



# Moodify

Progetto di Intelligenza Artificiale

# Indice

- Introduzione
- Obiettivi del progetto
- Dataset e analisi preliminare
- Soluzione proposta
- Implementazione tecnica
- Risultati
- Conclusioni e sviluppi futuri
- Domande

# Introduzione

## Obiettivo del Progetto:

La musica ha un ruolo unico nel modellare le nostre emozioni e stati d'animo.

L'obiettivo di questo progetto è quello di creare un programma capace di generare una playlist per ogni stato d'animo.

## Perchè generare playlist basate sulle emozioni?

Questo progetto nasce dall'idea di utilizzare tecniche di intelligenza artificiale per analizzare le caratteristiche musicali di un brano, come energia, ballabilità e positività, al fine di associare ciascuna canzone a un determinato stato emotivo (o "mood").

I & Music

Il nostro obiettivo è creare un programma capace di generare una playlist per ogni stato d'animo.

Classifica



# Obiettivi Del Progetto

- **Analizzare le caratteristiche musicali:**  
Utilizzare parametri come energia, ballabilità e valenza emotiva per definire il mood di una canzone.
- **Classificare le canzoni in base al mood:**  
Generare playlist che riflettano lo stato d'animo scelto dall'utente.
- **Clustering basato su caratteristiche:**  
Applicare algoritmi di clustering (come K-Means) per raggruppare canzoni con caratteristiche simili.
- **Visualizzazione e interpretazione:**  
Rappresentare graficamente i cluster per una migliore comprensione della distribuzione dei brani.

AI & Music

Questa slide illustra i concetti chiave del progetto, che si basano su dati e algoritmi per analizzare e classificare le canzoni in base al mood.

Analisi

# Dataset Most Streamed Spotify Songs 2023

- Dataset pubblico, sorgente kaggle
- Composto da circa 950 canzoni, aventi come attributi →
  - rack\_name
  - artist(s)\_name
  - artist\_count
  - released\_year
  - released\_month
  - released\_day
  - in\_spotify\_playlists
  - in\_spotify\_charts
  - streams
- Abbiamo effettuato la pulizia del dataset dai valori nulli e da attributi non attinenti al nostro obiettivo
  - in\_apple\_playlists
  - in\_apple\_charts
  - in\_deezer\_playlists
  - in\_deezer\_charts
  - in\_shazam\_charts
  - bpm
  - key
  - mode
  - danceability\_%
  - valence\_%
  - energy\_%
  - acousticness\_%
  - instrumentalness\_%
  - liveness\_%
  - speechiness\_%



# Soluzione Proposta

- 1- Abbiamo scaricato e importato il dataset su colab

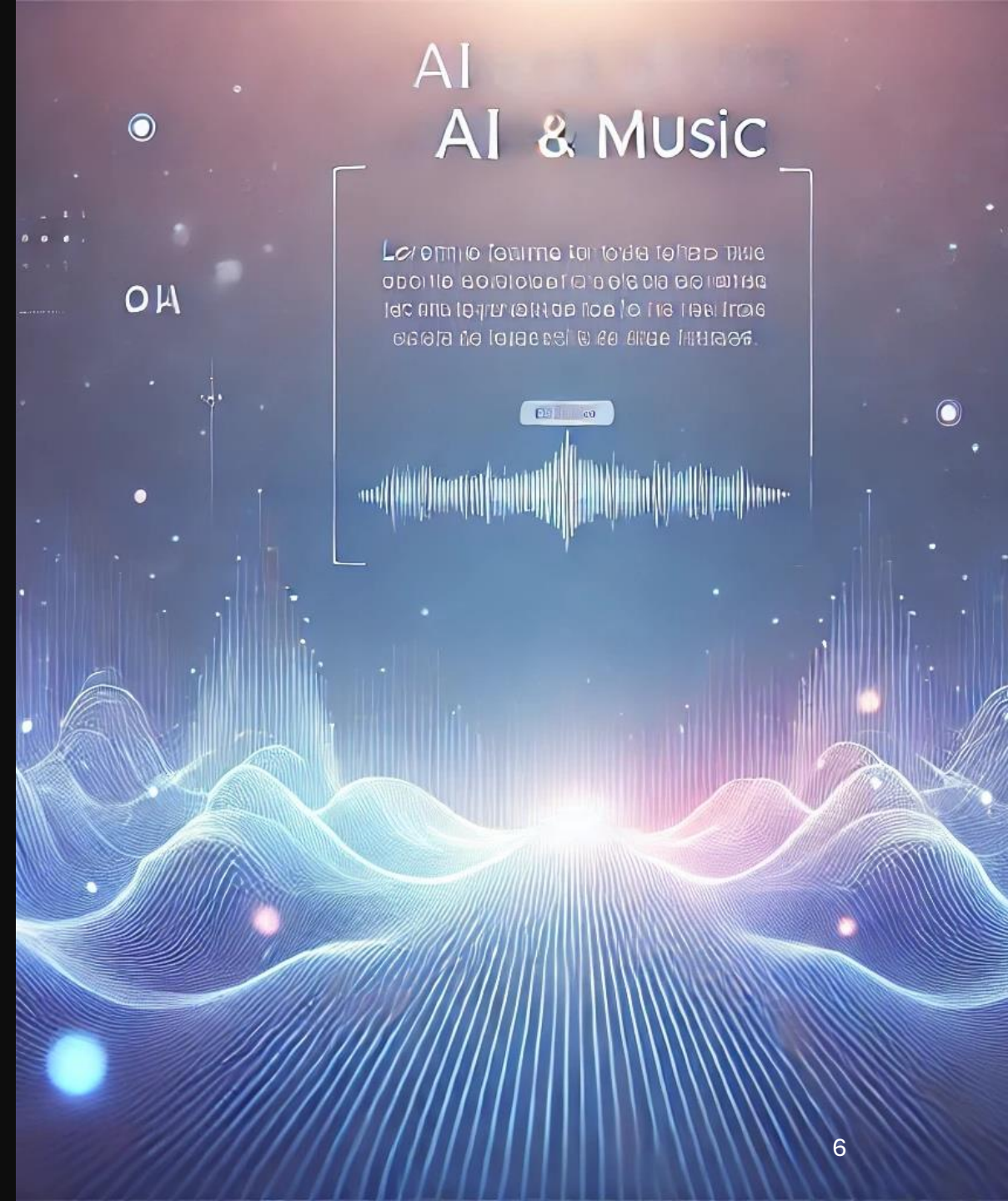
```
spotify_dataset = pd.read_csv('spotify-2023.csv',  
                               encoding='latin1')
```

- 2- Abbiamo filtrato il dataset, eliminando attributi non pertinenti e valori nulli

```
df = spotify_dataset.drop(['artist_count', 'in_spotify_charts',  
                           'in_apple_charts', 'in_deezer_charts', 'in_shazam_charts',  
                           'released_month', 'released_day'], axis=1)
```

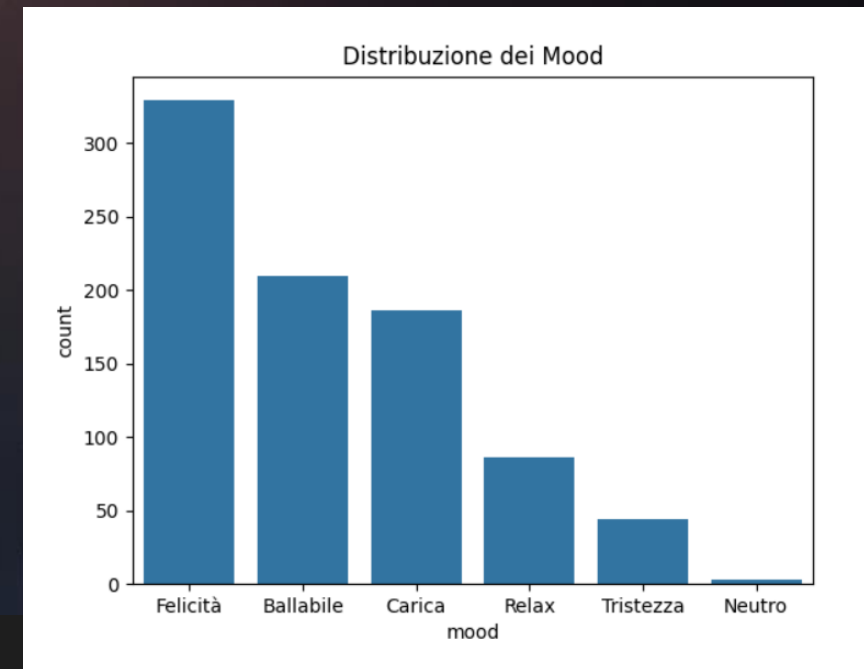
```
nan_mask = df.isna()
```

```
nan_count = nan_mask.sum()
```



- 3-Abbiamo assegnato un mood per ogni canzone, controllando i parametri: valence\_%, energy\_%, danceability\_%, acousticness\_% e bpm.

```
def assign_mood(row):
    if row['valence_%'] >= 65 and row['energy_%'] >= 55 and row['danceability_%'] >= 60:
        return 'Felicità' # Ridotte leggermente le soglie
    elif row['energy_%'] <= 60 and row['valence_%'] >= 20 and row['valence_%'] <= 70 and row['acousticness_%'] >= 35 and row['bpm'] <= 130:
        return 'Relax' # Maggiore tolleranza su bpm e acousticness
    elif row['valence_%'] <= 40 and row['energy_%'] <= 50 and row['acousticness_%'] >= 25:
        return 'Tristezza' # Soglie più ampie
    elif row['energy_%'] >= 60 and row['valence_%'] >= 25 and row['valence_%'] <= 80 and row['bpm'] >= 110:
        return 'Carica' # Aumentato il range di valence
    elif row['danceability_%'] >= 60 and row['bpm'] >= 80 and row['bpm'] <= 160 and row['energy_%'] >= 35:
        return 'Ballabile' # Maggiore flessibilità su energia e bpm
    else:
        # Logica di fallback per ridurre i Neutri
        if row['valence_%'] >= 50 or row['energy_%'] >= 50:
            return 'Felicità'
        elif row['acousticness_%'] >= 25 and row['bpm'] <= 110:
            return 'Relax'
        elif row['energy_%'] >= 55 and row['bpm'] >= 100:
            return 'Carica'
        elif row['danceability_%'] >= 50:
            return 'Ballabile'
        else:
            return 'Neutro'
```





- 4- Abbiamo filtrato le canzoni in base al mood selezionato dall'utente ed effettuato il cluster su altre caratteristiche

```
# Chiedi all'utente di scegliere un mood
user_mood = input("Scegli un mood (Felicità, Relax, Tristezza, Carica,Ballabile): ")

# Filtra il DataFrame per il mood selezionato
mood_set = test_set[test_set['mood'] == user_mood]

mood_st=mood_set.copy()

# Clustering: KMeans
n_clusters = f.dynamic_clusters(len(mood_st))
kmeans = KMeans(n_clusters, random_state=42)
mood_st['cluster'] = kmeans.fit_predict(scaled_features)
```

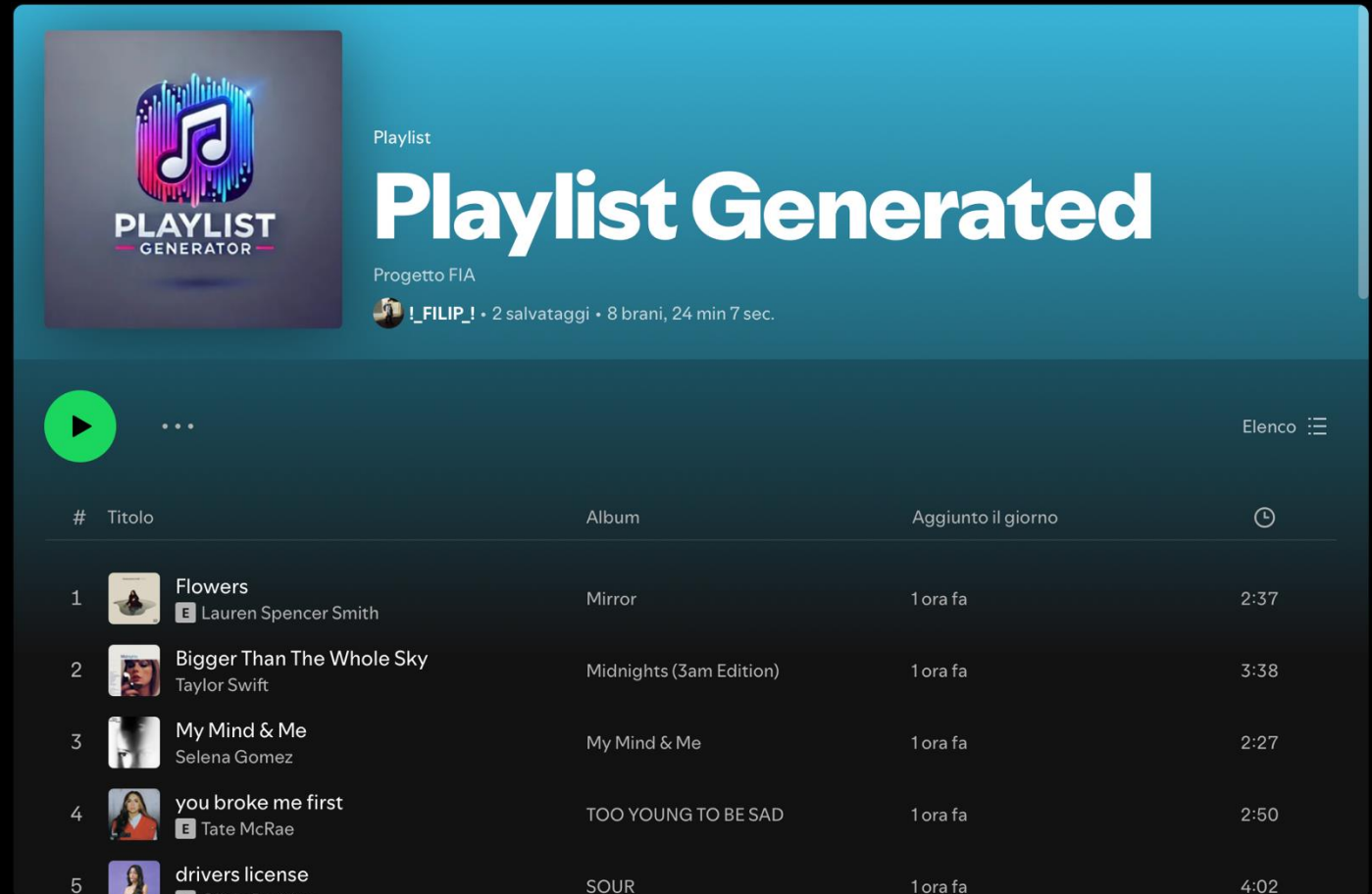
- 5- Sorteggiato uno dei cluster come playlist

```
indice=random.randint(1,kmeans.n_clusters)-1
print(indice)
playlist_scelta=mood_st[mood_st['cluster']==indice][['track_name', 'artist(s)_name', 'cluster']]
print(playlist_scelta)
```








# Collegamento Spotify

- Abbiamo collegato spotify al nostro progetto, per permettere di creare delle playlist reali, con le canzoni selezionate, su spotify



The screenshot shows a Spotify playlist interface. At the top, there's a header with a 'PLAYLIST GENERATOR' logo and the title 'Playlist Generated' by 'Progetto FIA'. Below the header, there's a play button and a menu icon. The main content is a table of songs in the playlist.

#	Titolo	Album	Aggiunto il giorno	
1	 <b>Flowers</b> Lauren Spencer Smith	Mirror	1 ora fa	2:37
2	 <b>Bigger Than The Whole Sky</b> Taylor Swift	Midnights (3am Edition)	1 ora fa	3:38
3	 <b>My Mind &amp; Me</b> Selena Gomez	My Mind & Me	1 ora fa	2:27
4	 <b>you broke me first</b> Tate McRae	TOO YOUNG TO BE SAD	1 ora fa	2:50
5	 <b>drivers license</b> Olivia Rodrigo	SOUR	1 ora fa	4:02

Abbiamo creato  
una playlist con  
all'interno le  
canzoni scelte,  
restituendo il link  
alla playlist

```
funzione principale che gestisce la playlist
def manage_playlist(playlist_id):
    # Verifica se la playlist è vuota
    if not is_playlist_empty(playlist_id):
        print("La playlist non è vuota, svuotando...")
        clear_playlist(playlist_id) # Svuota la playlist se non è vuota

    # Carica il CSV con i dati delle canzoni
    #df = pd.read_csv('Playlist_scelta.csv') # Assicurati che il percorso d

    # Assicurati che il CSV abbia le colonne "Nome" e "Artista" (o i nomi de
    tracks_to_add = [(row['track_name'], row['artist(s)_name']) for _, row i

    # Cerca le tracce e ottieni gli URI
    track_uris_to_add = [search_track(name, artist) for name, artist in trac
    track_uris_to_add = [uri for uri in track_uris_to_add if uri is not None

    # Aggiungi i brani alla playlist
    if track_uris_to_add:
        add_tracks_to_playlist(playlist_id, track_uris_to_add)
    # Costruisci il link Spotify
    playlist_link = f"https://open.spotify.com/playlist/{playlist_id}"
    return playlist_link
```





# Clustering

- Il clustering è una tecnica di machine learning non supervisionato che raggruppa dati simili in insiemi (cluster). Ogni gruppo contiene elementi con caratteristiche simili tra loro e diversi da quelli di altri gruppi. È utile per scoprire pattern nascosti o segmentare dati.



# Algoritmo K-Means

K-Means è uno degli algoritmi di clustering più popolari per la sua **semplicità ed efficienza**. Ecco come funziona:

1. **Definizione di K:** Si sceglie il numero di cluster (**K**) che si vuole trovare.
2. **Selezione iniziale:** L'algoritmo inizia scegliendo casualmente K punti nel dataset come centroidi (i “centri” dei cluster).
3. **Assegnazione ai cluster:** Ogni punto del dataset viene assegnato al cluster del centroide più vicino (calcolando la distanza, spesso con la distanza euclidea).
4. **Aggiornamento centroidi:** Si ricalcolano i centroidi spostandoli verso la **media dei punti** assegnati al loro cluster.
5. **Ripetizione:** I passi 3 e 4 si ripetono finché i cluster non smettono di cambiare o raggiungono un numero massimo di iterazioni.



# Pro e Contro di K-Means:

- **Vantaggi:**

- Semplice e veloce.
- Funziona bene con dataset di medie dimensioni.

- **Svantaggi:**

- Devi specificare il numero di cluster ( $K$ ) in anticipo.
- Sensibile ai valori iniziali dei centroidi (si possono ottenere risultati diversi).
- Funziona meglio con cluster di forma sferica e distribuzione omogenea.

# Implementazione

## Librerie Utilizzate:

- **Pandas:** Libreria per la manipolazione e analisi dei dati, ideale per lavorare con tabelle e dataset strutturati (DataFrame).
- **Sklearn:** Libreria per il machine learning, che offre strumenti per classificazione, regressione, clustering, e preprocessing dei dati. (Clustering K-Means)
- **Gradio:** Libreria per creare rapidamente interfacce grafiche web per modelli di machine learning e altre applicazioni Python.
- **Spotipy:** Libreria per interagire con l'API di Spotify, utile per recuperare dati su artisti, canzoni e playlist.



---

Per effettuare il cluster abbiamo generato delle nuove caratteristiche con delle interazioni tra i vari attributi:

- **Rapporto Ballabilità/Felicità (dance\_valence\_ratio):** rapporto tra danceability e valence
  - **Differenza Energia-Acustica (energy\_acoustic\_diff):** differenza tra energy e acousticness
  - **Combinazione Energia-Ballabilità (energy\_dance\_combo):** prodotto tra energy e danceability
  - **Combinazione Acustica-Felicità (acoustic\_valence\_combo):** prodotto tra acousticness e valence
- 

```
mood_st['dance_valence_ratio'] = mood_st['danceability_%'] / (mood_st['valence_%'] + 1e-5)
mood_st['energy_acoustic_diff'] = mood_st['energy_%'] - mood_st['acousticness_%']
mood_st['energy_dance_combo'] = mood_st['energy_%'] * mood_st['danceability_%']
mood_st['acoustic_valence_combo'] = mood_st['acousticness_%'] * mood_st['valence_%']

print(mood_st)
```

# Generatore di Playlist

Seleziona un mood

Tristezza

Clear

Submit

Playlist generata















track_name ▲	artist(s)_name ▲	counter ▲
Flowers	Lauren Spencer Smith	0
Bigger Than The Whole Sky	Taylor Swift	0
My Mind & Me	Selena Gomez	0
you broke me first	Tate McRae	0
drivers license	Olivia Rodrigo	0
Christmas Tree	V	0
traitor	Juan Cruz Toledo, Huilen Toledo	0
Bored	Billie Eilish	0
Revenge	XXXTENTACION	0
		0

Abbiamo applicato una  
semplice interfaccia grafica



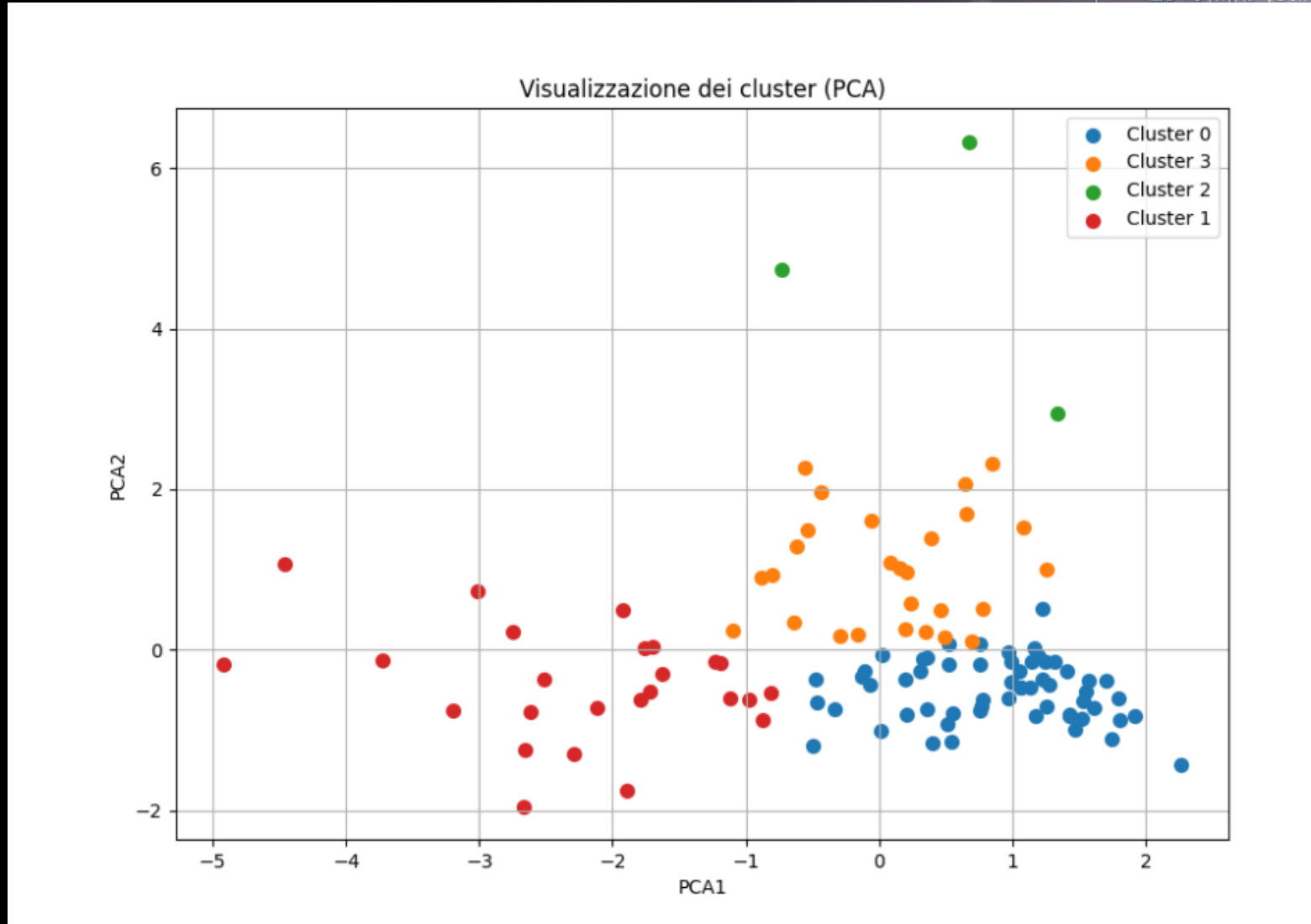
27/01/2025

17

 Playlist Generated				
#	Titolo	Album	Aggiunto il giorno	🕒
1	 <b>CAIRO</b> E KAROL G, Ovy On The Drums	MAÑANA SERÁ BONITO	21 secondi fa	3:18
2	 <b>New Jeans</b> NewJeans	NewJeans 'Super Shy'	21 secondi fa	1:48
3	 <b>Bebe Dame</b> Fuerza Regida, Grupo Frontera	Sigan Hablando	21 secondi fa	4:31
4	 <b>Leave The Door Open</b> Bruno Mars, Anderson .Paak, Silk Sonic	An Evening With Silk Sonic	21 secondi fa	4:02
5	 <b>Antes de Perderse</b> Duki	Temporada de Reggaetón 2	21 secondi fa	2:56
6	 <b>Mejor Que Yo</b> E Anuel AA, DJ Luian, Mambo Kingz	Mejor Que Yo	21 secondi fa	4:02
7	 <b>TQG</b> E KAROL G, Shakira	MAÑANA SERÁ BONITO	21 secondi fa	3:17
8	 <b>Te Felicito</b> Shakira, Rauw Alejandro	Te Felicito	21 secondi fa	2:52
9	 <b>Christmas (Baby Please Come Home)</b> Darlene Love	A Christmas Gift For You From Phil Spector	21 secondi fa	2:46
	 <b>Too Late</b> SZA	SOS Deluxe: LANA SOS Deluxe: LANA	21 secondi fa	✓ 2:44 ...
11	 <b>Seek &amp; Destroy</b> E SZA	SOS Deluxe: LANA	21 secondi fa	3:23
12	 <b>psychofreak (feat. WILLOW)</b> Camila Cabello, WILLOW	Familia	21 secondi fa	3:21
13	 <b>Stay Alive (Prod. SUGA of BTS)</b> Jung Kook	Stay Alive (Prod. SUGA of BTS)	21 secondi fa	3:30

Esempio playlist  
con mood «Carica»

# Esempio cluster



# CONCLUSIONE

Questo progetto ci ha permesso di avvicinarci ad un nuovo linguaggio, Python, ci ha portato a utilizzare librerie nuove, per la creazione dell'interfaccia grafica e per lavorare con spotify, permettendoci di creare, modificare e cancellare una playlist/canzoni. Ci ha aiutato a capire meglio il funzionamento del clustering e di come si lavora con i dataset.

