<u>AN INCLUSIVE MUSIC RECOMMENDATION SYSTEM DATA SCIENCE PROJECT</u>

MORINGA DS-FT 14

GROUP IV PROJECT

PRESENTED BY:

NIKITA EDEL SAMBA

SUMAIYA OSMAN

RANIEL MUSYOKI

BEVERLYNE LANGAT

SIMIYU DAVID MASAI

Table of Contents

1. Business Understanding

Business Overview

This project aimed to analyze trends in musical data then using machine learning models to enhance recommendations.

Problem Statement

Music streaming platforms struggle to keep users engaged and help them discover new artists they might like. Traditional recommendation systems often focus on individual songs, but users may also want recommendations for similar artists and their music.

The goal was to build a robust and inclusive recommender system that accurately identified patterns in music attributes and ensure equitable exposure to all artists in our e-music platform while enhancing user recommendations.

Objectives

The objective of this project is divided into three key areas:

1. *Develop an Intelligent Artist Recommendation System*
   - Build a deep learning-based system that identifies and suggests similar artists based on *audio features, genre, and popularity*.
   - Utilize autoencoders to capture artist similarities effectively.
2. *Enhance User Engagement & Music Discovery*
   - Improve user experience by introducing them to new artists that align with their listening preferences.
   - Encourage exploration beyond mainstream artists by presenting diverse recommendations.
3. *Provide a Scalable and Efficient Recommendation Model*
   - Ensure the system is computationally efficient and scalable to handle large music databases.
   - Integrate the recommendation system into music streaming platforms via an API or user interface.

2. Data Understanding

Data Collection

The dataset used for this project is the **Spotify Tracks Dataset**, sourced from **Kaggle**.

It contains detailed information about Spotify tracks across 125 genres, along with associated audio features and popularity metrics.

Dataset Format: CSV (Tabular)

Key Features Include:

1.Track Metadata: Song title, album, artist(s), genre, duration.

2.Audio Features: Danceability, energy, loudness, tempo, and valence.

3.User Engagement: Popularity score based on plays and recency

Feature Description

- **track_id:**
  - The Spotify ID for the track
- **artists:**
  - The artists' names who performed the track. If there is more than one artist, they are separated by a ;
- **album_name:**
  - The album name in which the track appears
- **track_name:**
  - Name of the track
- **popularity:**
  - The popularity of a track is a value between 0 and 100, with 100 being the most popular.
  - The popularity is calculated by algorithm and is based, in the most part, on the total number of plays the track has had and how recent those plays are.

    - Generally speaking, songs that are being played a lot now will have a higher popularity than songs that were played a lot in the past.
    - Duplicate tracks (e.g. the same track from a single and an album) are rated independently.
    - Artist and album popularity is derived mathematically from track popularity.
- **duration_ms:**
- The track length in milliseconds
- **explicit:**
    - Whether or not the track has explicit lyrics (true = yes it does; false = no it does not OR unknown)
- **danceability:**
    - Danceability describes how suitable a track is for dancing based on a combination of musical elements including tempo, rhythm stability, beat strength, and overall regularity.
    - A value of 0.0 is least danceable and 1.0 is most danceable
- **energy:**
    - Energy is a measure from 0.0 to 1.0 and represents a perceptual measure of intensity and activity.
    - Typically, energetic tracks feel fast, loud, and noisy.
    - For example, death metal has high energy, while a Bach prelude scores low on the scale
- **key:**
    - The key the track is in.
    - Integers map to pitches using standard Pitch Class notation. E.g. 0 = C, 1 = C♯/D♭, 2 = D, and so on.
    - If no key was detected, the value is -1
- **loudness:**
    - The overall loudness of a track in decibels (dB)

    - A positive dB value indicates a signal is louder than the reference level.
    - A negative dB value indicates a signal is quieter than the reference level.
- **mode:**
    - mode feature refers to the modality of the track, which indicates whether the track is in a major or minor key.
    - Major mode (1) typically corresponds to more "happy," "bright," or "cheerful" sounds.
    - Minor mode (0) typically corresponds to more "sad," "dark," or "serious" sounds.
- **speechiness:**
    - Speechiness detects the presence of spoken words in a track.
    - The more exclusively speech-like the recording (e.g. talk show, audio book, poetry), the closer to 1.0 the attribute value.
    - Values above 0.66 describe tracks that are probably made entirely of spoken words.
    - Values between 0.33 and 0.66 describe tracks that may contain both music and speech, either in sections or layered, including such cases as rap music.
    - Values below 0.33 most likely represent music and other non-speech-like tracks
- **acousticness:**
    - A confidence measure from 0.0 to 1.0 of whether the track is acoustic.
    - 1.0 represents high confidence the track is acoustic
- **instrumentalness:**
    - Predicts whether a track contains no vocals.
    - "Ooh" and "aah" sounds are treated as instrumental in this context.
    - Rap or spoken word tracks are clearly "vocal".

    - The closer the instrumentalness value is to 1.0, the greater likelihood the track contains no vocal content
- **liveness:**
    - Detects the presence of an audience in the recording.
    - Higher liveness values represent an increased probability that the track was performed live.
    - A value above 0.8 provides strong likelihood that the track is live
- **valence:**
    - A measure from 0.0 to 1.0 describing the musical positiveness conveyed by a track.
    - Tracks with high valence sound more positive (e.g. happy, cheerful, euphoric), while tracks with low valence sound more negative (e.g. sad, depressed, angry)
- **tempo:**
- The overall estimated tempo of a track in beats per minute (BPM).
    - In musical terminology, tempo is the speed or pace of a given piece and derives directly from the average beat duration
- **time_signature:**
    - An estimated time signature.
    - The time signature (meter) is a notational convention to specify how many beats are in each bar (or measure).
    - The time signature ranges from 3 to 7 indicating time signatures of 3/4, to 7/4.
- **track_genre:**
    - The genre in which the track belongs

The dataset provides a structured foundation for building personalized, mood-based music recommendations.

Validation of Dataset

Why This Data is Useful

- The Spotify Tracks Dataset is ideal for solving Neural Beats' business problem because it includes rich audio and user interaction features that allow us to:

Build Personalized Recommendation Models

- Utilize audio features (e.g., danceability, energy, valence) to match user preferences.
- Move beyond simple genre-based recommendations by considering song mood and context.

Improve Music Discovery & Engagement

- Predict emerging trends by analyzing song popularity over time.
- Recommend undiscovered tracks based on listening behavior and similar song attributes.

Develop Context-Aware AI Models

- Suggest music based on user activity (e.g., workout, relaxation, focus).
- Use tempo, acousticness, and valence to enhance mood-based recommendations.

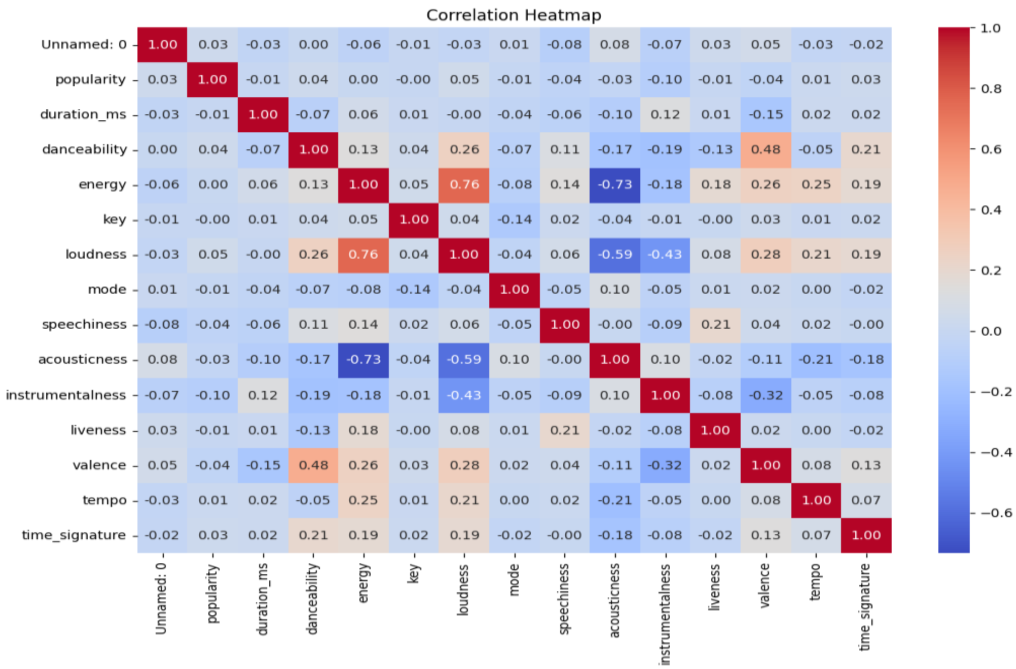
Exploratory Data Analysis

Initial analysis was performed to explore feature distributions and relationships between variables.

The dataset had 89741 records (rows) and 21 columns (fields), with datatypes below

```
 #   Column            Non-Null Count   Dtype
---  ------            --------------   -----
 0   Unnamed: 0        114000 non-null  int64
 1   track_id          114000 non-null  object
 2   artists           113999 non-null  object
 3   album_name        113999 non-null  object
 4   track_name        113999 non-null  object
 5   popularity        114000 non-null  int64
 6   duration_ms       114000 non-null  int64
 7   explicit          114000 non-null  bool
 8   danceability      114000 non-null  float64
 9   energy            114000 non-null  float64
 10  key               114000 non-null  int64
 11  loudness          114000 non-null  float64
 12  mode              114000 non-null  int64
 13  speechiness       114000 non-null  float64
 14  acousticness      114000 non-null  float64
 15  instrumentalness  114000 non-null  float64
 16  liveness          114000 non-null  float64
 17  valence           114000 non-null  float64
 18  tempo             114000 non-null  float64
 19  time_signature    114000 non-null  int64
 20  track_genre       114000 non-null  object
```

Below is a correlation heatmap of the numerical features from the dataset



Key Interpretations:

1. energy and loudness (0.76) → Strong positive correlation

   Louder songs tend to have higher energy.

2. acousticness and energy (-0.73) → Strong negative correlation

   Acoustic songs tend to have lower energy levels.

3. instrumentalness and loudness (-0.43) → Moderate negative correlation

   More instrumental tracks tend to be quieter.

4. danceability and valence (0.48) → Moderate positive correlation

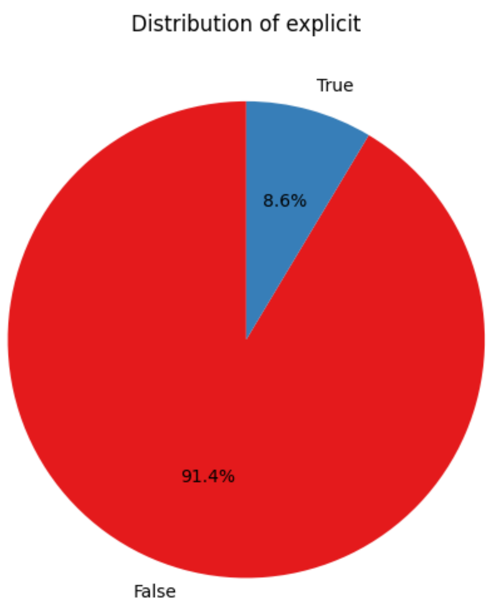
   Happier songs (high valence) are more danceable.

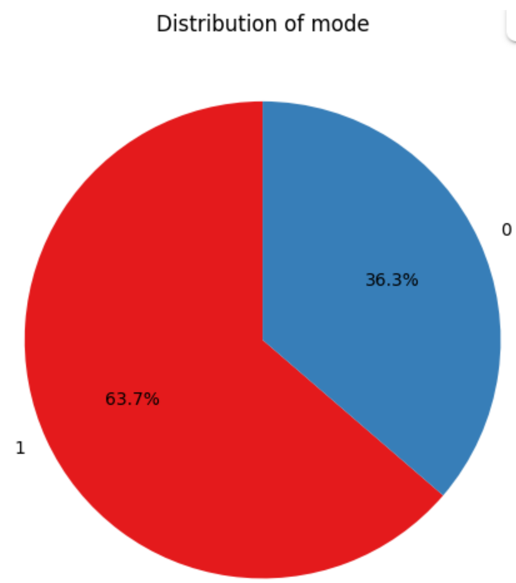
5. popularity and other features

   Popularity has weak correlations (close to 0) with most features, meaning song attributes like energy, danceability, or duration do not strongly determine a song’s popularity.

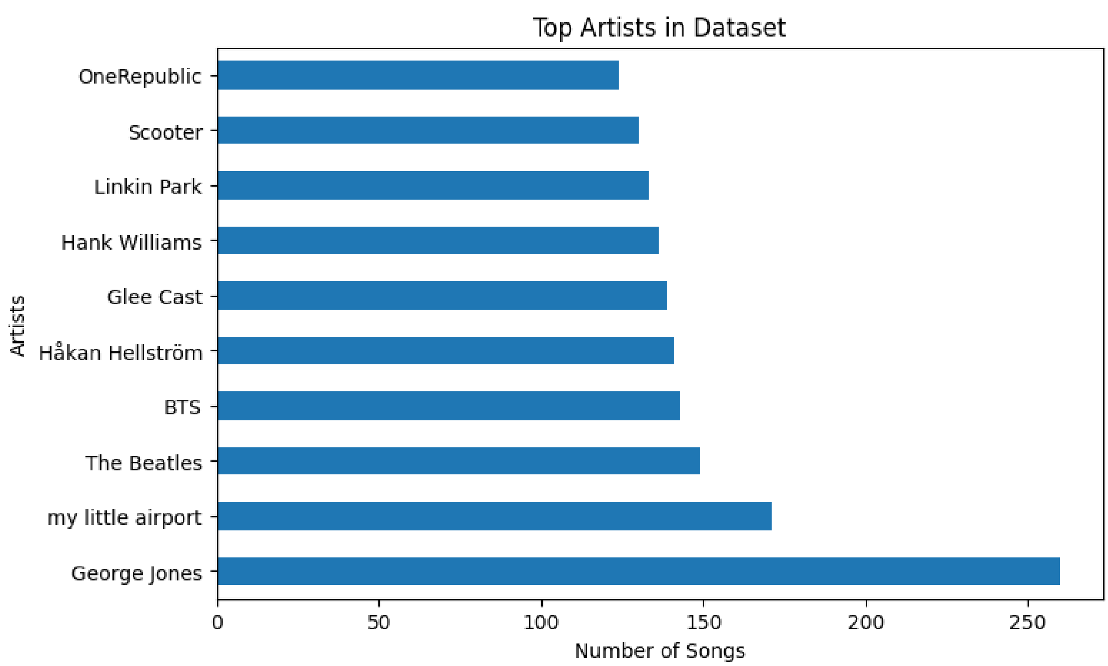
The dataset had **<u>113 unique Genres</u>**

Below is a piechart for the categorical columns showing the distributions of the various features

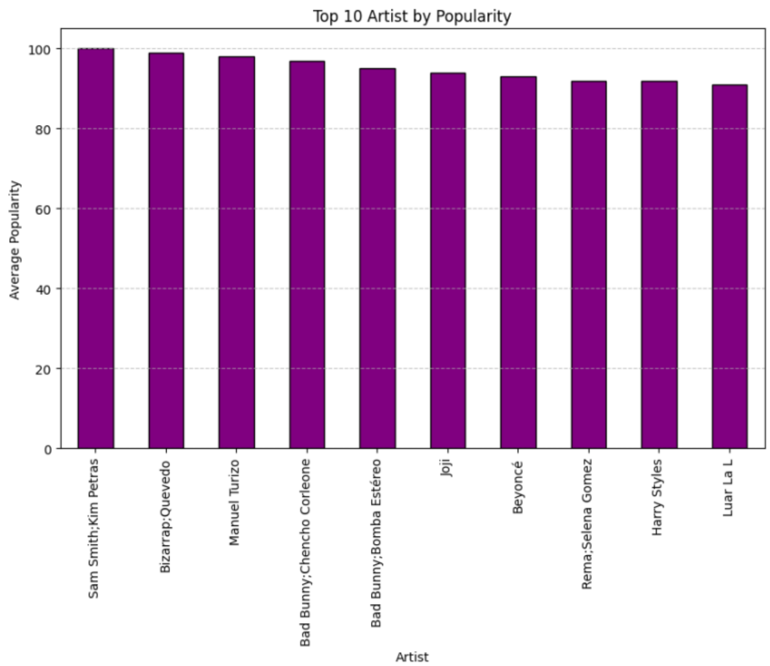




Artist with the most songs in the dataset was **George Jones**:



Top 10 Artists by song popularity



General Findings from Exploratory Data Analysis

1. Popularity & Artist Influence

- Our analysis revealed that **collaborations between artists** tend to have higher popularity scores (e.g., *Sam Smith & Kim Petras* and *Bizarrap & Quevedo*).
- Certain artists like **Bad Bunny** appear multiple times, indicating **strong dominance in the industry**.

2. Genre Diversity & Music Trends

- The dataset includes a **wide variety of genres**, showing that recommendations should consider more than just popularity.
- Some **less mainstream artists** still achieved high popularity, proving that emerging artists can attract large audiences.

3. Audio Features & Music Similarity

- Audio features like **danceability, energy, and tempo** showed strong correlations with genre and popularity.
- Songs within **similar audio feature clusters** are likely to be perceived as related, justifying the use of **Autoencoders** to extract meaningful latent features.

3. Data Preparation

1. Remove duplicates and missing values.
2. Standardize **track_name** and **artist_name** fields.
3. Column manipulation which involved:

- check column names to see if they are the same
- change the column names to lowercase
- rename column names to make them more undesrstandable
- remove whitespaces in the data and column names if any
- drop unnecessary column names

4. Engineer new features like **artist_exposure_score**, which measures how often an artist is recommended.
5. Normalize numerical features (e.g., popularity, tempo) to ensure fair weighting.

4. Modelling

In creating our recommender system, we followed the following steps.

Model Used: **Autoencoder** for finding **artist similarity**.

1. Train an autoencoder to learn a compressed representation (latent space) of artist features.
2. Extract artist embeddings from the latent space.
3. Compute pairwise similarities between artists.

```python
from sklearn.preprocessing import StandardScaler
# Select numerical features to represent artists
features = ["danceability", "energy", "valence", "tempo", "acousticness", "instrumentalness", "liveness"]

# Group by artist and compute average feature values
artist_features = df.groupby("artists")[features].mean()

# Normalize the features
scaler = StandardScaler()
artist_features_scaled = scaler.fit_transform(artist_features)

# Convert back to a DataFrame
artist_features_scaled_df = pd.DataFrame(artist_features_scaled, index=artist_features.index, columns=features)

# Display the first few rows
artist_features_scaled_df.head()
```

artists	danceability	energy	valence	tempo	acousticness	instrumentalness	liveness
!nvite	1.414172	-0.493969	-0.180573	-1.399798	0.580849	-0.512943	-0.493316
"Puppy Dog Pals" Cast	0.729730	0.887943	1.837894	0.586548	-0.685855	-0.547748	-0.364857
"Weird Al" Yankovic	0.447657	-0.264334	1.115283	-0.187631	-0.198669	-0.550025	-0.049885
#Kids;Nursery Rhymes;Nursery Rhymes and Kids Songs	-0.011008	-0.596485	1.809635	1.030864	-0.834213	2.020753	-0.591350

2. This was followed by a Deep Learning Model:

Using **deep learning embeddings** to find similar artists based on their song attributes. This involved using TensorFlow to create an artist-song feature matrix (average danceability, energy, valence, tempo, etc. per artist).

Model Evaluation : Mean Squared Error

1. *Measures Reconstruction Quality*

- Autoencoders compress input data into a lower-dimensional space and then reconstruct it.
- MSE quantifies how much information is lost in this process.
- A lower MSE means better reconstruction, meaning the Autoencoder captures important patterns.

2. *Sensitive to Large Errors*

- Since MSE squares the differences, larger reconstruction errors contribute more to the final score.
- This helps identify cases where the Autoencoder struggles to reconstruct specific data points.

3. *Smooth & Differentiable*

- MSE is a convex and differentiable function, making it easy to optimize using gradient descent.
- This helps in efficient training of the Autoencoder by minimizing reconstruction loss.

5. Results and Discussion

The Autoencoder effectively captured latent artist similarities, transforming high-dimensional features into compact embeddings.

With an MSE of 0.1198, the model demonstrated strong reconstruction ability, preserving essential artist relationships.

The model allows for efficient and scalable artist recommendations, suitable for integration into streaming services.

6. Recommendations

❖ **Improving Music Discovery**

- The model successfully groups artists with **similar audio characteristics**, promoting discovery beyond mainstream artists.
- Users will be introduced to **new artists aligned with their listening preferences**, increasing engagement.

❖ **Scalability & Efficiency**

- The low-dimensional embeddings from the **Autoencoder** make the system **faster and more scalable** than traditional similarity-based models.
- This approach is suitable for **large-scale music databases** and can be integrated into streaming platforms via an API.

❖ **Enhancing Recommendations with Additional Data**

- While the model captures **audio-based similarities**, incorporating **user listening behavior** could refine recommendations further.
- Future improvements could integrate **real-time user preferences** and **social listening patterns**.

7. Conclusion and Next Steps

- **Deploy & Monitor:** Implement the recommendation system and track user feedback.
- **Optimize Model Performance:** Experiment with different latent space dimensions to refine recommendations.
- **Enhance User Personalization:** Combine **content-based filtering** with **user interaction data** to improve artist discovery.