# Spotify health analysis

## Problem Description :

**Problem Description:** The Spotify Health Analysis project aims to explore and analyze music data from the popular music streaming platform, Spotify. With millions of songs and diverse genres, Spotify offers a wealth of data that can provide valuable insights into music preferences, trends, and patterns. This analysis seeks to uncover interesting information about artists, genres, and the characteristics of popular

music, thereby providing music enthusiasts, researchers, and the music industry with valuable knowledge.

**Background Information**: Spotify is a leading digital music streaming platform that offers a vast library of songs from various artists and genres. As users listen to and interact with music on Spotify, the platform gathers a wealth of data about user preferences, such as the number of plays (popularity), danceability, energy, tempo, and more.

The Spotify Health Analysis project utilizes a dataset containing information about songs available on Spotify. This dataset includes attributes such as 'artist_name,' 'track_name,' 'popularity,' 'genre,' 'danceability,' 'energy,' 'tempo,' and more. The dataset serves as the foundation for exploring the world of music, understanding the popularity of different tracks, and analyzing the musical characteristics associated with various genres.

**Objective:** The primary objective of the Spotify Health Analysis project is to gain insights into the following aspects:

1. **Popularity Trends:** Discover the popularity distribution of songs and identify the most popular tracks and artists on Spotify.

2. **Genre Analysis:** Explore the diversity of music genres, analyze their popularity, and understand the characteristics that distinguish them from one another.
3. **Musical Attributes:** Examine the danceability, energy, tempo, and other musical attributes of songs to identify patterns and correlations.

**Data Description:** The dataset used for this analysis contains multiple attributes that provide a comprehensive view of the songs available on Spotify. Here are some of the key attributes:

- **'artist_name':** The name of the artist who created the song.
- **'track_name'**: The name of the song track.
- **'popularity':** A measure of the song's popularity on Spotify, ranging from 0 to 100.
- **'genre':** The genre classification of the song, indicating its musical style or category.
- **'danceability':** A measure of how suitable the song is for dancing, ranging from 0 to 1.
- **'energy':** A measure of the intensity and activity of the song, ranging from 0 to 1.
- **'tempo':** The tempo of the song in beats per minute (BPM).

**Expected Insights:** By analyzing the Spotify music data, the project aims to provide insights into the following:

- The most popular tracks and artists on Spotify.
- The distribution of music popularity across different genres.
- Musical characteristics that distinguish various genres from one another.
- Potential correlations between musical attributes and popularity.
- Trends and patterns in music preferences based on genre and other attributes.

Overall, the Spotify Health Analysis project aims to provide an engaging and informative exploration of the world of music using real-world Spotify data, offering valuable insights for music lovers and industry professionals alike.

# Possible Framework :

**1. Importing Libraries and Loading the Dataset**

- Import the necessary libraries like numpy, pandas, and matplotlib.pyplot.
- Load the dataset using pd.read_csv() into a pandas DataFrame (df).

**2. Initial Data Exploration**

- Print the columns of the dataset using print(df.columns) to understand the available attributes.
- Display the first few rows of the dataset using print(df.head()) to get an overview of the data.

**3. Basic Statistics and Information**

- Calculate the number of unique artists in the dataset using df['artist_name'].nunique().
- Calculate the average popularity of tracks using df['popularity'].mean().

**4. Identifying the Most Popular Track**

- Find the maximum popularity value using df['popularity'].max().
- Filter the dataset to get the track(s) with the maximum popularity value.
- Display the name of the most popular track and its artist using boolean indexing.

**5. Visualizing Popularity Distribution**

- Create a histogram using plt.hist(df['popularity'], bins=20) to visualize the distribution of popularity values.
- Label the axes and add a title to the histogram for better understanding.

**6. Analyzing Genres and Music Attributes**

- Calculate the average danceability and energy using df['danceability'].mean() and df['energy'].mean() respectively.
- Count the occurrences of each genre using df['genre'].value_counts().
- Plot a bar chart to visualize genre counts using plt.bar().

**7. Grouping and Analyzing Data**

- Group the data by 'genre' and calculate the mean danceability for each genre using df.groupby('genre')['danceability'].mean().
- Create a bar plot to visualize top genres based on danceability.

**8. Clustering Analysis**

- Prepare the features for clustering (e.g., 'danceability', 'energy', 'key') and scale them using StandardScaler.
- Apply KMeans clustering algorithm with KMeans(n_clusters=3, random_state=42).
- Assign cluster labels to the data using kmeans.labels_.
- Add the cluster labels as a new column in the DataFrame.

**9. Exploring Music Clusters**

- Analyze the unique genres in each cluster using data.groupby('cluster')['genre'].unique().
- Print the cluster number along with the genres it contains.

**10. Further Visualization**

- Create bar plots to visualize the top 10 most and least listened genres using plt.bar() and value_counts().
- Sort the data based on 'energy' and 'danceability' to identify top genres with the highest values for each attribute.
- Visualize the top 20 artists using a bar plot.
- Group data by genre and calculate the average tempo, instrumentalness, acousticness, speechiness, and loudness for each genre.
- Plot bar charts to visualize top genres by highest and lowest average values for each attribute.

**11. Summarizing Insights**

- Summarize the key insights gained from the analysis, including popular tracks, genre popularity, musical attributes, and clustering results.

**12. Conclusion**

- Conclude the Spotify Health Analysis project by summarizing the overall findings and potential implications for music enthusiasts and the music industry.

# Code Explanation :

*If this section is empty, the explanation is provided in the .ipynb file itself.

**Step 1: Importing Libraries and Loading the Dataset**

- We start by importing the necessary Python libraries, such as numpy, pandas, and matplotlib.pyplot. These libraries provide powerful tools for data manipulation, analysis, and visualization.
- Next, we load the dataset into a pandas DataFrame named df using the pd.read_csv() function. This DataFrame will hold our music data, and we can perform various operations on it.

**Step 2: Initial Data Exploration**

- The initial data exploration aims to get a quick glimpse of the dataset. We print the column names using print(df.columns) to see the available attributes in the data. This helps us understand what kind of information we have in the dataset.
- To get an overview of the data itself, we use print(df.head()). This displays the first few rows of the dataset, showing us what the data looks like.

**Step 3: Basic Statistics and Information**

- Here, we calculate basic statistics and information about the data. We find the number of unique artists in the dataset using df['artist_name'].nunique(). This gives us an idea of how many different artists are present in the dataset.
- Additionally, we calculate the average popularity of tracks using df['popularity'].mean(). This tells us the average popularity score across all tracks, indicating how popular the songs are on average.

**Step 4: Identifying the Most Popular Track**

- This step focuses on identifying the track with the highest popularity in the dataset. We use df['popularity'].max() to find the maximum popularity value in the 'popularity' column.
- Then, we filter the dataset to get the track(s) with the maximum popularity value using boolean indexing. This helps us identify the most popular track(s) in the dataset.

- Finally, we display the name of the most popular track and its artist using print(max_popularity_track).

**Step 5: Visualizing Popularity Distribution**

- Visualization is a powerful tool to understand data better. Here, we create a histogram using plt.hist(df['popularity'], bins=20). The histogram shows us the distribution of popularity values across the tracks.
- By labeling the axes and adding a title to the histogram, we make the visualization more informative.

**Step 6: Analyzing Genres and Music Attributes**

- We explore music genres and their attributes. First, we calculate the average danceability and energy of songs using df['danceability'].mean() and df['energy'].mean() respectively. This gives us an idea of how danceable and energetic the songs are on average.
- Then, we count the occurrences of each genre using df['genre'].value_counts(). This helps us understand the popularity and frequency of different music genres in the dataset.
- We use a bar chart to visualize the genre counts, making it easier to compare the popularity of different genres.

**Step 7: Grouping and Analyzing Data**

- Here, we group the data by 'genre' and calculate the mean danceability for each genre using df.groupby('genre')['danceability'].mean(). This allows us to compare the danceability of different genres.
- We create a bar plot to visualize the top genres based on danceability. This helps us identify which genres are known for being more danceable than others.

**Step 8: Clustering Analysis**

- Clustering is a technique used to group similar data points together. We prepare the features for clustering, such as 'danceability', 'energy', and 'key', and scale them using StandardScaler. Scaling ensures that each feature contributes equally to the clustering process.

- We apply the KMeans clustering algorithm with KMeans(n_clusters=3, random_state=42). KMeans aims to find three clusters (groups) of songs based on their danceability, energy, and key attributes.
- After fitting the KMeans model, we assign cluster labels to the data using kmeans.labels_. Each song is now associated with a specific cluster.

**Step 9: Exploring Music Clusters**

- With clustering done, we can now analyze the unique genres present in each cluster. We use data.groupby('cluster')['genre'].unique() to group songs by cluster and get the genres present in each cluster.
- This gives us an idea of which genres are grouped together based on their musical attributes.

**Step 10: Further Visualization**

- We continue with additional visualizations. This includes bar plots to visualize the top 10 most and least listened genres, sorting the data to identify top genres based on energy and danceability, and exploring top artists and their frequency in the dataset.

**Step 11: Summarizing Insights**

- In this step, we summarize the key insights gained from the analysis. These insights include popular tracks, genre popularity, musical attributes, and clustering results.

**Step 12: Conclusion**

- Finally, we conclude the Spotify Health Analysis project by summarizing the overall findings and potential implications for music enthusiasts and the music industry.

# Future Work :

**Step 1: Data Enrichment and Augmentation**

**Explanation:** To improve the analysis and gain more meaningful insights, consider enriching the dataset with additional information such as artist backgrounds, song lyrics, release dates, and user listening habits. Augment the dataset with data from other sources like Spotify's Web API to obtain more attributes.

**Step-by-Step Guide:**

- Explore Spotify's Web API documentation and register for an API key.
- Utilize Python libraries like spotipy to fetch additional data for each track and artist.
- Merge the acquired data with the existing dataset based on common identifiers like track names and artist names.

**Step 2: Sentiment Analysis on Song Lyrics**

**Explanation:** Conduct sentiment analysis on song lyrics to understand the emotions conveyed in songs. This can help discover correlations between emotional content and music attributes like energy and danceability.

**Step-by-Step Guide:**

- Obtain song lyrics using libraries like lyricsgenius or web scraping techniques.
- Use natural language processing (NLP) tools like nltk or TextBlob for sentiment analysis.
- Associate sentiment scores with each song in the dataset and analyze their relationship with other attributes.

**Step 3: User Segmentation and Music Recommendations**

**Explanation:** Implement user segmentation to group listeners with similar music preferences. Create personalized music recommendations based on cluster profiles.

**Step-by-Step Guide:**

- Collect user listening data and build user profiles based on genres, favorite artists, and listening history.

- Apply clustering algorithms, like KMeans or hierarchical clustering, to group users with similar preferences.
- Develop a recommendation system using collaborative filtering or content-based filtering to suggest songs or playlists to users based on their clusters.

**Step 4: Time Series Analysis**

**Explanation:** Explore temporal patterns in music popularity and attributes. Analyze trends over time and identify the influence of seasonal events or artist releases on music preferences.

**Step-by-Step Guide:**

- Extract the release date of each track from the dataset.
- Use time series analysis tools like pandas and statsmodels to analyze trends over time.
- Visualize temporal patterns using line plots or seasonal decomposition.

**Step 5: Genre Evolution Analysis**

**Explanation:** Investigate how music genres have evolved over time. Identify emerging genres and track their popularity and attributes throughout different decades.

**Step-by-Step Guide:**

- Group songs by decade based on release dates.
- Analyze genre distributions and average attributes for each decade.
- Visualize genre trends over time using line plots or stacked bar charts.

**Step 6: Music Genre Classification**

**Explanation:** Build a machine learning model to predict the genre of a song based on its musical attributes. Evaluate the model's accuracy and use it for genre labeling in the dataset.

**Step-by-Step Guide:**

- Prepare the dataset by encoding the genre labels into numerical values.
- Split the data into training and testing sets.

- Train a classification algorithm like Random Forest or Support Vector Machine on the training data.
- Evaluate the model's performance using metrics like accuracy, precision, recall, and F1-score.

**Step 7: Genre-specific Music Feature Comparison**

**Explanation:** Analyze music attributes within specific genres to identify characteristic patterns unique to each genre.

**Step-by-Step Guide:**

- Select a few prominent genres for analysis.
- Create separate datasets for each chosen genre.
- Compare average attribute values across genres using box plots or bar plots.

**Step 8: User Engagement Analysis**

**Explanation:** Measure user engagement with tracks and artists by analyzing play counts, skip rates, and listening durations.

**Step-by-Step Guide:**

- Obtain user listening behavior data (if available) from streaming platforms.
- Calculate play counts, skip rates, and average listening durations for tracks and artists.
- Perform user segmentation based on engagement levels.

**Step 9: Cross-platform Analysis**

**Explanation:** Extend the analysis to include data from other music streaming platforms like Apple Music, YouTube Music, or Amazon Music. Compare user preferences, genre distributions, and popularity across platforms.

**Step-by-Step Guide:**

- Obtain data from multiple platforms and merge them into a unified dataset.
- Analyze and visualize differences in user behavior and music preferences.

**Step 10: Interactive Data Visualization**

**Explanation:** Create interactive visualizations using libraries like Plotly or Bokeh to allow users to explore the dataset and discover insights on their own.

**Step-by-Step Guide:**

- Use Plotly or Bokeh to create interactive plots and dashboards.
- Add filters and controls to the visualizations for user interaction.
- Deploy the interactive dashboard using web frameworks like Flask or Dash.

**Conclusion:**

By implementing these future work steps, the Spotify Health Analysis project can be further enhanced, providing more comprehensive insights into music trends, user preferences, and genre evolution. These advancements can be valuable for music enthusiasts, artists, and the music industry as a whole.

# Concept Explanation :

**Algorithm: K-Means Clustering**

Ah, welcome to the world of music analysis! Today, we'll explore a funny and friendly concept called "K-Means Clustering." Imagine you're a DJ at a party, and you want to group the partygoers based on their dance moves. Well, K-Means Clustering is just like that – it helps us group similar things together based on certain characteristics!

**Concept Explanation:**

Imagine you have a bunch of party animals on the dance floor, and you want to create clusters of people who dance similarly. You take a look at their dance moves and notice a few common characteristics, like their energy level and danceability. These characteristics are like the features we use in K-Means Clustering!

Now, the "K" in K-Means represents the number of dance groups you want to form. It's like deciding how many clusters (dance groups) you want to create on the dance floor. For example, if you decide to have three dance groups, K = 3.

**Step 1: Dance Group Formation**

You randomly select three party animals as the dance group leaders (we call them centroids), and each of them stands at a different spot on the dance floor. Now, other party animals look at these leaders and join the dance group whose leader they feel closest to in terms of energy and danceability.

**Step 2: Party Animal Shuffle**

Next, you ask everyone to start dancing again, and each party animal looks at their current dance group leader. Some of them might feel closer to a different leader now based on their dance moves. So, they decide to switch dance groups and join the leader they feel closest to.

**Step 3: Party Animal Average Dance**

After the shuffle, each dance group leader looks at their new group members and calculates the average energy and danceability of their group. The leader moves to a

new position on the dance floor based on this average, like a dance group leader doing a new dance move!

**Step 4: Repeat Until Settled**

We repeat the shuffle and leader dance move steps multiple times until all the party animals decide not to switch dance groups anymore. Once they all settle down, the dance groups are formed!

**Step 5: The Final Dance Groups**

Now, we have our final dance groups! Each group consists of party animals who dance similarly, based on their energy and danceability. It's like we magically formed groups of friends who share the same dance vibes!

**Example:**

**Let's see how this works with some party animals and their dance moves:**

Party Animal A: Energy = 8, Danceability = 7 Party Animal B: Energy = 5, Danceability = 6 Party Animal C: Energy = 9, Danceability = 8 Party Animal D: Energy = 2, Danceability = 3 Party Animal E: Energy = 7, Danceability = 6

If we decide to form two dance groups (K = 2), we randomly choose two party animals as dance group leaders:

Leader 1: Party Animal A (Energy = 8, Danceability = 7) Leader 2: Party Animal D (Energy = 2, Danceability = 3)

Now, based on their energy and danceability, the other party animals join the dance groups:

Dance Group 1: A, B, C, E (Closer to Leader 1) Dance Group 2: D (Closer to Leader 2)

After some shuffling and leader dance moves, the dance groups might change:

Dance Group 1: A, B, C (Closer to Leader 1) Dance Group 2: D, E (Closer to Leader 2)

The process continues until all the party animals are happy with their dance groups and the dance groups' leaders settle at the average dance moves of their members.

And voilà! We have successfully used K-Means Clustering to form dance groups on the dance floor based on energy and danceability!

# Exercise Questions :

**1. What is the purpose of the Spotify Health Analysis project?**

**Answer:** The purpose of the Spotify Health Analysis project is to analyze music data to gain insights into music preferences, genre popularity, and the relationship between musical attributes like danceability and energy. It aims to provide meaningful information for music enthusiasts, artists, and the music industry.

**2. How can you determine the most popular track in the dataset?**

**Answer:** To determine the most popular track, we can calculate the track's popularity score using the 'popularity' column in the dataset. The track with the highest popularity score will be considered the most popular.

**3. What are the potential future work steps to enhance the project's analysis?**

**Answer:** Some future work steps include data enrichment and augmentation, sentiment analysis on song lyrics, user segmentation and music recommendations, time series analysis, and cross-platform analysis.

**4. How can you use K-Means Clustering to group songs based on their musical attributes?**

**Answer:** K-Means Clustering can group songs based on musical attributes by first defining the number of clusters (K) we want to create. Then, the algorithm iteratively assigns songs to clusters based on their similarity in musical attributes like danceability and energy.

**5. Can you explain the steps to build a music genre classification model?**

**Answer:** To build a music genre classification model, we first prepare the dataset by encoding genre labels. Then, we split the data into training and testing sets. Next, we train a classification algorithm like Random Forest or Support Vector Machine on the training data. Finally, we evaluate the model's performance using metrics like accuracy, precision, recall, and F1-score.

**6. How can you identify the top 10 most listened genres in the dataset?**

**Answer:** To identify the top 10 most listened genres, we can use the 'value_counts()' function on the 'genre' column to count the occurrences of each genre. Then, we select the top 10 genres based on their frequency.

**7. How can you analyze the correlation between danceability and energy across different music genres?**

**Answer:** To analyze the correlation between danceability and energy across different music genres, we can use a scatter plot with danceability on the x-axis and energy on the y-axis. We can then use color-coding to represent different genres and observe the patterns.

**8. What is the purpose of using sentiment analysis on song lyrics in the future work?**

**Answer:** Sentiment analysis on song lyrics helps understand the emotions conveyed in songs. It can be valuable in identifying correlations between emotional content and music attributes, providing insights into how different musical styles evoke specific feelings.

**9. How can you create interactive data visualizations for the Spotify Health Analysis project?**

**Answer:** We can create interactive data visualizations using libraries like Plotly or Bokeh. These libraries allow adding filters and controls to the visualizations, enabling users to explore the dataset and discover insights on their own.

**10. How can you analyze the evolution of music genres over time using the dataset?**

**Answer:** To analyze the evolution of music genres over time, we can group songs by decade based on their release dates. Then, we can compare genre distributions and average attributes for each decade. Visualizing genre trends over time using line plots or stacked bar charts will help observe changes in genre popularity throughout different decades.