance elements.	3	56	17.963
23	speechiness_%	Amount of spoken words in the song.	2	46	9.232

Table 3: Summery of user's playlist songs.

2.1.3 User's playlist

Finding a user's playlist that matches the same attributes as section 2.1.2. was a challenge. Unfortunately, I could not find a suitable user's playlist for this project. *But a user's playlist is always a subset of all available songs.* If all avaialble songs are the dataset in section 2.1.2, then user's playlist is a subset of this dataset. For this experiment, I have used 10% *(82 songs)* of random data from section 2.1.2. as user's playlist in order to understand user's preference on music. The playlist has been uploaded to a google drive. Columns are same as 2.1.2.

The table 3 shows the columns of this dataset and a short description of data source is provided below:

- Metadata URL: https://drive.google.com/file/d/1ABRo-uctp4EpTimyGx46BhJscy9MG6N5/view?usp=drive_link
- Data URL: https://drive.google.com/file/d/1ABRo-uctp4EpTimyGx46BhJscy9MG6N5/view?usp=drive_link
- Data Type: ZIP

3 Methodology

The methodology, as outlined in figure 2, has been followed while conducting this project work. The steps are as follows:

Understanding user's preference takes place in steps 7 to 10, which together make up the User's Songs Preference Algorithm. I have broken down this algorithm in detail in Section 3.4. This section explains the step-by-step process I used to figure out and cater to each person's specific music preferences.

The ETL-Pipeline used in this project work has been explained in section 3.1. In section 3.1.2, I have presented a python package dedicated to run ETL-pipelines followed by the data pre-processing in section 3.2. A short description of Logistic regression algorithm is described in section 3.3.

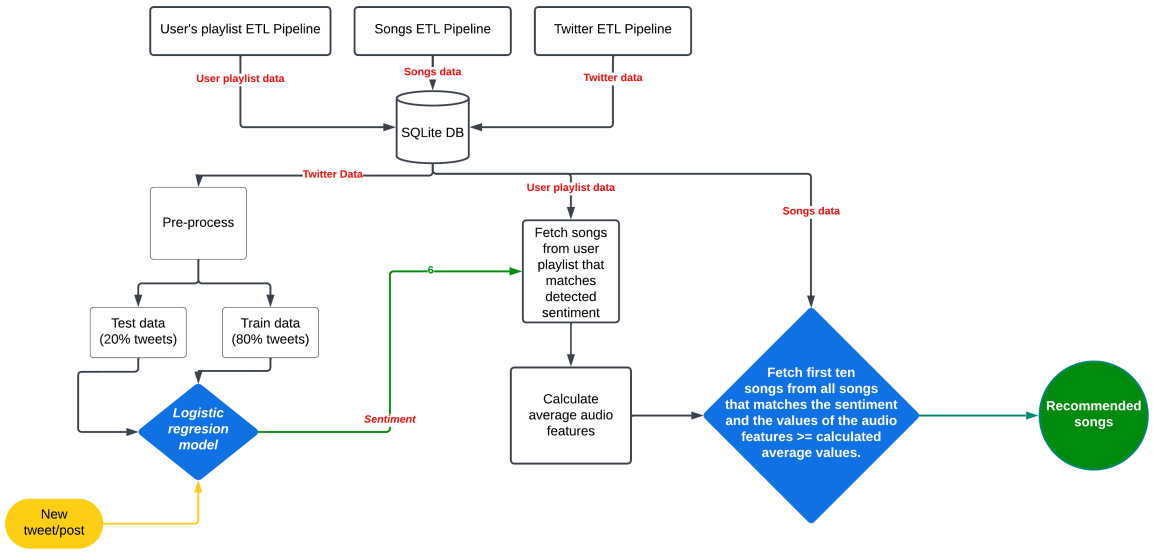

Figure 2: Methodology followed in this project.

Algorithm 1: Steps followed in this project's methodology.

1. Run three Extract Transform Load (ETL) pipelines.
2. Perform pre-processing on twitter data.
3. Split the twitter data into train and test set.
4. Train the Logistic regression model.
5. Evaluate the model.
6. Take new tweet/post as input.
7. Detect the sentiment of the user from the tweet.
8. Fetch user's playlist songs from the SQLite DB that matches the detected sentiment.
9. Calculate average of audio features (bpm, danceability, energy, acousticness, instrumentalness, liveness, and speechiness).
10. Fetch first ten songs from all songs that matches the sentiment and the values of audio features are >= average values calculated audio features.
11. Return the songs in descending order of valence as output.

3.1 Extract Transform Load (ETL) Pipeline

Given the project's need for three distinct datasets, three ETL pipelines were meticulously developed for data collection. Each pipeline initiates data extraction from its designated source, tailors transformations to meet individual requirements (outlined in Section 3.1.1), and culminates by storing the processed data in a SQLite database for subsequent utilization.

The figure 3 visualizes the structure of ETL pipelines used in this project work. An ETL Queue has been used to run the pipelines sequentially. The item number in the figure shows the execution sequence of the pipelines in the queue.

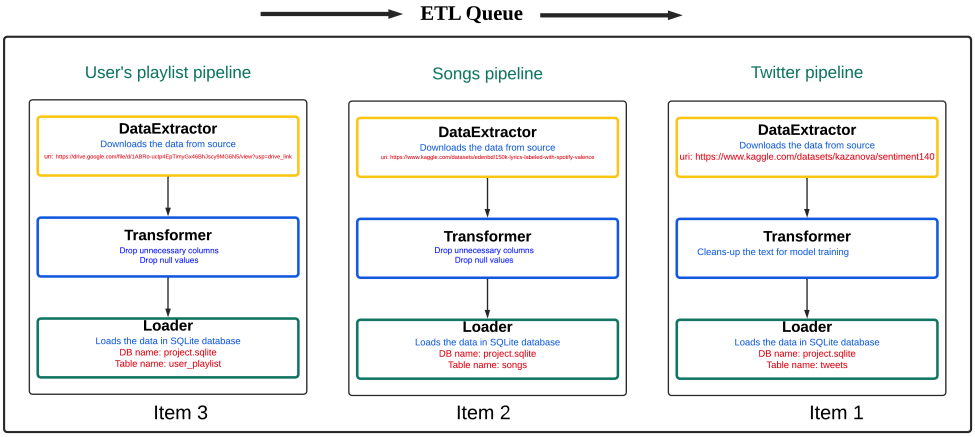

Figure 3: Extract Transform Load (ETL) Pipeline used in this project work.

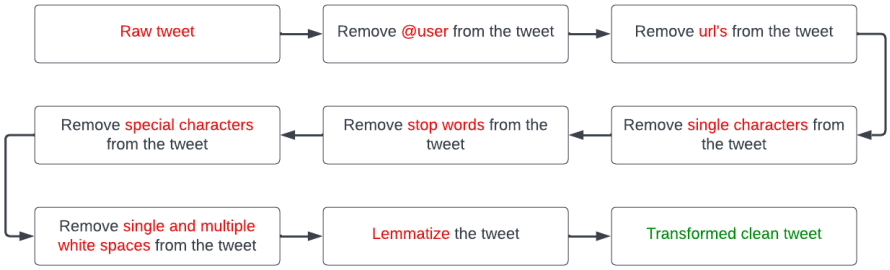

Figure 4: Transformation in ETL pipeline for Tweeter dataset.

3.1.1 Data transformation in the ETL pipeline

The Twitter dataset served as the foundation for model training; however, it posed challenges due to the presence of special characters, @user tags, URLs, single characters, and extraneous white spaces within the tweets. To prepare the dataset for effective model training, a comprehensive cleaning process was initiated, involving the removal of these artifacts. Additionally, the dataset underwent a refinement process that included the removal of stop words and lemmatization. The cleaning process is visually represented in Figure 4, illustrating the step-by-step execution for each tweet. This cleaning procedure was integral to the transformation step of the Twitter ETL pipeline. Conversely, for the other two pipelines, the transformation was limited to the removal of null values, and dropping unnecessary columns with no further data alterations applied. There was no null values in twitter data.

3.1.2 The Python package for ETL pipeline

Inspired by Jayvee [5], we collaboratively developed an open-source Python package [6] throughout this coursework [7]. My colleague, Arni Islam [8], and I engaged in distinct projects within this course, each with unique objectives. Consequently, our respective ETL pipelines exhibited different structures. In my project, the challenge involved extracting data from Kaggle archives and Google Drive, applying transformations outlined in Figures 6 and 7, and subsequently loading it into the database. Conversely, Arni's project focused on extracting data from a source providing direct CSV files, accompanied by her specific transformations. Recognizing these fundamental requirements for any data science project, we systematically structured our code to be generic, leading to the inception of this reusable Python package.

Given the widespread popularity of Python in the realms of data science and machine learning [9–11], a multitude of resources have emerged to support engineers in these domains. We firmly believe that the package offers a compelling blend of simplicity, flexibility, reliability, and community support, positioning it as a valuable tool for data engineers and analysts engaged in ETL workflows. Its user-friendly design, coupled with Python-powered customization capabilities, is poised to deliver a seamless experience for users seeking to streamline their data

processing tasks. The provided code snippet serves as an illustrative example of pipeline creation using etl-pipeline-runner.

```
## Importing services
from etl_pipeline_runner.services import (
    ETLPipeline,
    DataExtractor,
    CSVHandler,
    SQLiteLoader,
    ETLQueue,
)

## Configuring SQLite Loader
songs_loader = SQLiteLoader(
    db_name="project.sqlite",
    table_name="songs",
    if_exists=SQLiteLoader.REPLACE,
    index=False,
    method=None,
    output_directory=DATA_DIRECTORY,
)

## Defining Transformation
def transform_songs(data_frame: pd.DataFrame):
    data_frame = data_frame.dropna(axis=0)
    return data_frame

## Configuring CSV Handler
songs_csv_handler = CSVHandler(
    file_name="spotify-2023.csv",
    sep=",",
    names=None,
    transformer=transform_songs,
    loader=songs_loader,
    encoding="latin-1",
)

## Configuring Extractor
songs_data_extractor = DataExtractor(
    data_name="Spotify-songs",
    url="https://www.kaggle.com/datasets/nelgiriyewithana/top-spotify-songs-2023",
    type=DataExtractor.KAGGLE_ARCHIVE,
    file_handlers=(songs_csv_handler,),
)

## Configuring ETL Pipeline
songs_pipeline = ETLPipeline(
    extractor=songs_data_extractor,
)

ETLQueue(etl_pipelines=(songs_pipeline,)).run()
```

Note: If you are trying to extract data from kaggle, you will be requiring credentials. There is a section in the package documentation [6] on how to setup kaggle credentials.

Features as of version 1.1.0:

1. Data Extraction Capabilities:
    - Extract data from Kaggle (Contributed by me).
    - Extract data directly from sources providing CSV files (Contributed by Arni).
    - Handle CSV files seamlessly (Joint contribution).
2. Data Transformation and Loading:
    - Perform project-specific transformations with flexibility (Joint contribution).
    - Load data efficiently into SQLite databases (Joint contribution).
3. Pipeline Management:
    - Run multiple pipelines in a queue for streamlined execution (Joint contribution).

Possible features for future:

1. Expand Data Source Compatibility:
    - Enhance data extraction capabilities by allowing archives to be extracted from sources other than Kaggle.
2. Database Interaction:
    - Enable data extraction directly from databases, broadening the scope of data sources.
3. File Format Compatibility:
    - Handle XL/XLS files to accommodate a wider range of data formats.
4. Database Flexibility:
    - Extend data loading capabilities to support other types of databases.
5. Parallel execution:
    - Execute multiple pipelines concurrently.

You are warmly invited to submit feature requests, actively contribute to the package's development, and collectively address the evolving needs of the data science community.

3.2 Data pre-processing

The success of sentiment analysis relies heavily on effective text preprocessing, transforming raw textual data into a format suitable for machine learning algorithms. In this project work, Natural Language Toolkit (nltk)'s TfidfVectorizer was employed to preprocess the Twitter posts before training the logistic regression model.

3.2.1 The processing steps

1. Tokenization: The raw text data, consisting of Twitter posts, was tokenized into individual words or phrases using nltk's tokenization capabilities. This step is crucial for breaking down the text into its basic components, allowing for further analysis.
2. Stopword removal: Common words that do not contribute significantly to the sentiment of the text, known as stopwords, were removed from the tokenized data. This helps reduce noise in the dataset and focuses the model on more meaningful words.
3. Lemmatization: Lemmatization, the process of reducing words to their base or root form, was applied to ensure that variations of words (e.g., "running" and "ran") were treated as the same feature. This promotes a more consistent and meaningful representation of the text.
4. Tfidf rransformation: To convert the tokenized and preprocessed text into a numerical format suitable for machine learning models, the TfidfVectorizer from nltk was utilized. Tfidf (Term Frequency-Inverse Document Frequency) considers the importance of each term not only within a specific document but also across the entire corpus. This transformation helps highlight words that are more discriminative and relevant to the sentiment of the Twitter posts.

The step 2 and 3 below has been performed in ETL pipeline transformation.

3.2.2 Feature extraction with TfidfVectorization

1. Term Frequency (TF) The TfidfVectorizer computes the Term Frequency (TF) for each term in the document, indicating how frequently a term appears in a specific Twitter post.

2. Inverse Document Frequency (IDF) The Inverse Document Frequency (IDF) is calculated to assess the significance of a term across the entire corpus. Rare terms that occur in only a few documents receive a higher IDF score.

3. Tfidf representation The combination of TF and IDF produces the final Tfidf representation for each term in the dataset. This vectorization process results in a numerical matrix that captures the importance of each term in each Twitter post.

3.3 Logistic regression model

The logistic regression model transforms the linear combination of input features into a probability using the logistic function (sigmoid function). For a binary classification task, the logistic regression equation can be expressed as follows:

$$P(Y = 1) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_n X_n)}}$$

Here,

- $P(Y = 1)$ is the probability of the event Y occurring (e.g., positive sentiment),
- β_0 is the intercept term,
- $\beta_1, \beta_2, ..., \beta_n$ are the coefficients corresponding to the input features $X_1, X_2, ..., X_n$,
- e is the base of the natural logarithm.

3.4 Understanding user's preference in music

To enhance the personalization of music recommendations, this project work focuses on a comprehensive analysis of the user's taste by considering key audio features in their current playlist. The algorithm developed for this purpose intricately examines the average values of bpm, danceability, energy, acousticness, instrumentalness, liveness, and speechiness, providing a nuanced understanding of the musical elements that resonate with the user.

3.4.1 Algorithmic workflow

1. Sentiment Analysis: The initial step involves determining the user's sentiment through the trained model discussed in section 3.3. This sentiment analysis lays the groundwork for tailoring music recommendations to match the user's emotional context.

2. Playlist Filtering: Songs from the user's playlist that align with the predicted sentiment are extracted. This ensures that the subsequent analysis focuses exclusively on the subset of songs associated with the user's prevailing sentiment.

3. Calculation of Audio Feature Averages: The algorithm then computes the average values of essential audio features— bpm, danceability, energy, acousticness, instrumentalness, liveness, and speechiness—based on the songs selected from the user's playlist. This step provides a consolidated representation of the musical characteristics favored by the user.

4. Top Ten Song Selection: From the broader set of songs that match the user's sentiment, the algorithm identifies the top ten songs. This selection is refined further by considering songs with audio feature values greater than or equal to the previously calculated averages. This meticulous filtering ensures that the recommended songs not only align with the user's sentiment but also exhibit specific musical attributes reflective of their preferences.

5. Outcome and Significance: By adopting this feature-driven approach, the algorithm aims to transcend conventional music recommendation systems. Instead of solely relying on genre or popularity, the emphasis is placed on the nuanced analysis of audio features that contribute to the user's musical experience. This approach offers a more refined and personalized music recommendation, aligning closely with the user's sentiment and preferences as captured by the intricate interplay of bpm, danceability, energy, and other key features.

In summary, this algorithmic workflow represents a step forward in understanding and accommodating user-specific music preferences. By leveraging sentiment analysis and audio feature analysis, the system endeavors to offer a more immersive and tailored musical journey for users, contributing to a richer and more personalized music streaming experience.

4 Challenges

The pursuit of suitable datasets aligned with project requirements posed a significant challenge, necessitating several changes during the initial exploration. After multiple revisions, datasets that met the project's criteria were ultimately identified.

A notable challenge in data collection revolved around the unavailability of user playlists that matched the project's requirements. To address this, a workaround involved utilizing a subset (10%) of all songs, manually uploaded to Google Drive for extraction by the ETL pipeline.

Further complications emerged during data extraction from Kaggle and Google Drive, both requiring authentication. Overcoming these challenges was crucial for accessing the required data.

An additional noteworthy hurdle was encountered in the inconsistency of metadata. For instance, the Twitter dataset's metadata indicated three sentiment labels, yet only two were found in the actual data. This incongruity introduced complexities in accurately interpreting and utilizing the dataset.

Execution time presented a substantial challenge, particularly with the Twitter dataset containing 1.6 million entries. The time-intensive cleanup process required for such a vast dataset underscored the need for optimization in pipeline execution to maintain efficiency.

5 Results

6 Discussion

References

[1] P. Linardatos, V. Papastefanopoulos, and S. Kotsiantis, "Explainable ai: A review of machine learning interpretability methods," *Entropy*, vol. 23, no. 1, 2021. [Online]. Available: https://www.mdpi.com/1099-4300/23/1/18

[2] F. Xu, H. Uszkoreit, Y. Du, W. Fan, D. Zhao, and J. Zhu, "Explainable ai: A brief survey on history, research areas, approaches and challenges," in *Natural Language Processing and Chinese Computing*, J. Tang, M.-Y. Kan, D. Zhao, S. Li, and H. Zan, Eds. Cham: Springer International Publishing, 2019, pp. 563–574.

[3] A. Holzinger, A. Saranti, C. Molnar, P. Biecek, and W. Samek, *Explainable AI Methods - A Brief Overview*. Cham: Springer International Publishing, 2022, pp. 13–38. [Online]. Available: https://doi.org/10.1007/978-3-031-04083-2_2

[4] B. R. Go A. and H. L., "Twitter sentiment classification using distant supervision." *CS224N Project Report, Stanford*, p. 12, 2009.

[5] P. of Open Source Software at the University of Erlangen, "The jvalue projec." [Online]. Available: https://jvalue.github.io/jayvee/

[6] "etl-pipeline-runner." [Online]. Available: https://github.com/prantoamt/etl-pipeline-runner

[7] F.-A. U. E.-N. Professorship for Open-Source Software, "Advanced data engineering." [Online]. Available: https://oss.cs.fau.de/teaching/specific/made/

[8] "Arni islam." [Online]. Available: https://github.com/islam15-8789

[9] S. Raschka, J. Patterson, and C. Nolet, "Machine learning in python: Main developments and technology trends in data science, machine learning, and artificial intelligence," *Information*, vol. 11, no. 4, 2020. [Online]. Available: https://www.mdpi.com/2078-2489/11/4/193

[10] A. Nagpal and G. Gabrani, "Python for data analytics, scientific and technical applications," in *2019 Amity International Conference on Artificial Intelligence (AICAI)*, 2019, pp. 140–145.

[11] Z. Dobesova, "Programming language python for data processing," in *2011 International Conference on Electrical and Control Engineering*, 2011, pp. 4866–4869.